

Teamcenter 10.1

Utilities Reference

Proprietary and restricted rights notice

This software and related documentation are proprietary to Siemens Product Lifecycle Management Software Inc.

© 2013 Siemens Product Lifecycle Management Software Inc. All Rights Reserved.

Siemens and the Siemens logo are registered trademarks of Siemens AG. Teamcenter is a trademark or registered trademark of Siemens Product Lifecycle Management Software Inc. or its subsidiaries in the United States and in other countries. All other trademarks, registered trademarks, or service marks belong to their respective holders.

Contents

Proprietary and restricted rights notice	2
Getting started with Teamcenter utilities	1-1
Before you begin	1-1
Manually configuring your environment for Teamcenter utilities	1-1
Manually configure your UNIX environment for Teamcenter utilities	1-2
Manually configure your Windows environment for Teamcenter utilities	1-2
Log files produced by Teamcenter	1-2
System log files	1-3
Application log files	1-4
Manage password files	1-5
Preferences and variables used to control application logging	1-6
Syntax definitions	1-6
Configuration utilities	2-1
Preference management	2-1
Data access management	2-34
Business Modeler IDE	2-49
Localization	2-115
Attribute sharing	2-125
Organization	2-128
Object validation	2-140
Teamcenter reporting	2-143
Teamcenter interface	2-151
Product configuration utilities	3-1
Content management	3-1
Product structure maintenance	3-5
Effectivity mode	3-69
Product structure clearance analysis	3-71
Appearance Configuration	3-73
Teamcenter Rapid Start utilities	4-1
Collaborative Product Development utilities	5-1
Workflow utilities	6-1
Data sharing utilities	7-1
Visualization utilities	8-1
harvest_mmv_index	8-1
Repeatable Digital Validation (RDV) utilities	9-1

RDV cache maintenance	9-48
Manufacturing utilities	10-1
Classification utilities	11-1
Sample files for smlutility and icsutility	11-20
Understanding the smlutility and SML file format	11-20
Query utilities	12-1
Maintenance utilities	13-1
Installation	13-1
Audit Manager	13-13
Backup and Recovery	13-32
Dispatcher	13-38
Migration	13-46
Portfolio, Program, and Project Management	13-50
Server manager	13-57
Subscription Manager	13-62
System maintenance	13-64
Document management	13-107
Teamcenter Integration for NX utilities	14-1
Integration utilities	15-1
Teamcenter/Community Collaboration	15-1
Teamcenter/System Engineering and Requirements Management	15-7
Teamcenter Automotive Edition-GM Overlay utilities	16-1
Aerospace and Defense utilities	17-1
Materials Management and Substance Compliance	18-1
Consumer Packaged Goods utilities	19-1
Computer-aided engineering (CAE) utilities	20-1
Teamcenter mechatronics process management utilities	21-1
Volume and database management utilities	22-1
File Management System (FMS) utilities	22-57
Systems Engineering utilities	23-1
Index	Index-1

Chapter

1 *Getting started with Teamcenter utilities*

Before you begin

Unless otherwise noted in the description of a utility, the program files associated with the utilities described in this manual are located in the **bin** directory under the Teamcenter application root directory.

Prerequisites	Unless stated otherwise in the description of the utility, you must be a member of the dba group or a group that is granted dba privileges.
Enable the utilities	<p>Most of the command utilities are enabled by user authentication using the user ID and password supplied in the -u and -p arguments. If these arguments are not required, the command is enabled by default.</p> <p>If Security Services single sign-on (SSO) is enabled for your server, the -u and -p arguments are authenticated externally through SSO rather than being authenticated against the Teamcenter database. If you do not supply these arguments, the utility attempts to join an existing SSO session. If no session is found, you are prompted to enter a user ID and password.</p>
Configure the utilities	For information about configuring the utilities, see Manually configuring your environment for Teamcenter utilities .
Start the utilities	Each utility is initiated by entering its name and optional parameters.

Manually configuring your environment for Teamcenter utilities

Teamcenter administrators typically configure site workstations and computers so that users can log on without manually setting the environment. If this has not been done by your administrator, you must manually set the Teamcenter environment before you can run a session.

If you are unsure whether to perform this procedure, consult your administrator for additional information.

You configure utilities using environment variables.

TC_ROOT and **TC_DATA** are the only environment variable settings required to run the core Teamcenter application. These variables can be set automatically

at login. However, there are several stand-alone utilities such as **install** and **clearlocks** that also require that the entire Teamcenter environment be set.

If you are using Teamcenter Integration for NX, you must enter the following commands at the command line prompt:

```
set UGII_BASE_DIR=path to where NX6 is installed
set UGII_ROOT_DIR=%UGII_BASE_DIR%\ugii
set PATH=%UGII_ROOT_DIR%;%PATH%
```

Manually configure your UNIX environment for Teamcenter utilities

Note The following procedures use default path names. If other path names were specified during Teamcenter installation, use those path names instead. Consult your administrator for additional information.

Manually configuring the Teamcenter environment on UNIX systems requires sourcing the **tc_profilevars** and **tc_cshvars** scripts. To manually set the Teamcenter environment, enter one of the following sets of commands:

Bourne/Korn shell:

```
TC_ROOT=/usr/tc2007; export TC_ROOT
TC_DATA=/usr/tc2007/tcdata; export TC_DATA
. $TC_DATA/tc_profilevars
```

C shell:

```
setenv TC_ROOT /usr/tc2007
setenv TC_DATA /usr/tc2007/tcdata
source $TC_DATA/tc_cshvars
```

Sourcing the **\$TC_DATA/tc_cshvars** file creates a **csh** subshell in which Teamcenter environment variables are set.

Manually configure your Windows environment for Teamcenter utilities

Note The following procedures use default path names. If other path names were specified during Teamcenter installation, use those path names instead. Consult your administrator for additional information.

Manually configuring the Teamcenter environment on Windows systems requires running the **tc_profilevars.bat** script. This script is called automatically when exiting to an MS-DOS shell from the Teamcenter menu, but the environment can also be set manually. To manually set the Teamcenter environment, enter the following commands:

```
set TC_ROOT=c:\tc2007
set TC_DATA=c:\tc2007\tcdata
call %TC_DATA%\tc_profilevars
```

Log files produced by Teamcenter

Teamcenter utilities often produce log files.

Teamcenter creates two types of log files: system log files and application log files. *System log files* are used to record information about global system events; *application log files* are used to record information about specific Teamcenter applications.

Note In this context, *application* means either a Teamcenter application (such as Teamcenter or Structure Manager) or a Teamcenter utility (such as **clearlocks**).

System log files and application log files are stored in different directories and controlled by different preferences. Site preferences control system log files and are write protected. Site preferences must be set in the **tc_preferences.xml** preference file stored in the *TC_DATA* directory. Various user, group, and role preferences control application log files. These preferences can be managed using the **Options** dialog box, accessed from the **Edit** menu. Use the **Options** dialog box to search for preferences, set preference values, create new preferences, and remove existing preferences.

For additional information about using this dialog box, see the [Rich Client Interface Guide](#).

Best practices

Siemens PLM Software recommends that old log files be deleted. Log files can consume large amounts of hard disk space. Log file size should be monitored periodically and old log files deleted to recover hard disk space. Any required new log files can be recreated after old log files are deleted. Log files can be deleted while users are logged on to Teamcenter.

Siemens PLM Software recommends that directory write permissions are left open. It is extremely important to ensure that directories used to store log file have the required write permissions at all times. For example, it is especially important that Teamcenter be able to write to the **\$TC_LOG** directory if any system logging is enabled. Otherwise, the system repeatedly attempts to write to this directory and performance is affected.

For information about log files, see the [System Administration Guide](#).

System log files

The following log files record information about installation and global system events:

- Administration log file (**administration.log**)

Contains a record of the following Teamcenter system administration objects when they are created, modified, deleted, or released:

- o **user**
- o **group**
- o **group member creation**
- o **dataset type**
- o **tool creation**
- o **tool deletion**
- o **volume creation**
- o **volume deletion**

- o **modificaiton**
- o **move**
- o **grant access**
- o **revoke access**

The protected **TC_Administration_Logging** site preference enables administrative logging.

- Installation log file (**installdate-time.log**)

This log file is in the **install** directory under the application root directory. The *date-time* stamp represents the date and time Teamcenter Environment Manager was run. For example, **install0522241627.log** indicates that Teamcenter Environment Manager was run at 4:27 on February 24, 2005.

The protected **TC_Installation_Logging** site preference enables installation logging.

- POM utilities log file (**tc_install.log**)

Contains the standard output from the POM utilities called by Teamcenter Environment Manager. This log file is in the **logs** directory under the application root directory. The **logs** directory also contains the logs (and when POM utilities fail, **syslogs**) for utilities that Teamcenter Environment Manager calls.

- Security log file (**security.log**)

Contains a record of attempted protection violations by users. For example, logs are kept of attempts to open a dataset without read permission on that dataset.

The protected **TC_Security_Logging** site preference enables security logging. Security logging creates a record of invalid access of Teamcenter objects and writes the data to the **\$TC_LOG/security.log** file. Enabling security logging also requires creating a file named **security.log** in the **TC_DATA** directory. The system first checks for the existence of this file; if it exists, it checks the value of this preference. If set to **ON**, security logging is enabled.

- System log file (**system.log**)

Contains a record of global system events such as database shutdowns and system startup.

The protected **TC_System_Logging** site preference enables system logging.

Application log files

Each set of application log files consists of a journal file, monitor file, object log file and a **syslog** file. The **TC_Application_Logging** preference controls the logging of journal files. **syslog** files are always created and can not be suppressed.

Each application log file name is a concatenation of the application name, the OS process ID (PID), and a descriptive file extension. This ensures application log file names are unique for each session and prevents overwriting valuable troubleshooting information. The following is an example of Structure Manager log file names:

```
PSEPID.jnl
PSEPID.log
PSEPID.syslog
```


The following application log files are used by Siemens PLM Software support and development to troubleshoot and debug Teamcenter:

- **Journal files (.jnl)**
Contains diagnostic information and is intended for Siemens PLM Software use only.
- **Syslog files (.syslog)**
Contains diagnostic information and is intended for Siemens PLM Software use only.
- **Object log files (.log)**
Contains a record of Teamcenter objects (users, groups, volumes, and so on) created, modified, or deleted during the application session.

Manage password files

To provide the best password security, you can store an encrypted password in a designated file and directory location on a local disk. You create the file containing the encrypted password using Teamcenter Environment Manager (TEM) or the [install](#) utility. An environment variable contains the password string to be encrypted. The variable is designated in TEM or by an [install](#) utility argument. The environment variable is not maintained, it is used only during the encryption process to ensure the clear text password is not persisted.

For more information about password encryption, see the [install](#) utility.

Note You can update the encrypted Teamcenter user password using TEM or the [install](#) utility. However, this does not change the password in the Teamcenter database. This must be done manually.

The encryption process uses an AES 256-bit encryption key.

For information about managing the encryption key, see the [System Administration Guide](#).

The **-pf** argument provides enhanced password security by allowing you to place an unencrypted password in a text file and secure the file using operating system-level security. This is stronger security than is provided by the **-p** argument, in which passwords are placed on the utility program command line, allowing a user to run **ps -ef** to display all running utilities and gain access to the utilities' passwords.

The file must contain only the password. Do not include user names or other text. The password must be one line; new lines and carriage returns are considered a terminator. The password must also be in character encoding consistent with the processes reading it.

You must place the file on a local disk to ensure that access control is managed securely by the operating system representing the file.

- To prepare the password file on UNIX, run **chmod 400 file-name**.
- To prepare the password file on Windows, right-click the file and choose **Properties**, and then click the **Security** tab and ensure that **Administrators** is the only group with read access on the local machine.

Note

File access control becomes complicated when mapping between UNIX and Windows platforms.

- On UNIX, do not place the password file on disks mounted from other machines. You can run **df** to obtain a list of such disks.
- On Windows, do not place the password file on drives shared with other machines. Using a removal disk (**usb**) ensures that even local administrators only have access when the disk is physically present.

Preferences and variables used to control application logging

The following preferences and environment variable control application logging:

- **TC Application Logging** preference
Enables or suppresses application logging. This command can also be set in the rich client interface using the **Edit→Options** command.
This preference only enables or suppresses application logging for journal and monitor files; **syslog** files are always created and cannot be suppressed.
- **TC_Journalling** preference
Globally enables or suppresses creation of all journal files independently of monitor and **syslog** files. This command can also be set in the rich client interface using the **Edit→Options** command.
- **CLASSPATH** environment variable
Defines the directory for storing the rich client object log (**.log**) files when the **java.io.tmpdir** key is defined in the Virtual Machine **CLASSPATH** variable, as follows:

```
-Djavaio.tmpdir=  
path-to-temp-directory
```

Syntax definitions

This manual uses a set of conventions to define the syntax of Teamcenter commands, functions, and values. Following is a sample syntax format:

```
verify_tasks -u=user-id {-p=password | -pf=password-file}  
[-g=group-name] [-m={list | delete}] [-h]
```

The conventions are:

Bold Bold text represents words and symbols you must enter exactly as shown.

For example, you enter **verify_tasks** exactly as shown.

Italic Italic text represents values that you supply.

For example, you supply your values for *user-ID*, *password* and *group-name*.

	<p>A vertical bar (also called a pipe) represents a choice between mutually exclusive elements.</p> <p>For example, you must specify either the list value or the delete value for the -m argument.</p>
[]	<p>Brackets represent optional elements.</p> <p>For example, the -g=, -m= and -h arguments are optional.</p> <p>(Arguments not in brackets are mandatory. For example, the -u= argument is required.)</p>
{text}	<p>Braces surround mutually exclusive elements that are required.</p> <p>For example, a password value is required. You must use either the -p= or -pf= argument.</p>
...	<p>An ellipsis indicates that you can repeat the preceding element.</p>
text-text	<p>A hyphen separates two words that describe a single value.</p> <p>For example, <i>group-name</i> indicates that you input a single value.</p>

Following are examples of correct syntax for the **verify_tasks** command:

```
$TC_ROOT/bin/verify_tasks -u=dba -p=DBA
$TC_ROOT/bin/verify_tasks -u=dba -p=DBA -m=list
$TC_ROOT/bin/verify_tasks -u=dba -pf=passwords.txt -m=delete
$TC_ROOT/bin/verify_tasks -u=dba -p=DBA -h
```

Chapter

2 *Configuration utilities*

Preference management

preferences_manager

Migrates legacy preference files to the database. In addition, this utility can be used to convert legacy preference files to XML format and to import and export preferences to and from the database.

SYNTAX**preferences_manager****-u**=*user-name*{**-p**=*password* | **-pf**=*password-file*}**-g**=*group-name***-mode**= {**append** | **category** | **cleanup** | **cleanup_definitions** | **clear** | **export** | **generatexml** | **import** | **migrate** | **delete** | **remove** | **upgradexml**}
[-h]**ARGUMENTS****-u**

Specifies the user ID.

This is generally **infodba** or another user with administration privileges. If this argument is used without a value, the operating system user name is used.

Note

If Security Services single sign-on (SSO) is enabled for your server, the user and password arguments are authenticated externally through SSO rather than being authenticated against the Teamcenter database. If you do not supply these arguments, the utility attempts to join an existing SSO session. If no session is found, you are prompted to enter a user ID and password.

-p

Specifies the user's password.

If used without a value, the system assumes a null value. If this argument is not used, the system assumes the *user-ID* value to be the password.

This argument is mutually exclusive with the **-pf** argument.

-pf

Specifies the password file. If used without a value, the system assumes a null value. If this argument is not used, the system assumes the *user-ID* value to be the password.

For more information about managing password files, see [Manage password files](#).

This argument is mutually exclusive with the **-p** argument.

-g

Specifies the group associated with the user.

If used without a value, the user's default group is assumed.

-mode=append

Appends specified values to an existing preference in the database.

For more information about syntax specific to this mode, see [-mode=append](#).

-mode=category

Manages preference categories. The list of categories is separated by a delimiter using the **delimiter** option. The default delimiter is a comma.

For more information about syntax specific to this mode, see [-mode=category](#).

-mode=cleanup

Removes stale preferences not removed by other administrative tasks.

For more information about syntax specific to this mode, see [-mode=cleanup](#).

-mode=cleanup_definitions

Deletes redundant preference definitions from the preference definition file stored in the database.

Redundant preference definitions are created when a saved query is performed in Teamcenter Engineering. Such preferences are named *_**query**_*.

For more information about syntax specific to this mode, see [-mode=cleanup_definitions](#).

-mode=clear

Removes all preferences of the specified scope.

For more information about syntax specific to this mode, see [-mode=clear](#).

-mode=export

Exports scope-based preferences to a specified output file.

For more information about syntax specific to this mode, see [-mode=export](#).

-mode=generatexml

Converts a specified legacy preference file to XML format in the specified output location.

For more information about syntax specific to this mode, see [-mode=generatexml](#).

-mode=import

Imports a specified XML or legacy preference file into the database.

For more information about syntax specific to this mode, see [-mode=import](#).

-mode=migrate

Migrates preferences from legacy preference files into the database.

For more information about syntax specific to this mode, see [-mode=migrate](#).

-mode=delete

Deletes the specified non-foundation preference definitions from the system. Foundation preferences are the ones available from a basic Teamcenter installation.

For more information about syntax specific to this mode, see [-mode=delete](#).

-mode=remove

Removes specified preferences of the specified scope from the database.

For more information about syntax specific to this mode, see [-mode=remove](#).

-mode=upgradexml

Upgrades a preference file from a specified format to the current format.

For more information about syntax specific to this mode, see [-mode=upgradexml](#).

-h

Displays help for this utility.

ENVIRONMENT

As specified in [Manually configuring your environment for Teamcenter utilities](#).

FILES

As specified in [Log files produced by Teamcenter](#).

RESTRICTIONS

Creation and deletion of preferences is governed by the following rules:

- If you are a user, you can only create or delete user preferences.
- If you are a group administrator, you can create or delete group, role, and user preferences.
- If you are a site administrator, you can create or delete site, group, role, and user preferences.

EXAMPLES

- To migrate the legacy site, user, role, and group preferences files, enter the following command on a single line:

```
preferences_manager -u=user-id -p=password -g=dba  
-mode=migrate -dir=some-directory
```

The structure of *some-directory* can be:

```
.iman_env (Site preference file)  
upfiles (Folder containing user preference files.  
The file names are <user name>.iman_env )  
rpfiles (Folder containing role preference files.  
The file names are <role name>.iman_env)  
gpfiles (Folder containing group preference files.  
The file names are <group name>.iman_env)
```

- To generate the site preferences XML file from the legacy site preference file, (can be a site, user, role, or group), enter the following command on a single line:

```
preferences_manager -u=user-id -p=password -g=dba  
-mode=generatexml -context=Teamcenter  
-file=legacy-preference-file -out_file=C:\temp\site_pref.xml
```

- To import the site preferences in an XML file, skipping the processing for all preferences in the XML file that exist in the database, enter the following command on a single line:

```
preferences_manager -u=infodba -p=password -g=dba -mode=import  
-scope=SITE -file=C:\temp\site_pref.xml -action=SKIP
```

- To import the site preferences in an XML file, overriding the values of preferences in the database with the values assigned to the same preference in the XML file, enter the following command on a single line:

```
preferences_manager -u=user-id -p=password -g=dba -mode=import  
-scope=SITE -file=C:\temp\site_pref.xml -action=OVERRIDE
```


- To import the site preferences in an XML file, merging the values of preferences in the database with the values assigned to the same preference in the XML file, enter the following command on a single line:

```
preferences_manager -u=infodba -p=password -g=dba -mode=import
-scope=SITE -file=C:\temp\site_pref.xml -action=MERGE
```

- To import the preferences in the site legacy preference file, overriding the values of the preferences in the database with the values of the same preference in the legacy file, enter the following command on a single line:

```
preferences_manager -u=infodba -p=password -g=dba -mode=import
-scope=SITE -file=C:\temp\site_pref -action=OVERRIDE
```

- To import a preference (specified on the command line) and override the values in the database with the values specified for the preference on the command line, enter the following command on a single line:

```
preferences_manager -u=infodba -p=password -g=dba
-mode=import -scope=SITE -file=C:\temp\site_pref
-preference=TestPreference -values=Val1,Val2,Val3 -action=OVERRIDE
```

- To export all user preferences in the database for the user **smith**, enter the following command on a single line:

```
preferences_manager -u=smith -p=password -g=design -mode=export
-scope=USER -out_file=C:\temp\smith.xml
```

Note In this example, the utility must be run by the user.

- To export all group preferences in the database for the logged-on group of user **smith**, enter the following command on a single line:

```
preferences_manager -u=smith -p=password -g=design
-mode=export -scope=GROUP -out_file=C:\temp\design.xml
```

- To export preferences specified in an input file, enter the following command on a single line:

```
preferences_manager -u=smith -p=password -g=design
-mode=export -file=C:\temp\input_file.txt
-out_file=c:\temp\exported_preferences.xml
```

- To generate an XML preference file in the Teamcenter context, enter the following command on a single line:

```
preferences_manager -u=smith -p=password -g=design
-mode=generatexml -file=C:\temp\input_file.txt
-context=Teamcenter -out_file=c:\temp\exported_preferences.xml
```

- For the system administrator to export user preferences for Teamcenter user **smith**, enter the following command on a single line:

```
preferences_manager -u=infodba -p=password -g=dba
-mode=export -scope=USER -target=smith -out_file=c:\temp\some-file
```

- To generate an XML file, enter the following command on a single line:

```
preferences_manager -u=user-id -p=password -g=dba
-mode=generatexml -file=IMAN_DATA\gpfiles\design.iman_env
-scope=GROUP -out_file=C:\temp\design.xml
```

This command can be executed by any user, but the actual import is governed by the user's privileges.

- For the system administrator to remove the **pref1** and **pref2** preferences for the user **smith**, enter the following command on a single line:

```
preferences_manager -u=infodba -p=infodba -g=dba  
-mode=remove -scope=USER -target=smith -preferences=pref1,pref2
```

- For the system administrator to remove the **pref1** and **pref2** preferences for the **Engineering** group, enter the following command on a single line:

```
preferences_manager -u=infodba -p=infodba -g=dba  
-mode=remove -scope=GROUP -target=Engineering -preferences=pref1,pref2
```

- For the system administrator to append the **HRN_Cavity** value to the existing **Connection:HRN_Core,HRN_GeneralWire** values on the **Connected_ToRules** preference, enter the following command on a single line:

```
preferences_manager -u=infodba -p=infodba -g=dba  
-mode=append -scope=SITE -preference=Connected_ToRules  
-prefix="Connection:" -values="HRN_Cavity" -delimiter=","
```

-mode=append

Appends specified values to an existing preference in the database.

SYNTAX

```
preferences_manager
-u=user-name
{-p=password | -pf=password-file}
-g=group-name
-mode=append
-scope= {SITE | GROUP | ROLE | USER}
-preference=preference-name
[-prefix=prefix-to-be-searched-in-value]
-values=values-to-be-appended
-delimiter=delimiter-to-be-used-for-append
[-h]
```

ARGUMENTS**-u**

Specifies the user ID.

This is generally **infodba** or another user with administration privileges. If this argument is used without a value, the operating system user name is used.

Note

If Security Services single sign-on (SSO) is enabled for your server, the user and password arguments are authenticated externally through SSO rather than being authenticated against the Teamcenter database. If you do not supply these arguments, the utility attempts to join an existing SSO session. If no session is found, you are prompted to enter a user ID and password.

-p

Specifies the user's password.

If used without a value, the system assumes a null value. If this argument is not used, the system assumes the *user-ID* value to be the password.

This argument is mutually exclusive with the **-pf** argument.

-pf

Specifies the password file. If used without a value, the system assumes a null value. If this argument is not used, the system assumes the *user-ID* value to be the password.

For more information about managing password files, see [Manage password files](#).

This argument is mutually exclusive with the **-p** argument.

-g

Specifies the group associated with the user.

If used without a value, the user's default group is assumed.

-mode=append

Appends specified values to an existing preference in the database.

-scope

Specifies the location to which the specified preferences are appended. It is created in the specified location. Valid values are:

- **SITE**

Appends the values to preference at the site location. This means that the logged-on user must be a system administrator.

- **GROUP**

Appends the values to the preference at the group location. This means that the logged-on user must be a group administrator.

- **ROLE**

Appends the values to the preference at the role location. This means that the logged-on user must be a group administrator.

- **USER**

Appends the values to the preference at the user location.

Note In all these cases, if the preference does not exist at the location, the preference instance is created at the location with the specified value.

-preference

Specifies the preference to be used for append. If the preference is not found, it is created only if already defined in the database, and only if its protection scope allows for a definition at the given location.

-prefix

Specifies the prefix to be searched in the values. If the prefix is found, the value is appended to it. If the prefix is not specified, values separated by the delimiter are appended individually to the end of the values list.

-values

Specifies the values to append to the existing preference values.

-delimiter

Specifies the delimiter between the values.

-h

Displays help for this utility.

-mode=category

Manages preference categories. The list of categories is separated by a delimiter using the **delimiter** option. The default delimiter is a comma.

SYNTAX

preferences_manager

-u=*user-name*

{**-p**=*password* | **-pf**=*password-file*}

-g=*group-name*

-mode=category

-action= CREATE

-categories=*category-names*

[**-delimiter**=*delimiter-used-between-categories*]

[**-h**]

ARGUMENTS

-u

Specifies the user ID.

This is generally **infodba** or another user with administration privileges. If this argument is used without a value, the operating system user name is used.

Note If Security Services single sign-on (SSO) is enabled for your server, the user and password arguments are authenticated externally through SSO rather than being authenticated against the Teamcenter database. If you do not supply these arguments, the utility attempts to join an existing SSO session. If no session is found, you are prompted to enter a user ID and password.

-p

Specifies the user's password.

If used without a value, the system assumes a null value. If this argument is not used, the system assumes the *user-ID* value to be the password.

This argument is mutually exclusive with the **-pf** argument.

-pf

Specifies the password file. If used without a value, the system assumes a null value. If this argument is not used, the system assumes the *user-ID* value to be the password.

For more information about managing password files, see [Manage password files](#).

This argument is mutually exclusive with the **-p** argument.

-g

Specifies the group associated with the user.

If used without a value, the user's default group is assumed.

-mode=category

Manages preference categories.

Note This mode requires system administration privileges.

-action

Specifies the action performed upon the category. The only valid value is **CREATE**.

Note You can delete categories from the **Options** dialog box in the rich client.

-categories

Specifies categories to be created. Categories can be separated by a specified delimiter.

-delimiter

Specifies the delimiter used between categories. If not specified, the default delimiter is a comma.

-h

Displays help for this utility.

RESTRICTIONS

Only system administrators can create new categories using the **-mode=category** option.

-mode=cleanup

Removes stale preferences not removed by other administrative tasks.

SYNTAX

```
preferences_manager
-u=admin-username
{-p=password | -pf=password-file}
-g=admin-group
-mode=cleanup
[-target= {user-ID | role-ID | group-ID}]
[-dry_run]
[-exception_file=path to the exception file]
[-report_file=full-path-to-report-file]
[-h]
```

ARGUMENTS**-u**

Specifies the user ID.

This is generally **infodba** or another user with administration privileges. If this argument is used without a value, the operating system user name is used.

Note If Security Services single sign-on (SSO) is enabled for your server, the user and password arguments are authenticated externally through SSO rather than being authenticated against the Teamcenter database. If you do not supply these arguments, the utility attempts to join an existing SSO session. If no session is found, you are prompted to enter a user ID and password.

-p

Specifies the user's password.

If used without a value, the system assumes a null value. If this argument is not used, the system assumes the *user-ID* value to be the password.

This argument is mutually exclusive with the **-pf** argument.

-pf

Specifies the password file. If used without a value, the system assumes a null value. If this argument is not used, the system assumes the *user-ID* value to be the password.

For more information about managing password files, see [Manage password files](#).

This argument is mutually exclusive with the **-p** argument.

-g

Specifies the group associated with the user.

If used without a value, the user's default group is assumed.

-mode=cleanup

Deletes redundant preferences.

Note This mode requires system administration privileges.

-target

Specifies the user or role or group ID whose preferences are to be deleted. Must be a valid user, role or group ID.

If not specified, the cleanup applies to all preferences (including the site location).

-dry_run

If specified, no preferences are removed.

The system only parses the preferences to be removed and prints the findings in the specified report file.

-exception_file

Specifies the full path to the file that contains names of preferences (one per line) that are to be excluded from the cleanup operation. The information can be obtained from a run in **dry_run** mode.

If the argument is not specified, the system deletes all redundant preferences.

-report_file

Specifies the full path to the file containing logging information on the task.

If the argument is not provided, the system creates a default file.

-h

Displays help for this utility.

-mode=cleanup_definitions

Deletes redundant preference definitions from the preference definition file stored in the database.

Redundant preference definitions are created when a saved query is performed in Engineering Process Management. Such preferences are named ***_query_***.

SYNTAX**preferences_manager**

-u=*user-name*

{**-p**=*password* | **-pf**=*password-file*}

-g=*group-name*

-mode=cleanup_definitions

[-h]

ARGUMENTS**-u**

Specifies the user ID.

This is generally **infodba** or another user with administration privileges. If this argument is used without a value, the operating system user name is used.

Note

If Security Services single sign-on (SSO) is enabled for your server, the user and password arguments are authenticated externally through SSO rather than being authenticated against the Teamcenter database. If you do not supply these arguments, the utility attempts to join an existing SSO session. If no session is found, you are prompted to enter a user ID and password.

-p

Specifies the user's password.

If used without a value, the system assumes a null value. If this argument is not used, the system assumes the *user-ID* value to be the password.

This argument is mutually exclusive with the **-pf** argument.

-pf

Specifies the password file. If used without a value, the system assumes a null value. If this argument is not used, the system assumes the *user-ID* value to be the password.

For more information about managing password files, see [Manage password files](#).

This argument is mutually exclusive with the **-p** argument.

-g

Specifies the group associated with the user.

If used without a value, the user's default group is assumed.

-mode=cleanup_definitions

Deletes redundant preference definitions from the preference definition file stored in the database.

-h

Displays help for this utility.

-mode=clear

Removes all the preferences in the database for the specified scope and/or targets.

SYNTAX

```
preferences_manager
-u=user-name
{-p=password | -pf=password-file}
-g=group-name
-mode=clear
{-scope= {SITE | GROUP | ROLE | USER}
[-target=user-ID | role-ID | group-ID]
|
[-u_target= list-of-user-IDs] [-r_target= list-of-role-IDs]
-g_target=list-of-group-IDs]
[-h]
```

ARGUMENTS**-u**

Specifies the user ID.

This is generally **infodba** or another user with administration privileges. If this argument is used without a value, the operating system user name is used.

Note If Security Services single sign-on (SSO) is enabled for your server, the user and password arguments are authenticated externally through SSO rather than being authenticated against the Teamcenter database. If you do not supply these arguments, the utility attempts to join an existing SSO session. If no session is found, you are prompted to enter a user ID and password.

-p

Specifies the user's password.

If used without a value, the system assumes a null value. If this argument is not used, the system assumes the *user-ID* value to be the password.

This argument is mutually exclusive with the **-pf** argument.

-pf

Specifies the password file. If used without a value, the system assumes a null value. If this argument is not used, the system assumes the *user-ID* value to be the password.

For more information about managing password files, see [Manage password files](#).

This argument is mutually exclusive with the **-p** argument.

-g

Specifies the group associated with the user.

If used without a value, the user's default group is assumed.

-mode=clear

Removes all preferences of the specified scope.

For more information about syntax specific to this mode, see [-mode=clear](#).

Note This mode requires system administration privileges.

-scope

Specifies the location from which the specified preferences are cleared. It is created in the specified location. Valid values are:

- **SITE**

Clears only site overlay preferences (that is, only the site preferences that have been modified at a site). This means that the logged-on user must have system administrator privileges.

- **GROUP**

Clears all the group preferences of the current logged-on group.

- **ROLE**

Clears all the role preferences of the current logged-on role. This means that the logged-on user must have group administrator privileges.

- **USER**

Clears all the user preferences of the current logged-on user.

-target

Specifies the user or role or group ID of the user whose preferences are to be cleared.

This option can only be used with the **-scope** option.

-u-target

Specifies the list of users for which all preference instances are to be cleared. It cannot be used with the **-scope** argument but can be used with the **-r_target** and **-g_target** arguments. Entries are the IDs of users (separated by a comma) for which the logged-on user has privileges.

-r-target

Specifies the list of roles for which all preference instances are to be cleared. It cannot be used with the **-scope** argument but can be used with the **-u_target** and **-g_target** arguments. Entries are the IDs of roles (separated by a comma) for which the logged-on user has privileges.

-g-target

Specifies the list of groups for which all preference instances are to be cleared. It cannot be used with the **-scope** argument, but can be used with the **-r_target** and **-u_target** arguments. Entries are the IDs of groups (separated by a comma) for which the logged-on user has privileges.

-h

Displays help for this utility.

-mode=export

Exports scope-based preferences to a specified output file.

SYNTAX**preferences_manager**

-u=*user-name*

{**-p**=*password* | **-pf**=*password-file*}

-g=*group-name*

-mode=export

[**-scope**= {**SITE** | **GROUP** | **ROLE** | **USER**}]

[**-target**= {*user-ID* | *role-ID* | *group-ID*}]

[**-file**=*input-file*]

[**-categories**=*comma-separated-categories*]

[**-delimiter**=*value-delimiter*]

[**-out_file**=*output-file-name*]

[**-report_file**=*full-path-to-report-file*]

[**-h**]

ARGUMENTS**-u**

Specifies the user ID.

This is generally **infodba** or another user with administration privileges. If this argument is used without a value, the operating system user name is used.

Note If Security Services single sign-on (SSO) is enabled for your server, the user and password arguments are authenticated externally through SSO rather than being authenticated against the Teamcenter database. If you do not supply these arguments, the utility attempts to join an existing SSO session. If no session is found, you are prompted to enter a user ID and password.

-p

Specifies the user's password.

If used without a value, the system assumes a null value. If this argument is not used, the system assumes the *user-ID* value to be the password.

This argument is mutually exclusive with the **-pf** argument.

-pf

Specifies the password file. If used without a value, the system assumes a null value. If this argument is not used, the system assumes the *user-ID* value to be the password.

For more information about managing password files, see [Manage password files](#).

This argument is mutually exclusive with the **-p** argument.

-g

Specifies the group associated with the user.

If used without a value, the user's default group is assumed.

-mode=export

Exports scope-based preferences to a specified output file.

Note This mode requires system administration privileges.

-scope

Specifies the location to which the specified preferences are exported. It is created in the specified location. Valid values are:

- **SITE**
Only the site preferences matching the specified criteria are considered for export.
- **GROUP**
Only the group preferences (of the current logged-on group) matching the specified criteria are considered for export
- **ROLE**
Only the role preferences (of the current logged-on role) matching the specified criteria are considered for export
- **USER**
Only the user preferences (of the current logged-on user) matching the specified criteria are considered for export.

-target

Specifies the user or role or group ID whose preferences are to be exported. Must be a valid user, role or group ID.

If not specified, the export applies to all preferences (including the site location).

-file

Input file specifying the preferences to be exported. This file contains the preference names (in each line). For example:

Item_show_relations

Item_DefaultChildProperties

-categories

Specifies the categories to export. Categories are a comma-separated list, unless a different delimiter is provided with the **-delimiter** option. This option is ignored if the **-file** option is provided.

-delimiter

Specifies the delimiter to be used for the categories. The default delimiter is a comma if the **-delimiter** option is not specified.

-out_file

Specifies the file to which preferences are exported. The output is generated in **XML** format.

-report_file

Specifies the file that contains logging output. If not specified, a default file is created in the current directory.

-h

Displays help for this utility.

-mode=generatexml

Converts a specified legacy preference file to XML format in the specified output location.

SYNTAX

preferences_manager
-u=*user-name*
 {-**p**=*password* | -**pf**=*password-file*}
-g=*group-name*
-mode=generatexml
-file=*file-containing-preferences*
-context=*Teamcenter* | *NXManagerUnigraphics*
-out_file=*output-file-name*
[-h]

ARGUMENTS**-u**

Specifies the user ID.

This is generally **infodba** or another user with administration privileges. If this argument is used without a value, the operating system user name is used.

Note If Security Services single sign-on (SSO) is enabled for your server, the user and password arguments are authenticated externally through SSO rather than being authenticated against the Teamcenter database. If you do not supply these arguments, the utility attempts to join an existing SSO session. If no session is found, you are prompted to enter a user ID and password.

-p

Specifies the user's password.

If used without a value, the system assumes a null value. If this argument is not used, the system assumes the *user-ID* value to be the password.

This argument is mutually exclusive with the **-pf** argument.

-pf

Specifies the password file. If used without a value, the system assumes a null value. If this argument is not used, the system assumes the *user-ID* value to be the password.

For more information about managing password files, see [Manage password files](#).

This argument is mutually exclusive with the **-p** argument.

-g

Specifies the group associated with the user.

If used without a value, the user's default group is assumed.

-mode=generatexml

Converts a specified legacy preference file to XML format in the specified output location.

Note This mode requires system administration privileges.

-file

Specifies the preference file containing the preferences to be converted.

-context

Specifies the context. Valid values are:

- **Teamcenter**
- **NXManagerUnigraphics**

-out_file

Specifies the file to which the converted preferences are exported.

-h

Displays help for this utility.

-mode=import

Imports a specified XML or legacy preference file into the database.

SYNTAX**preferences_manager**

```

-u=user-name
{-p=password | -pf=password-file}
-g=group-name
-mode=import
{[-scope= {SITE | GROUP | ROLE | USER}]
[-target= {user-ID | role-ID | group-ID}]
|
[-u_target= list-of-user-IDs] [-r_target= list-of-role-IDs]
[-g_target= list-of-group-IDs]
|
[-file=input-file] [-preview]
[-categories=categories-to-import] [-delimiter=delimiter]
|
[-preference=preference-name] [-values=comma-separated-values]
[-delimiter=delimiter]}
[-protection_scope=default-protection-scope] [-enable_environment=activate]
-action= {SKIP | OVERRIDE | MERGE }
-report_file=file-name
[-h]

```

ARGUMENTS**-u**

Specifies the user ID.

This is generally **infodba** or another user with administration privileges. If this argument is used without a value, the operating system user name is used.

Note If Security Services single sign-on (SSO) is enabled for your server, the user and password arguments are authenticated externally through SSO rather than being authenticated against the Teamcenter database. If you do not supply these arguments, the utility attempts to join an existing SSO session. If no session is found, you are prompted to enter a user ID and password.

-p

Specifies the user's password.

If used without a value, the system assumes a null value. If this argument is not used, the system assumes the *user-ID* value to be the password.

This argument is mutually exclusive with the **-pf** argument.

-pf

Specifies the password file. If used without a value, the system assumes a null value. If this argument is not used, the system assumes the *user-ID* value to be the password.

For more information about managing password files, see [Manage password files](#).

This argument is mutually exclusive with the **-p** argument.

-g

Specifies the group associated with the user.

If used without a value, the user's default group is assumed.

-mode=import

Imports a specified XML or legacy preference file into the database.

-target

Specifies the user or role or group ID whose preferences are to be imported. Must be a valid user, role or group ID.

If not specified, the import applies to all preferences (including the site location).

-u-target

Specifies the list of users whose preference instances are to be imported. This argument can be used with the **-r_target** and **-g_target** arguments. Entries are the IDs of users (separated by a comma) for which the logged-on user has privileges.

-r-target

Specifies the list of roles whose preference instances are to be imported. This argument can be used with the **-u_target** and **-g_target** arguments. Entries are the IDs of roles (separated by a comma) for which the logged-on user has privileges.

-g-target

Specifies the list of groups whose preference instances are to be imported. This argument can be used with the **-r_target** and **-u_target** arguments. Entries are the IDs of groups (separated by a comma) for which the logged-on user has privileges.

-preference

Specifies the preference name that has to be imported to the database. This works only when the **-file** option is not specified. The preference must have already been defined in the system for this option to work.

-scope

Specifies the modified (if needed) protection scope to give to the preference specified in the **-preference** option when the preference is a hierarchical preference already defined in the database, and when the logged-on user is a system administrator.

Valid values are:

- **SITE**

Clears only site overlay preferences (that is, only the site preferences that have been modified at a site). This means that the logged-on user must have system administrator privileges.

- **GROUP**

Clears all the group preferences of the current logged-on group.

- **ROLE**

Clears all the role preferences of the current logged-on role. This means that the logged-on user must have group administrator privileges.

- **USER**

Clears all the user preferences of the current logged-on user.

-enable_env

Activates the specified environment.

-values

Specifies the values for the preference specified in the **-preference** option. This can be comma-separated values if the **-delimiter** option is not specified. In order to specify a delimiter other than comma, use the **-delimiter** option. This option is valid only with the **-preference** option. If needed, values can be surrounded by double quotation marks. For example, "my value".

-delimiter

Specifies the delimiter to be used either for the values (when used in conjunction with the **-values** option) or for the categories (when used with the **-categories** option). The default delimiter is comma if the **-delimiter** option is not specified.

-categories

Specifies the categories to import. Categories are a comma-separated list, unless specified through the **-delimiter** option.

-action

Indicates the action to be taken if a preference exists in the database with a different value. Valid values are:

- **SKIP**

The preference values in the database are untouched.

- **OVERRIDE**

The preference values in the database are overridden with the new values in the input file

- **MERGE**

Merges the values in the database with the values in the input file (that is, the union of values in the database and input file).

-report_file

Specifies the file that to which import results are logged. If not specified, a default file is created. The report file logs the following results:

- That the import file contains only user scope-protected preferences.
- That the import file contains role scope-protected preferences. These preferences were not imported for the given users (and a warning was printed in the output report).
- That the import file contains group scope-protected preferences. These preferences were not imported for the given users/roles (and a warning was printed in the output report).
- That the import file contains some scope-protected preferences or system preferences. These preferences were not imported for the given users/roles/groups (and a warning was printed in the output report).

- That the import file contains preferences that are not yet declared at the site level. These preferences were not imported for the given users/roles/groups (and a warning is printed in the output report).

-h

Displays help for this utility.

-mode=migrate

Migrates preferences from legacy preference files into the database.

SYNTAX

```
preferences_manager
-u=user-name
{-p=password | -pf=password-file}
-g=group-name
-mode=migrate
-dir=directory
[-h]
```

ARGUMENTS**-u**

Specifies the user ID.

This is generally **infodba** or another user with administration privileges. If this argument is used without a value, the operating system user name is used.

Note

If Security Services single sign-on (SSO) is enabled for your server, the user and password arguments are authenticated externally through SSO rather than being authenticated against the Teamcenter database. If you do not supply these arguments, the utility attempts to join an existing SSO session. If no session is found, you are prompted to enter a user ID and password.

-p

Specifies the user's password.

If used without a value, the system assumes a null value. If this argument is not used, the system assumes the *user-ID* value to be the password.

This argument is mutually exclusive with the **-pf** argument.

-pf

Specifies the password file. If used without a value, the system assumes a null value. If this argument is not used, the system assumes the *user-ID* value to be the password.

For more information about managing password files, see [Manage password files](#).

This argument is mutually exclusive with the **-p** argument.

-g

Specifies the group associated with the user.

If used without a value, the user's default group is assumed.

-mode=migrate

Migrates the preferences to the database.

-dir

Points to the directory containing the legacy site, group, role and user preference files. If the directory contains the legacy site preference file, all preferences in this file are migrated to the database. Additionally, the following rules apply for migrating user, role, and group preferences.

- If the **TC_GROUP_PFILE**, **TC_ROLE_PFILE** and **TC_USER_PFILE** environment variables are set, then the legacy files for group, role, and user are picked up from these directories.
- If these environment variables are not specified and the directory supplied contains the **fpfiles**, **rpfiles** and **upfiles** subdirectories, they are used for migration. To import all of the site, user, role and group preferences, the user must be a system administrator.

-h

Displays help for this utility.

-mode=delete

Deletes the specified non-foundation preference definitions from the system. Foundation preferences are the ones available from a basic Teamcenter installation.

SYNTAX**preferences_manager**

-u=*user-name*

{**-p**=*password* | **-pf**=*password-file*}

-g=*group-name*

-mode=delete

[**-preferences**=*preference_name-1,preference_name-2, ..., preference_name-n* |

-exception_file=*exception-file-name*]

[**-file**=*file-name*]

[**-dry_run**]

[**-report_file**= *report-file-name*]

[**-h**]

ARGUMENTS**-u**

Specifies the user ID.

This is generally **infodba** or another user with administration privileges. If this argument is used without a value, the operating system user name is used.

Note

If Security Services single sign-on (SSO) is enabled for your server, the user and password arguments are authenticated externally through SSO rather than being authenticated against the Teamcenter database. If you do not supply these arguments, the utility attempts to join an existing SSO session. If no session is found, you are prompted to enter a user ID and password.

-p

Specifies the user's password.

If used without a value, the system assumes a null value. If this argument is not used, the system assumes the *user-ID* value to be the password.

This argument is mutually exclusive with the **-pf** argument.

-pf

Specifies the password file. If used without a value, the system assumes a null value. If this argument is not used, the system assumes the *user-ID* value to be the password.

For more information about managing password files, see [Manage password files](#).

This argument is mutually exclusive with the **-p** argument.

-g

Specifies the group associated with the user.

If used without a value, the user's default group is assumed.

-mode=delete

Removes the specified non-foundation preference definitions from the system. You can supply the preference names in a file or on the command line. If you do not supply either, all non-foundation preferences are deleted.

Note This mode requires system administration privileges.

-preferences

Specifies a command separated list of preference names to delete from the database. Cannot contain foundation preference names. This argument cannot be used when the **-exception_file** argument is specified.

-exception_file

Specifies the full path and name of a file that contains names of preferences that are excluded from the delete operation. This argument cannot be used when the **-preferences** argument is specified. Each preference name to be excluded must be on a separate line in the file.

-file

Contains a list of preferences to be deleted. Each preference name must be on a separate line in the file.

Note This option cannot be used with the **-preferences** option.

-dry_run

Provides a report of the delete operation but does not actually delete the preferences.

-report_file

Full path and file name of the file that contains the log information for the utility. If you do not specify this argument, the utility creates a file using a default path and file name.

-h

Displays help for this utility.

-mode=remove

Removes the specified preference instances from the specified location in the database.

SYNTAX

```
preferences_manager
-u=user-name
{-p=password | -pf=password-file}
-g=group-name
-mode=remove
[-scope= {SITE | GROUP | ROLE | USER}]
[-target= {user-ID | role-ID | group-ID}] |
[-u_target= list-of-user-IDs] [-r_target= list-of-role-IDs]
[-g_target=list-of-group-IDs]
[-preferences=preference-name]
[-file=file-name]
[-h]
```

ARGUMENTS**-u**

Specifies the user ID.

This is generally **infodba** or another user with administration privileges. If this argument is used without a value, the operating system user name is used.

Note

If Security Services single sign-on (SSO) is enabled for your server, the user and password arguments are authenticated externally through SSO rather than being authenticated against the Teamcenter database. If you do not supply these arguments, the utility attempts to join an existing SSO session. If no session is found, you are prompted to enter a user ID and password.

-p

Specifies the user's password.

If used without a value, the system assumes a null value. If this argument is not used, the system assumes the *user-ID* value to be the password.

This argument is mutually exclusive with the **-pf** argument.

-pf

Specifies the password file. If used without a value, the system assumes a null value. If this argument is not used, the system assumes the *user-ID* value to be the password.

For more information about managing password files, see [Manage password files](#).

This argument is mutually exclusive with the **-p** argument.

-g

Specifies the group associated with the user.

If used without a value, the user's default group is assumed.

-mode=remove

Removes the specified preference instances from the specified location in the database.

Note This mode requires system administration privileges.

-scope

Specifies the location under which the preference instances are deleted. This option may be used with the **-target** option, but never with any of the **-u_target**, **-r_target** or **-g_target** options. This option can accept one of the following keywords:

- **SITE**

Removes the specified preferences if they exist in the current logged-on group preferences list. This means that the logged-on user must have group administrator privileges.

- **GROUP**

Removes the specified preferences if they exist in the current logged-on group preferences list. This means that the logged-on user must have group administrator privileges.

- **ROLE**

Removes the specified preferences if they exist in the current logged-on role preferences list. This means that the logged-on user must have group administrator privileges.

- **USER**

Removes the specified preferences if they exist in the current logged-on user preferences list.

Note If not specified, this is the default value.

-target

Specifies the user or role or group ID whose preferences are to be removed. Must be a valid user, role or group ID.

If not specified, the removal applies to all preferences (including the site location).

-preferences

Specifies comma-separated preference names for which the preference instances are to be deleted from the database under the specified locations.

-file

Contains a list of preferences to be deleted. Each preference should be on a separate line in the file.

Note This option cannot be used with the **-preferences** option.

-h

Displays help for this utility.

-mode=upgradexml

Upgrades a preference file from a specified format to the current format. The [upgrade_preferences_file.pl](#) perl script can also be used to perform these actions.

For more information about the administering preferences, see the [Application Administration Guide](#)

SYNTAX**preferences_manager**

```
-u=user-name
{-p=password | -pf=password-file}
-g=group-name
[-mode=upgradexml]
-input_file=path-to-input-file
-definition_information=path-to-information-file
[-separator=separator-used-in-information-file ]
[-default_protection_scope= {SITE | GROUP | ROLE | USER} ]
[-default_env_variable_status= {true | false} ]
[-correct_errors]
-output_file=path-to-output-file
-report_file=path-to-report-file
[-h]
```

ARGUMENTS**-u**

Specifies the user ID.

This is generally **infodba** or another user with administration privileges. If this argument is used without a value, the operating system user name is used.

Note

If Security Services single sign-on (SSO) is enabled for your server, the user and password arguments are authenticated externally through SSO rather than being authenticated against the Teamcenter database. If you do not supply these arguments, the utility attempts to join an existing SSO session. If no session is found, you are prompted to enter a user ID and password.

-p

Specifies the user's password.

If used without a value, the system assumes a null value. If this argument is not used, the system assumes the *user-ID* value to be the password.

This argument is mutually exclusive with the **-pf** argument.

-pf

Specifies the password file. If used without a value, the system assumes a null value. If this argument is not used, the system assumes the *user-ID* value to be the password.

For more information about managing password files, see [Manage password files](#).

This argument is mutually exclusive with the **-p** argument.

-g

Specifies the group associated with the user.

If used without a value, the user's default group is assumed.

-mode=upgradexml

Upgrades a preference file from a specified format to the current format.

Note This mode requires system administration privileges.

-input_file

Specifies the full path to the preferences file to be updated. If the file is in the format of a previous Teamcenter release, the output file is in the current format. If the file is in the current format, the output file is in the current format and the information updated using the file specified by the **definition_information** argument. This is useful to update the preference file from an external text file.

-definition_information

Specifies the full path to the text file containing information about updated preferences using the file passed in the desired protection scope and the desired environment variable status. If the file is in the format of a previous Teamcenter release, the output file is in the current format.

The content of the text file is in the format:

preference-name;protection-scope;environment-variable-status

The content of the definition information text file must be in the format:

preference-name;protection-scope;environment-variable-capability

The *protection-scope* and *environment-variable-capability* settings are optional. If these values are not specified, the values of the **-default_protection_scope** and **-default_env_variable_status** arguments are used.

Valid values for the *protection-scope* setting are **User**, **Role**, **Group**, **Site**, or **System**.

Valid values for the *environment-variable-capability* setting are **true** (indicating that the preference can also be set using an environment variable) or **false**.

For example:

prefA;User>false

prefB;Role;

prefC;Site

prefD>false

Tip

You can create a one-to-one correspondence between each preference listed in the input file and the specified default settings in the definition file. Alternatively, omitting default settings in the definition information file for any given preference listed in the input file applies the values of the **-default_protection_scope** and **-default_env_variable_status** arguments to the preference.

If the **-default_protection_scope** argument is not specified, the system assigns that preference a default protection scope of **User**.

If the **-default_env_variable_status** argument is not specified, the system assigns that preference the environment variable capability of **false**.

-separator

Specifies the string used between each field in the file specified in the **definition_information** parameter. If not given, the character **;** is assumed to be the separator.

-default_protection_scope

Specifies the default protection scopes to preferences not listed in the file specified in the **definition_information** parameter, or when the supplied information is incorrect or absent. Valid values are:

- **SITE**

Upgrades the specified values to preferences at the site location. Only users logged on as system administrators can upgrade these preferences.

- **GROUP**

Upgrades the specified values to preferences at the site location. Only users logged on as group administrators can upgrade these preferences.

- **ROLE**

Upgrades the specified values to preferences at the site location. Only users logged on as role administrators can upgrade these preferences.

- **USER**

Upgrades the specified values to preferences at the site location. Only users logged on as system administrators can modify these preferences.

Note

If not specified, this is the default value.

- **SYSTEM**

Upgrades the specified values to preferences at the system location. Only users logged on as system administrators can upgrade these preferences.

-default-env-variable-status

Specifies the default value for the **envEnabled** attribute to apply to the preferences either not listed in the file specified in the **definition_information** parameter, or when the supplied information is incorrect or absent. Valid values are:

- **false**

Note If not specified, this is the default value.

- **true**

-correct_errors

Corrects errors encountered whenever possible. Types of errors that can be corrected include incorrect types, array status, protection scope, and category errors. Errors that must be manually corrected are indicated in the report file.

-output_file

Specifies the full path to the output file containing the updated preferences. If the file already exists, it is overwritten.

-report_file

Full path to the file containing logging information about the upgrade task. If the argument is not provided, the system creates a default file.

-h

Displays help for this utility.

Data access management

am_install_tree

Installs an Access Manager (AM) rule tree at your site. Teamcenter supplies a default rule tree that provides a starting point for creating rules at your site. If you do not specify the **[operation]** option, it imports a rule tree. In that case, the **[mode]** option is required.

SYNTAX

```
am_install_tree -u=user-id {-p=password | -pf=password-file} [-g=group]
[-operation={import | export} ] -path=file-name [-mode=replace_all |
replace_tree | no_replace}] -format={xml | txt} [-h]
```

ARGUMENTS

-u

Specifies the user ID.

This is generally **infodba** or another user with administration privileges. If this argument is used without a value, the operating system user name is used.

Note

If Security Services single sign-on (SSO) is enabled for your server, the **-u** and **-p** arguments are authenticated externally through SSO rather than being authenticated against the Teamcenter database. If you do not supply these arguments, the utility attempts to join an existing SSO session. If no session is found, you are prompted to enter a user ID and password.

-p

Specifies the password.

If used without a value, the system assumes a null value. This argument is mutually exclusive with the **-pf** argument.

If this argument is not used, the system assumes the *user-ID* value to be the password.

-pf

Specifies the password file. If used without a value, the system assumes a null value. If this argument is not used, the system assumes the *user-ID* value to be the password.

For more information about managing password files, see [Manage password files](#).

This argument is mutually exclusive with the **-p** argument.

-g

Specifies the group associated with the user.

If used without a value, the user's default group is assumed.

-operation

Specifies whether to import or export the rule tree file.

If you do not specify the **[operation]** option, **am_install_tree** assumes the default of importing a rule tree. In that case, the **[-mode]** option is required.

=import

Default mode. Imports a rule tree.

=export Exports a rule tree.

If you specify **[-operation=export]**, the option **[-mode]** is not required.

-path

Specifies the full path of the rule tree file.

-mode

Specifies one of the following installation modes:

=replace_all

Overwrites both the existing rule tree and any named access control lists (ACLs) in the system.

=replace_tree

Overwrites the existing rule tree but does not overwrite existing named ACLs in the system.

=no_replace

Default mode. Used only when installing the first rule tree at your site. Does nothing if an existing rule tree is detected.

-format

Specifies one of the following file formats:

=xml

XML file format.

=txt

Text file format.

-h

Displays help for this utility.

ENVIRONMENT

As specified in [Manually configuring your environment for Teamcenter utilities](#).

FILES

As specified in [Log files produced by Teamcenter](#).

RESTRICTIONS

None.

RETURN VALUES

Return value 0
upon success

Return value 1
upon failure

EXAMPLE

The following example exports the rule tree **export_rule.txt**:

```
am_install_tree -u=tcadmin -p=tcadmin -g=admin -operation=export  
-path=D:\export_rule.txt -format=txt
```

ada_util

Provides an alternate procedure to the user interface-based setting of classification/clearance and license information so it can be called from scripts. This utility supports the following authorized data access (ADA) functions:

- **set_classification**
- **set_clearance**
- **newlicense**
- **addlicense**
- **modlicense**
- **adduser**
- **adduser**

For more information about ADA licenses, see the [Authorized Data Access License Guide](#).

SYNTAX

```
ada_util [-u=user-id -p=password | -pf=filename g=group]
-setclassification
  -c=classification
  -item=item [-rev=revision [-ot=object-type -on=object-name]]
  -type=[itar | ip]
-setclearance
  -c=clearance-level
  -uid=user-id
  -type=[itar | ip]
-newlicense
  -l=license-id
  [-d=date]
  [-reason=reason]
  [-type=[itar | ip | exclude]
  [-lock_date=date]
  [-qualifying_cfr=string]
  [-category=string]
  [-citizenships=citizenship-list]
-addlicense
  -l=license-id
  [-ead_paragraph=string]
  -item=item [-rev=revision [-ot=object-type -on=object-name]]
-modlicense
  -l=license-id
  -d=date
  [-lock_date=date]
  [-qualifying_cfr=string]
  [-category=string]
  [-citizenships=citizenship-list]
-adduser
  -l=license-id
  -uid=user
-addgroup
  -l=license-id
  -gid=group
```

[-h]

ARGUMENTS

-u

Specifies the user ID.

This is generally **infodba** or another user with **IP Admin** or **ITAR Admin** privilege according to the classification/clearance or license type involved in the operation. If this argument is used without a value, the operating system user name is used.

Note

If Security Services single sign-on (SSO) is enabled for your server, the **-u** and **-p** arguments are authenticated externally through SSO rather than being authenticated against the Teamcenter database. If you do not supply these arguments, the utility attempts to join an existing SSO session. If no session is found, you are prompted to enter a user ID and password.

-p

Specifies the password.

If used without a value, the system assumes a null value. This argument is mutually exclusive with the **-pf** argument.

If this argument is not used, the system assumes the *user-ID* value to be the password.

-pf

Specifies the password file. If used without a value, the system assumes a null value. If this argument is not used, the system assumes the *user-ID* value to be the password.

For more information about managing password files, see [Manage password files](#).

This argument is mutually exclusive with the **-p** argument.

-g

Specifies the group associated with the user.

If used without a value, the user's default group is assumed.

-setclassification

Sets a classification for a given workspace object as specified by the **-c**, **-item**, **-rev**, **-ot**, **-on**, and **-type** arguments. This argument cannot be used with the **-setclearance** argument.

-setclearance

Sets a clearance level for a given user specified by the **-c**, **-uid**, and **-type** arguments. This argument cannot be used with the **-setclassification** argument.

-newlicense

Creates a license with given ID using the **-l** and **-type** argument with the optional **-d** expiration date, the optional **-reason** reason, the optional **-lock_date** lock date, and the optional **-qualifying_cfr** string. The **-qualifying_cfr** argument is applicable only to ITAR licenses. The **-newlicense** argument cannot be used with the **-addlicense** and **-modlicense** arguments.

-addlicense

Adds the license identified using the **-l** argument to the object specified by **-item**, **-rev**, **-ot**, and **-on** arguments. This argument cannot be used with the **-newlicense** and **-modlicense** arguments.

Additionally, qualifying paragraph information can be supplied using the **-ead_paragraph** argument.

-modlicense

Modifies a license identified using the **-l** argument to have a new expiration date specified using the **-d** argument, a new lock date specified using the **-lock_date** argument, and new qualifying Code of Federal Regulation (CFR) information specified for licenses of ITAR type using the **-qualifying_cfr** argument. This argument cannot be used with the **-newlicense** and **-addlicense** arguments.

-adduser

Adds the user identified using the **-uid** argument to the license identified by the **-l** argument.

-addgroup

Adds the group identified using the **-gid** argument to the license identified by the **-l** argument.

-c

Specifies a classification or clearance according to context. Use with the **-setclassification** argument to apply value given in this argument to a given object; use with the **-setclearance** argument to apply the value to the given user.

-item

Specifies the item to use in finding an object. See the description for the **-on** argument.

-rev

Specifies the revision within a given item to use in finding an object. See the description for the **-on** argument.

-ot

Specifies the object type to filter specification attachments of the given item/revision when finding an object. See the description for the **-on** argument.

-on

Specifies an object to which the various ADA operations apply. The value should be the object name, for example, a dataset name. This argument is required.

This argument also requires the **-item**, **-rev**, and **-ot** arguments to provide a basis for finding the object; first, uniquely identifying the item (**-item**), second, identifying the specific revision of that item (**-rev**), and third, filtering objects attached to the item revision based on object type and name (**-ot**).

- If you specify only the **-item** argument, the object is the item.
- If you specify the **-item** and **-rev** arguments, the object is the revision.
- If you specify the **-item**, **-rev**, and **-ot** arguments, the object is that named with the type on the item revision.

See **-setclassification** and **-addlicense**.

-type

Specifies the license type as **itar** (International Traffic in Arms Regulations), **ip** (intellectual property), or **exclude** (exclude license) to exclude certain users who are not allowed to see the data.

If **-type** is set to **itar**, the **-setclassification** argument applies to **gov_classification** and the **-setclearance** argument applies to **gov_clearance**.

If **-type** is set to **ip**, the **-setclassification** argument applies to **ip_classification** and the **-setclearance** argument applies to **ip_clearance**.

The default value is **ip**.

-uid

Specifies a user for the **-setclearance** and **-adduser** operations. This argument should not be confused with the login user argument, **-u**.

-l

Specifies a unique license ID for use with the **-newlicense**, **-addlicense**, and **-modlicense** arguments.

-d

Specifies a date for use in **-newlicense** and **-modlicense** operations.

The default date format is defined in **timelocal.xml** as *numericday-abbreviatedmonth-numericyear hours:minutes*. *hours* are in 24-hour format. For example, *11-dec-2006 15:20*. The month names and default date format may change with locale.

-reason

Specifies a string when using the **-newlicense** argument.

-lock_date

Enables authorized users to freeze or unfreeze the license specified by the **-l** argument on the specified date. This argument can only be used with the **-newlicense** or **-modlicense** arguments.

-qualifying_cfr

Specifies the information for the **In Accordance With** attribute (a string of 80 characters) for the **ITAR_License** object identified by the **-l** argument. This argument can only be used in with the **-newlicense** or **-modlicense** arguments.

-category

Specifies the category type (a string of 128 bytes) for an ADA license. This argument can only be used with the **-newlicense** or **-modlicense** arguments.

-citizenships

Specifies the user citizenships for an ADA license. Each citizenship is a two-letter country code from ISO 3166. Multiple citizenships are separated by a comma delimiter, such as:

```
-citizenships=US,GB
```

-ead_paragraph

Specifies the authorizing paragraph information (a string of 80 characters) recorded on the workspace object specified by **-item**, **-rev**, **-ot**, and **-on** arguments, while attaching an **ITAR_License** object identified by the **-l** argument.

-h

Displays help for this utility.

ENVIRONMENT

As specified in [Manually configuring your environment for Teamcenter utilities](#).

FILES

As specified in [Log files produced by Teamcenter](#).

RESTRICTIONS

None.

EXAMPLES

- To set a classification of **secret** on the **1234/A** UG master dataset attached to the item **1234** revision **A**:

```
ada_util -u=user -p=pass -g=group -setclassification -c=secret
        -item=1234 -rev=A -ot="UG Master" -on=1234/A -type=ip
```

- To set an IP clearance level of **secret** for the **user2** user:

```
ada_util -u=user -p=pass -g=group -setclearance -c=secret
        -uid=user2 -type=ip
```

- To create a new ITAR license **license001** with an expiration date of 11 December 2006 at 15:20:

```
ada_util -u=user -p=pass -g=group -newlicense -l=license001
        -d="11-dec-2006 15:20" -type=itar
```

- To add **user2** user to the **license001** license:

```
ada_util -u=user -p=pass -g=group -adduser -l=license001 -uid=user2
```

- To apply the **license001** license to the **1234/A** dataset attached to item **1234** revision **A**:

```
ada_util -u=user -p=pass -g=group -addlicense -l=license001
        -item=1234 -rev=A -ot="UG Master" -on=1234/A
```

- To apply the **license001** license to the **1234/A** dataset attached to item **1234** revision **A** without specifying the **-rev**, **-ot**, and **-on** options:

```
ada_util -u=user -p=pass -g=group -addlicense
        -l=license001 -item=1234
```

- To create the **ITAR_license001** ITAR license with a category of Category A:

```
ada_util -u=user -p=pass -g=group -newlicense
        -l=ITAR_license001 -category="Category A"
```

- To update the **ITAR_license001** ITAR license to Category B:

```
ada_util -u=user -p=pass -g=group -modlicense
        -l=ITAR_license001 -category="Category B"
```

- To create a new **ITAR_license01** with an allowed citizenship for Great Britain:

```
ada_util -u=user -p=password -g=group -newlicense
```

```
-l=ITAR_license01 -type=itar -citizenships=GB
```

- To update user citizenships on the license **ITAR_license01** to Great Britain and Japan:

```
ada_util -u=user -p=password -g=group -modlicense  
-l=ITAR_license01 -citizenships=GB,JP
```

install_authorization_rules

Creates system-level administration authorization rules.

SYNTAX

```
install_authorization_rules [-u=user-id {-p=password |  
-pf=password-file} [-g=group]  
-function {install | create | add | listaccessors | listapplications | listutilities}  
[-name=application-or-utility-name]  
[-ruledomain=rule-domain-value]  
[-role=role-name]  
[-group=group-name]  
[-h]
```

ARGUMENTS

-u

Specifies the user ID.

This is generally **infodba** or another user with administration privileges. If this argument is used without a value, the operating system user name is used.

Note

If Security Services single sign-on (SSO) is enabled for your server, the **-u** and **-p** arguments are authenticated externally through SSO rather than being authenticated against the Teamcenter database. If you do not supply these arguments, the utility attempts to join an existing SSO session. If no session is found, you are prompted to enter a user ID and password.

-p

Specifies the password.

If used without a value, the system assumes a null value. This argument is mutually exclusive with the **-pf** argument.

If this argument is not used, the system assumes the *user-ID* value to be the password.

-pf

Specifies the password file. If used without a value, the system assumes a null value. If this argument is not used, the system assumes the *user-ID* value to be the password.

For more information about managing password files, see [Manage password files](#).

This argument is mutually exclusive with the **-p** argument.

-g

Specifies the group associated with the user.

If used without a value, the user's default group is assumed.

function

Specifies one of the following functions:

install

Installs standard authorization rules for administration applications and utilities.

create

Creates new authorization rules. If you specify this function, you must include values for the **name** and **ruledomain** arguments.

add

Adds a new accessor to an existing authorization rule. If you specify this function, you must include values for the **name**, **ruledomain**, and **group** arguments.

listaccessors

Lists accessors specified by the **name** and **ruledomain** arguments.

listapplications

Lists all application names for which rules are defined in the database.

listutilities

Lists all utility names for which rules are defined in the database.

-name

Specifies the application or utility name. Use this argument with the **create**, **add**, or **listaccessors** functions.

-ruledomain

Specifies the value of the rule domain. Use this argument with the **create**, **add**, or **listaccessors** functions.

-role

Specifies the role name.

-group

Specifies the group name. Use this argument with the **add** function.

-h

Displays help for this utility.

ENVIRONMENT

As specified in [Manually configuring your environment for Teamcenter utilities](#).

FILES

As specified in [Log files produced by Teamcenter](#).

RESTRICTIONS

None.

EXAMPLES

- Install the default authorization rules for applications and utilities:

```
install_authorization_rules -u=infodba -p=infodba
-g=dba -install
```
- Create authorization rule for the **APP_1** application:

```
install_authorization_rules -u=infodba -p=infodba
-g=dba -create -name=APP_1 -ruledomain=application
```
- Add the **GRP_1** group as a valid accessor for the **APP_1** application:

```
install_authorization_rules -u=infodba -p=infodba
-g=dba -add -name=APP_1 -ruledomain=application -group=GRP_1
```
- List all the accessors for the **APP_1** application:


```
install_authorization_rules -u=infodba -p=infodba  
-g=dba -listaccessors -name=APP_1 -ruledomain=application
```

- **List all application names for which rules are defined in the database:**

```
install_authorization_rules -u=infodba -p=infodba  
-listapplications
```

- **List all utility names for which rules are defined in the database:**

```
install_authorization_rules -u=infodba -p=infodba  
-listutilities
```

install_callback

Registers the callbacks and persists them in the database.

SYNTAX

```
utility_name -u=user-ID {-p=password | -pf=password-file} [-g=group]  
[-mode=install | create | modify | delete | list]  
-type=  
-library=  
-function=  
-name=  
[-h]
```

ARGUMENTS**-u**

Specifies the user ID. The user must have administrative privileges.

If this argument is used without a value, the operating system user name is used.

Note

If your Teamcenter server uses Security Services single sign-on, see [Before you begin](#) for additional information.

-p

Specifies the user's password.

If used without a value, the system assumes a null value. If this argument is not used, the system assumes the *user-ID* value to be the password.

This argument is mutually exclusive with the **-pf** argument.

-pf

Specifies the password file. If used without a value, the system assumes a null value. If this argument is not used, the system assumes the *user-ID* value to be the password.

For more information about managing password files, see [Manage password files](#).

This argument is mutually exclusive with the **-p** argument.

-g

Specifies the group associated with the user.

-mode

Specifies the callback mode.

- **install**

Specifies the install mode.

- **create**

Specifies the create mode.

If running in this mode, specify **-type**, **-library**, **-function**, or **-name**.

- **modify**

Specifies the modify mode.

If running in this mode, specify **-type**, **-library**, **-function**, or **-name**.

- **delete**

Specifies the delete mode.

If running in this mode, specify **-type** or **-name**.

- **list**

Specifies the list mode.

-type

Specifies the type for the **create**, **modify**, or **delete** mode.

-library

Specifies the library for the **create** or **modify** mode.

-function

Specifies the function for the **create** or **modify** mode.

-name

Specifies the name for the **create**, **modify**, or **delete** mode.

-h

Displays help for this utility.

install_vminfo_acl

Creates access control list (ACL) rules for migration of the existing Teamcenter volume files. The Volume Management application introduces changes to the Access Manager (AM) rule tree. This utility verifies whether the AM rule tree contains the required rules (**HSM_Info** and **VM_Info**). If not, the rules are added to inherit the access privileges of the named ACL **POM Open Access** in par with the **ImanFile** object and saves the changes.

This utility runs automatically at install. Typically, there is no need for administrators to run the utility again. In cases where an administrator has overwritten the rule tree with custom rules, this utility can be run to ensure the required rules are added to the rule tree.

SYNTAX

install_vminfo_acl [-u=*user-id* {-p=*password* | -pf=*password-file*}
[-g=*group*] [-v] [-h]

ARGUMENTS**-u**

Specifies the user ID.

This is generally **infodba** or another user with administration privileges. If this argument is used without a value, the operating system user name is used.

Note

If Security Services single sign-on (SSO) is enabled for your server, the **-u** and **-p** arguments are authenticated externally through SSO rather than being authenticated against the Teamcenter database. If you do not supply these arguments, the utility attempts to join an existing SSO session. If no session is found, you are prompted to enter a user ID and password.

-p

Specifies the password.

If used without a value, the system assumes a null value. This argument is mutually exclusive with the **-pf** argument.

If this argument is not used, the system assumes the *user-ID* value to be the password.

-pf

Specifies the password file. If used without a value, the system assumes a null value. If this argument is not used, the system assumes the *user-ID* value to be the password.

For more information about managing password files, see [Manage password files](#).

This argument is mutually exclusive with the **-p** argument.

-g

Specifies the group associated with the user.

If used without a value, the user's default group is assumed.

-v

Verbose mode. Provides information about results and progress.

-h

Displays help for this utility.

ENVIRONMENT

- The generic command window set with all Teamcenter-related environments.
- As specified in [Manually configuring your environment for Teamcenter utilities](#).

FILES

As specified in [Log files produced by Teamcenter](#).

RESTRICTIONS

This utility is intended only for system-level users.

**RETURN
VALUES**

Return value 0
upon success

Return value 1
upon failure

EXAMPLES

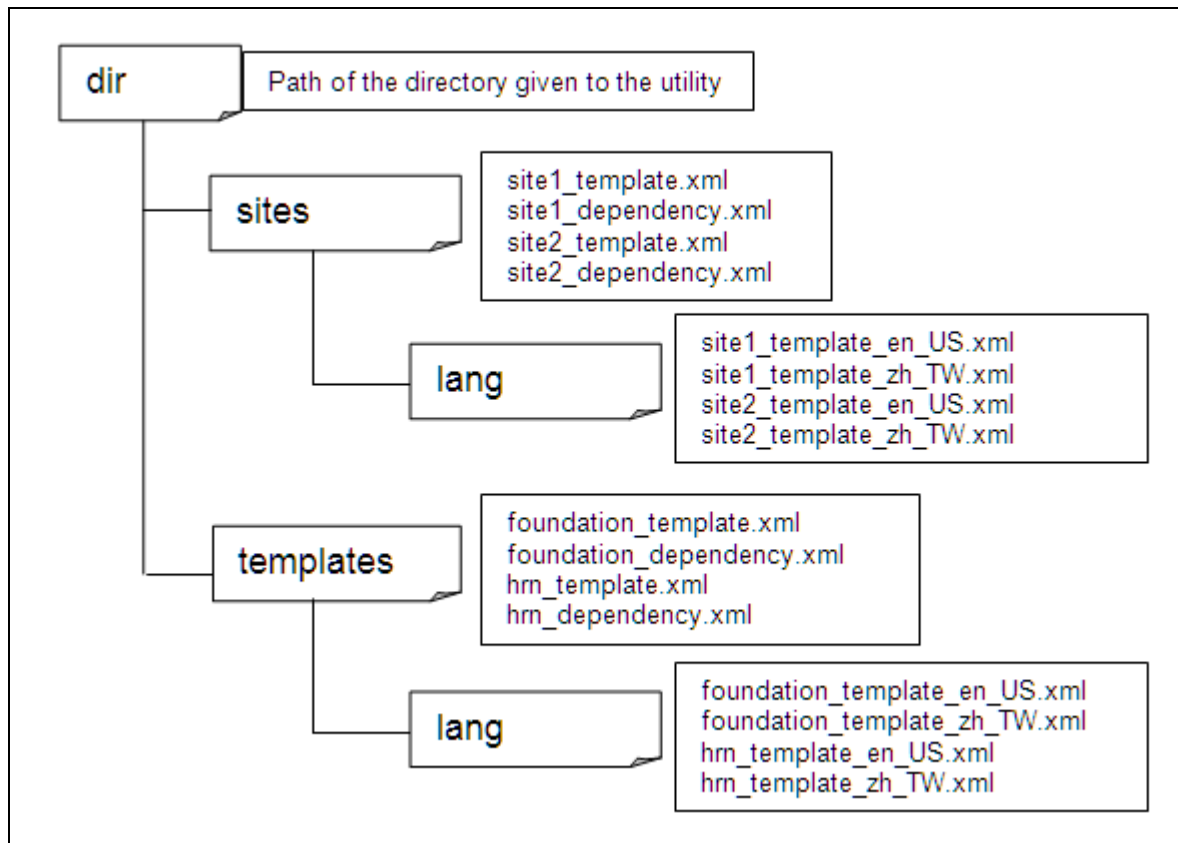
```
install_vminfo -u=infodba -p=infodba -g=dba -v
```

Business Modeler IDE

bmide_commontemplategenerator

Generates a common template between two or more sites and generates site-specific templates for each site. Use this utility when you have multiple sites containing custom data model. Previously, you had to manually analyze the site templates generated at each site to create a common template. This utility eliminates the manual work to create a template that contains custom data model common to all sites.

Place template files in the following directory structure:



Directory structure required by the **bmide_commontemplategenerator utility**

Obtain the files to place into the directories from the packaged *template-name_template.zip* files. The **sites** directory contains the templates from the different sites, the **templates** directory contains the standard templates that the site templates are dependent upon, and the **lang** directories contain the localization files used by the templates.

SYNTAX**bmide_commontemplategenerator**

-dir=site-templates-directory
-name=common-template-name
-displayname=common-template-display-name
-outputdir=output-directory
[-h]

ARGUMENTS**-dir**

Specifies the path of the directory where sites and templates folders are located. The directory should contain a **sites** subdirectory that contains all the site templates and dependency files and a **templates** subdirectory that contains all the dependant templates. The **sites** and **templates** directories should contain a **lang** subdirectory containing locale files.

-name

Specifies the name of the common template to be generated.

-displayname

Specifies the display name of the common template to be generated.

-outputdir

Specifies the path of the directory where the generated files are to be placed.

-h

Displays help for this utility.

ENVIRONMENT

As specified in the [Manually configuring your environment for Teamcenter utilities](#).

FILES

As specified in the [Log files produced by Teamcenter](#).

RESTRICTIONS

This utility has a requirement that a minimum of 1500 MB memory must be allocated to the Java heap space. Before running the utility, set the following environment variable:

```
set BMIDE_SCRIPT_ARGS=-Xmx1500M
```

For more information about the **BMIDE_SCRIPT_ARGS** environment variable, see the [Preferences and Environment Variables Reference](#).

EXAMPLES

```
bmidc_commontemplategenerator -dir=c:\templates\temp
-name=commontemplate -displayname="Common Template"
-outputdir=c:\templates\temp\output
```

bmide_comparator

Compares two complete Teamcenter model files and generates a differences file. This utility must be run with either the **-schema** or **-all** argument.

SYNTAX

**bmide_comparator -compare={schema | all} -old=old-model-file-path
-new=new-model-file-path -delta=differences-file-path
[-log=log-file-path] [-h]**

ARGUMENTS

-compare={schema | all}

Compare data model. You must specify one of these options:

- **schema**
Compares only classes.
- **all**
Compares all elements.

-old

Specifies the file path and name of the file containing the old Teamcenter model.

-new

Specifies the file path and name of the file containing the new Teamcenter model.

-delta

Specifies the file path and name of the file into which data model differences will be written.

-log

Specifies file path and name of the log file that contains the results of this execution. This argument is optional.

-h

Displays help for this utility.

ENVIRONMENT

As specified in the [Manually configuring your environment for Teamcenter utilities](#).

FILES

As specified in the [Log files produced by Teamcenter](#).

RESTRICTIONS

None.

EXAMPLES

None.

bmide_consolidator

Consolidates all templates listed in the master file into a single file.

SYNTAX

bmide_consolidator **-dir**=*master-file-directory* **-file**=*path-for-consolidated-file*
-consolidate=[**all** | **template** | **locale**]
-forceconsolidate
[**-version**=*version*]
[**-h**]

ARGUMENTS

-consolidate

all

Consolidates the template and its localization files.

template

Consolidates templates.

locale

Consolidates template localization files.

-dir

Specifies the directory path and name of the directory containing the **master.xml** file and the list of template files to be consolidated.

-file

Specifies the file path and name of the file which will contain the consolidated Teamcenter model.

-forceconsolidate

Indicates that localization files must be consolidated irrespective of what is included in the **master.xml** file.

-version

Specifies version for the consolidated file. This argument is optional.

-h

Displays help for this utility.

ENVIRONMENT

As specified in [Manually configuring your environment for Teamcenter utilities](#).

RESTRICTIONS

None.

EXAMPLES

None.

bmide_deployment_lock

Prevents simultaneous deployments to a database from any source, including Teamcenter Environment Manager (TEM) and the Business Modeler IDE. Only administrative users are allowed to run this utility.

SYNTAX

bmide_deployment_lock **-u**=*user-id* {**-p**=*password* | **-pf**=*password-file*} [**-g**=*group*]
-lock | **-release** | **-query**
[**-log**=*path*]
[**-h**]

ARGUMENTS**-u**

Specifies the user ID.

This is generally **infodba** or another user with administration privileges. If this argument is used without a value, the operating system user name is used.

Note If Security Services single sign-on (SSO) is enabled for your server, the user and password arguments are authenticated externally through SSO rather than being authenticated against the Teamcenter database. If you do not supply these arguments, the utility attempts to join an existing SSO session. If no session is found, you are prompted to enter a user ID and password.

-p

Specifies the user's password.

If used without a value, the system assumes a null value. If this argument is not used, the system assumes the *user-ID* value to be the password.

This argument is mutually exclusive with the **-pf** argument.

-pf

Specifies the password file. The file must be a single-line ASCII file containing the password in clear text. Teamcenter Environment Manager prompts you for a password and creates the password file during installation.

If used without a value, the system assumes a null value. If this argument is not used, the system assumes the *user-ID* value to be the password.

This argument is mutually exclusive with the **-p** argument.

-g

Specifies the group associated with the user.

If used without a value, the user's default group is assumed.

-lock

Sets the deployment lock. If one already exists, the utility returns an error indicating a deployment is in process. If one does not exist, the utility sets the deployment lock and returns success.

-release

Releases the deployment lock. If one already exists, the utility removes the deployment lock and returns success. If one does not exist, the utility returns success.

-query

Queries for the deployment lock. If one already exists, a **-1** is returned. If one does not exist, the utility returns success.

-log

Specifies the full path to the log file containing utility execution results.

-h

Displays help for this utility.

ENVIRONMENT

As specified in *[Manually configuring your environment for Teamcenter utilities](#)*.

FILES

As specified in *[Log files produced by Teamcenter](#)*.

RESTRICTIONS

None.

bmide_generatecode

Autogenerates C/C++ code for business objects and property operations from the Business Modeler IDE.

SYNTAX

bmide_generatecode
-templateProj=*source-template-project-input-location*
-templateDeps=*dependent-templates-input-location*
-srcDir=*skeleton-implementation-classes-output-location*
-gensrcDir=*autogenerated-classes-output-location*
-makefileDir=*root-makefile-output-location*
-serviceLibs=**[all | service-libraries]**
-log=*log-output-file*
[-h]

ARGUMENTS**-templateProj**

Specifies the file path and name of the input location of the source template project.

-templateDeps

Specifies the file path and name of the input location of the dependent templates.

-srcDir

Specifies the file path and name of the output location of the skeleton implementation classes. This argument is optional.

-gensrcDir

Specifies the file path and name of the output location for autogenerated classes.

-makefileDir

Specifies the root file path of the output location for generated makefiles.

-serviceLibs

Generates code for all custom service libraries defined in the specified Business Modeler IDE project. Specify a comma-delimited list of service library names or **all**, for example, **-serviceLibs=all**.

Before running this utility with the **-serviceLibs** argument, you must do the following:

1. Set the **JDK_HOME** environment variable to point to the installed JDK location.
2. Add the *JDK_HOME\bin* directory to the path variable.
3. In the Business Modeler IDE project, ensure that the **SoaExternalBuild.SoaClientKitLocation** value in the **ProjectInfo.xml** file is set to the location of the *soa_client* folder. This corresponds to the value in the **Teamcenter Services client kit home** box in the **Build Configuration** dialog box for the project. To access this dialog box, in the **Navigator** view, right-click the project and choose **Properties**, and in the left pane, choose **Teamcenter® Build configuration**.

-log

Specifies file path and name of the log file that contains the results of this execution. This argument is optional.

-h

Displays help for this utility.

ENVIRONMENT

As specified in [Manually configuring your environment for Teamcenter utilities](#).

FILES

As specified in [Log files produced by Teamcenter](#).

RESTRICTIONS

None.

EXAMPLES

```
bmide_generatecode
-templateProj=D:\udu\meta_dev10\custom1
-templateDeps=D:\udu\meta_dev10\templates
-srcDir=D D:\udu\meta_dev10\custom1\src\server
-gensrcDir=D:\udu\meta_dev10\custom1\output\server\gensrc
-makefileDir=D:\udu\meta_dev10\wnti32\drv\core
-log=D:\udu\meta_dev10\CodeGenUtil.log
```

bmide_generate_compare_report

Reports the differences between two sets of data models. For example, you can compare two Teamcenter database sites to determine if the data model is the same in both sites. The report generated by this utility shows the differences. You must provide two consolidated data model sources as input to generate the report. (You cannot provide other source for input.)

You can also generate this report using the Business Modeler IDE.

For more information, see the [Business Modeler IDE Guide](#).

SYNTAX

bmide_generate_compare_report
-file=consolidated-model-file
-comparefile=consolidated-model-file-to-compare
-reportfile=generated-report-file
-showequal=[true | false]
-log=log-file
[-h]

ARGUMENTS**-file**

Specifies the initial consolidated model file used to generate the report. A consolidated model file can be obtained by running the [business_model_extractor](#) utility to extract all the elements or by getting the **model.xml** file from the **TC_DATA\model** directory of a Teamcenter database site.

-comparefile

Specifies the consolidated model file to be compared to the initial file specified in the **-file** argument.

-reportfile

Specifies the file where the report is generated.

-showequal

Defines whether equal values from both data models are shown in the generated report. The valid values are **true** or **false**. The default value is **false**. If the value is **true**, the equal attributes are shown in the generated report.

-log

Specifies the path of the log file containing results of the execution.

-h

Displays help for this utility.

ENVIRONMENT

As specified in [Manually configuring your environment for Teamcenter utilities](#).

FILES

As specified in [Log files produced by Teamcenter](#).

RESTRICTIONS

None.

bmide_generate_condition_report

Reports the details of a condition and all model elements that use it in a set of data model. Conditions are attached to various business rules, LOVS, and so on to define the behavior of Teamcenter. The generated report is a single HTML page showing the model elements that refer to the condition. To generate this report, you must provide a condition name and a consolidated model file for the data model source.

You can also generate this report using the Business Modeler IDE.

For more information, see the [Business Modeler IDE Guide](#).

SYNTAX

bmide_generate_condition_report

-file=consolidated-model-file
-condition=condition-name
-reportfile=generated-report-file
-log=log-file
[-h]

ARGUMENTS

-file

Specifies the consolidated model file used to generate the report. A consolidated model file can be obtained by running the [business_model_extractor](#) utility to extract all the elements or by getting the **model.xml** file from the *TC_DATA\model* directory of a Teamcenter database site.

-condition

Specifies the condition name for which the report is to be generated.

-reportfile

Specifies the file where the report is generated.

-log

Specifies the path of the log file containing results of the execution.

-h

Displays help for this utility.

ENVIRONMENT

As specified in [Manually configuring your environment for Teamcenter utilities](#).

FILES

As specified in [Log files produced by Teamcenter](#).

RESTRICTIONS

None.

bmide_generate_datamodel_doc_report

Generates an HTML data model report of business objects, business rules, and LOVs of the current Teamcenter version. The data model for the report is built using the Business Modeler IDE templates found in the target templates directory specified by the user (for example, *TC_ROOT/bmide/templates*).

You can also generate this report using the Business Modeler IDE.

For more information, see the [Business Modeler IDE Guide](#).

The data model report consists of the following sections:

- **Overview**
Provides an overview of the data model report.
- **Template Data Model**
Provides a list of the template names that contribute to the report.
- **What's New**
Provides a list of what has changed in the data model since a previous version.
- **Glossary**
Provides a glossary of all the data model elements managed by the Business Modeler IDE.
- **Deprecated**
Provides a list of all libraries and operations that are deprecated.

SYNTAX

```
bmide_generate_datamodel_doc_report  
-targetTemplatesDir=target-file-path-name  
[-sourceReleaseVersions=[Tc200715 | Tc801 | Tc810 | Tc820 | Tc830 | all]  
-outputDir=directory-for-generated-reports  
[-skip=XML-file-excluded-templates]  
[-h]
```

ARGUMENTS**-targetTemplatesDir**

Specifies the full path to the directory containing all templates from the target version for which the data model report must be generated, such as *TC_DATA/model* or *TC_ROOT/bmide/templates*.

-sourceReleaseVersions

Specifies a comma-separated list of Teamcenter versions to generate a comparison, for example, **-sourceReleaseVersions=Tc820,Tc830**.

To generate a report for all versions, specify **-sourceReleaseVersions=all**.

-outputDir

Specifies the full path to the directory where the reports must be generated.

-skip

Specifies a list of templates in the target templates directory to skip during processing. This argument should specify the full path to the XML file that has the list of templates. Following is an example of the XML file format:

```
<?xml version="1.0" encoding="UTF-8" standalone="no">
<TcBusinessdataIncludes>
  <exclude file="hrn_template.xml"/>
  <exclude file="erp_template.xml"/>
</TcBusinessdataIncludes>
```

-h

Displays help for this utility.

RESTRICTIONS

Following are the limitations of this utility:

- This report only generates business objects and classes that are under the **Item**, **ItemRevision**, **Form**, **Dataset**, and **Folder** business objects or classes. This report also generates classes that are the form storage classes of any item master form or item revision master form business object. All other classes and business objects are not shown as they are considered internal to the product. Therefore, if you supply your own template in the target directory, only the business objects and classes that fall into these categories are shown in the report.
- The reporting tool contains a record of all Teamcenter templates that are included with Teamcenter for each prior version. Therefore, when generating the **What's New** section, the report is able to accurately generate a comparison of all elements in a target version against a designated former source version. This results in the **What's New** comparison displaying new, changed, and removed for each element. Because the reporting tool does not include a record of any other templates, including third-party and customer templates, the report automatically lists all elements from these templates as new in the **What's New** section. Disregard the new status because it does not reflect whether the element was really new, changed, or removed.

EXAMPLES

- To generate an HTML data model report, enter a command like the following on a single line:


```
bmid_generate_datamodel_doc_report
-targetTemplatesDir=C:\Siemens\config1\tcdata\model
-sourceReleaseVersions=all -outputDir=C:\datamodelreports\tc9.0
-skip=C:\skiplist.xml
```
- To generate the **What's New** section of the report, choose one or more source versions to compare with the current version, as shown in the following examples:
 - o To generate a report that compares the data model found in Teamcenter 8.1, use the following command:


```
bmid_generate_datamodel_doc_report
-targetTemplatesDir=C:\Siemens\config1\tcdata\model
-sourceReleaseVersions=Tc810 -outputDir=C:\datamodelreports\tc9.0
```
 - o To generate a report that compares Teamcenter 8.1 and Teamcenter 8.2, use the following command:

```
bmid_generate_datamodel_doc_report
-targetTemplatesDir=C:\Siemens\config1\tcdata\model
```

```
-sourceReleaseVersions=Tc810,Tc820 -outputDir=C:\datamodelreports\tc9.0
```

- o To generate a report that compares all available versions, use the following command:

```
bmide_generate_datamodel_doc_report
-targetTemplatesDir=C:\Siemens\config1\tcdata\model
-sourceReleaseVersions=all -outputDir=C:\datamodelreports\tc9.0
```

- To skip templates during generation, provide a list of templates in a file. For example, to skip the **hrn_template** and **erp_template** templates, use the following **skiplist.xml** file:

```
<?xml version="1.0" encoding="UTF-8" standalone="no"?>
<TcBusinessDataIncludes>
  <exclude file="hrn_template.xml"/>
  <exclude file="erp_template.xml"/>
</TcBusinessDataIncludes>
```

To use the **skiplist.xml** file, run the following command:

```
bmide_generate_datamodel_doc_report
-targetTemplatesDir=C:\Siemens\config1\tcdata\model
-sourceReleaseVersions=all -outputDir=C:\datamodelreports\tc9.0
-skip=C:\skiplist.xml
```

If a template is skipped, all the templates dependent on it are also be skipped. For example, if **template2** is dependent on **template1**, and **template1** is specified in the skip list file, **template1** and **template2** are both be skipped in the generation of the report.

bmide_generate_datamodel_report

Reports the details of a given category of model elements.

Business Modeler IDE users can define a number of model elements and store them in a Business Modeler IDE template. These templates can be deployed to any Teamcenter database. At times a Business Modeler IDE user may be interested in generating a report of all LOVS or GRM rules in the system, or a report that shows a combination of multiple model element categories such as all deep copy rules and all naming rules. A user can use the Business Modeler IDE client to examine all of this information; however, this report offers an easier means to examine all elements within a given category by generating this information into a single HTML page. The data model source for the report can be specified only in the form of a consolidated model file. (You cannot provide other source for input.)

You can also generate this report using the Business Modeler IDE.

For more information, see the [Business Modeler IDE Guide](#).

Note Not specifying the **-element** argument for the data model report generates a report for all elements. Because all elements are generated into a single file, this report has limitations. As the number of categories and elements within a category grows, so does the resulting file size. In cases where you want to generate a report of multiple categories, try using the [bmide_generate_datamodel_doc_report](#) utility.

SYNTAX

```
bmide_generate_datamodel_report  
-file=consolidated-model-file  
-element=element-name  
-reportfile=generated-report-file  
-log=log-file  
[-h]
```

ARGUMENTS

-file

Specifies the initial consolidated model file used to generate the report. A consolidated model file can be obtained by running the [business_model_extractor](#) utility to extract all the elements or by getting the **model.xml** file from the **TC_DATA\model** directory of a Teamcenter database site.

-element

Specifies the comma-separated element names to include in report.

-reportfile

Specifies the file where the report is generated.

-log

Specifies the path of the log file containing results of the execution.

-h

Displays help for this utility.

ENVIRONMENT

As specified in [Manually configuring your environment for Teamcenter utilities](#).

FILES

As specified in *Log files produced by Teamcenter*.

RESTRICTIONS

This report only generates business objects and classes that are under the **Item**, **ItemRevision**, **Form**, **Dataset**, and **Folder** business objects or classes. This report also generates classes that are the form storage classes of any item master form or item revision master form business object. All other classes and business objects are not shown as they are considered internal to the product. Therefore, if you supply your own template in the target directory, only the business objects and classes that fall into these categories are shown in the report.

EXAMPLES

- To generate a report of all lists of values (LOVs), use the following command:

```
bmide_generate_datamodel_report -report=model  
-file=model-file-path -element=lov -reportfile=report-file-path
```

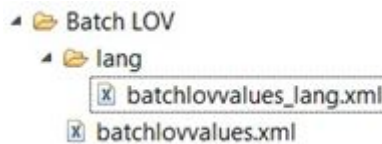
- To generate a complete list of element names that are supported as arguments, use the following command:

```
bmide_generate_datamodel_report -element -h
```

bmide_manage_batch_lovs

Manages the values for LOVs marked as externally managed, whose values are stored solely in the Teamcenter database. Use this utility to update and extract the values for externally managed LOVs and generate a report of these LOVs.

When you use this utility to update externally managed LOVs in the database, submit the LOV values in an XML file and submit the localizations in a separate XML file in a **lang** subdirectory, as shown in the following example:



If the localizations are in locale-specific files, the files must be named per their locales, as shown in the following example:



You can place either consolidated localization files or individual locale files into the **lang** directory. A consolidated localization file can contain localizations from all languages and is named *file-name_lang.xml*. Individual locale files contain localizations for one specific locale, and are named *file-name_lang_locale.xml*, for example, *file-name_en_US.xml*, *file-name_zh_CN.xml*, and so on. You can place either consolidated or locale-specific files in the **lang** folder, but not both at the same time. If you put both types in the **lang** folder, the utility does not know which to pick up and throws an error.

Note After updating the externally managed LOVs through the utility, if you are using client cache at your site, you must run the [generate_client_meta_cache](#) utility with the **generate lovs** command to update the LOV cache stored on the server.

For more information about externally managing LOVs, see the [Business Modeler IDE Guide](#).

SYNTAX

```
bmide_manage_batch_lovs -u=user-id {-p=password |
gener=password-file} [-g=group]
-option=[update | extract | report]
-lov=[all | comma-separated-list-of-lov-names]
-file=file-path
[-h]
```

ARGUMENTS

-u
Specifies the user ID.

This is generally **infodba** or another user with administration privileges. If this argument is used without a value, the operating system user name is used.

Note If Security Services single sign-on (SSO) is enabled for your server, the user and password arguments are authenticated externally through SSO rather than being authenticated against the Teamcenter database. If you do not supply these arguments, the utility attempts to join an existing SSO session. If no session is found, you are prompted to enter a user ID and password.

-p

Specifies the user's password.

If used without a value, the system assumes a null value. If this argument is not used, the system assumes the *user-ID* value to be the password.

This argument is mutually exclusive with the **-pf** argument.

-pf

Specifies the password file. If used without a value, the system assumes a null value. If this argument is not used, the system assumes the *user-ID* value to be the password.

For more information about managing password files, see [Manage password files](#).

This argument is mutually exclusive with the **-p** argument.

-g

Specifies the group associated with the user.

If used without a value, the user's default group is assumed.

-option

Valid values for option are:

- **update**
Updates the values for all the externally managed LOVs defined in the input file.
- **extract**
Extracts all the externally managed LOVs and their corresponding values into the specified file. The localizations also get extracted and are stored in the **lang** directory. Use with the **-lov** argument to specify the externally managed LOVs to be extracted.
- **report**
Provides a report of all the externally managed LOVs in the Teamcenter database. Use with the **-lov** argument to specify the LOVs in the report. The report is generated in the HTML format.

-lov

Use only in conjunction with the **-extract** or **-report** arguments. Valid values for option are:

- **all**
Extracts or runs a report on all externally managed LOVs.

- *comma-separated-list-of-lov-names*

Extracts or runs a report only on the listed externally managed LOVs.

If a list of LOV names is not specified, the **all** argument is assumed by default.

-file

Specifies the path to the file containing LOV values (when updating LOV values in the database), the path of file where the extracted LOVs are to be saved (when extracting LOV values from the database), or the file where the generated report needs to be saved (when running a report of LOV values in the database).

-h

Displays help for this utility.

ENVIRONMENT

As specified in [Manually configuring your environment for Teamcenter utilities](#).

FILES

As specified in [Log files produced by Teamcenter](#).

RESTRICTIONS

None.

EXAMPLES

When you install the Business Modeler IDE, sample files are installed to **TC_ROOT\bmide\client\samples\externallymanagedlovs**. Following are some additional examples:

- To update the LOV values and localizations, enter a command like the following on a single line:

```
bmide_manage_batch_lovs.bat -u=username -p=password -g=dba
-option=update -file=BatchLOV_LOV1.xml
```

Following is the **BatchLOV_LOV1.xml** file used in the example:

```
<?xml version="1.0" encoding="UTF-8" standalone="no"?>

<TcBusinessData xmlns="http://teamcenter.com/BusinessModel/TcBusinessData"
batchXSDVersion="1.0" >
  <Change>
    <TcLOV description="Simple Batch LOV" lovType="ListOfValuesString"
name="Ab0LOV" usage="Exhaustive" isManagedExternally="true" >
      <TcLOVValue conditionName="isTrue" description="L1 Desc" value="L1"/>
      <TcLOVValue conditionName="isTrue" description="L2 Desc" value="L2"/>
      <TcLOVValue conditionName="isTrue" description="L3 Desc" value="L3"/>
      <TcLOVValue conditionName="isTrue" description="L4 Desc" value="L4"/>
      <TcLOVValue conditionName="isTrue" description="L5 Desc" value="L5"/>
      <TcLOVValue conditionName="isTrue" description="L6 Desc" value="L6"/>
      <TcLOVValue conditionName="isTrue" description="L7 Desc" value="L7"/>
      <TcLOVValue conditionName="isTrue" description="L8 Desc" value="L8"/>
      <TcLOVValue conditionName="isTrue" description="L9 Desc" value="L9"/>
      <TcLOVValue conditionName="isTrue" description="L10 Desc" value="L10"/>
    </TcLOV>
  </Change>
</TcBusinessData>
```

Following is the **BatchLOV_LOV1_lang.xml** file that supplies the localization for the LOV values in the **BatchLOV_LOV1.xml** file. When you create a localization file, give it the same name as the LOV values file with **_lang** added to the file name, and place it in a **lang** subdirectory.

```
<?xml version="1.0" encoding="UTF-8"?>
<TcBusinessDataLocalization
xmlns="http://teamcenter.com/BusinessModel/TcBusinessDataLocalization"
batchXSDVersion="1.0" >
  <Add>
    <key locale="en_US" id="LOVValue{::}Ab0LOV{::}L1" >Level 1</key>
    <key locale="en_US" id="LOVValue{::}Ab0LOV{::}L2" >Level 2</key>
    <key locale="en_US" id="LOVValue{::}Ab0LOV{::}L3" >Level 3</key>
    <key locale="en_US" id="LOVValue{::}Ab0LOV{::}L4" >Level 4</key>
```

```

<key locale="en_US" id="LOVValue{::}Ab0LOV{::}L5" >Level 5</key>
<key locale="en_US" id="LOVValue{::}Ab0LOV{::}L6" >Level 6</key>
<key locale="en_US" id="LOVValue{::}Ab0LOV{::}L7" >Level 7</key>
<key locale="en_US" id="LOVValue{::}Ab0LOV{::}L8" >Level 8</key>
<key locale="en_US" id="LOVValue{::}Ab0LOV{::}L9" >Level 9</key>
<key locale="en_US" id="LOVValue{::}Ab0LOV{::}L10" >Level 10</key>

<key locale="en_US" id="LOVValueDescription{::}Ab0LOV{::}0" >Level 1 Desc</key>
<key locale="en_US" id="LOVValueDescription{::}Ab0LOV{::}1" >Level 2 Desc</key>
<key locale="en_US" id="LOVValueDescription{::}Ab0LOV{::}2" >Level 3 Desc</key>
<key locale="en_US" id="LOVValueDescription{::}Ab0LOV{::}3" >Level 4 Desc</key>
<key locale="en_US" id="LOVValueDescription{::}Ab0LOV{::}4" >Level 5 Desc</key>
<key locale="en_US" id="LOVValueDescription{::}Ab0LOV{::}5" >Level 6 Desc</key>
<key locale="en_US" id="LOVValueDescription{::}Ab0LOV{::}6" >Level 7 Desc</key>
<key locale="en_US" id="LOVValueDescription{::}Ab0LOV{::}7" >Level 8 Desc</key>
<key locale="en_US" id="LOVValueDescription{::}Ab0LOV{::}8" >Level 9 Desc</key>
<key locale="en_US" id="LOVValueDescription{::}Ab0LOV{::}9" >Level 10 esc</key>

</Add>
</TcBusinessDataLocalization>

```

- To add LOV values and sub-LOV attachments to cascading LOVs, enter a command like the following on a single line:

```

bmide_manage_batch_lovs.bat -u=username -p=password -g=dba
-option=update -file=BatchLOV_ValidCascading.xml

```

Note You can add sub-LOV attachments in this way only on externally managed LOVs.

Following is the **BatchLOV_ValidCascading.xml** file used in the example:

```

<?xml version="1.0" encoding="UTF-8" standalone="no"?>

<TcBusinessData xmlns="http://teamcenter.com/BusinessModel/TcBusinessData"
batchXSDVersion="1.0" >
  <Add>
    <TcLOVValueSubLOVAttach conditionName="isTrue" subLOVName="ExcelTemplateRules"
targetLOVName="SampeCascadingBatchLOV" targetValue="v1"/>
    <TcLOVValueSubLOVAttach conditionName="isTrue" subLOVName="SampleBatchLOV"
targetLOVName="SampeCascadingBatchLOV" targetValue="v2"/>
  </Add>
  <Change>
    <TcLOV description="Sample Cascading Batch LOV" lovType="ListOfValuesString"
name="SampeCascadingBatchLOV" usage="Exhaustive" isManagedExternally="true">
      <TcLOVValue description="v1 description" value="v1"/>
      <TcLOVValue description="v2 description" value="v2"/>
    </TcLOV>

    <TcLOV description="Sample Batch LOV" lovType="ListOfValueString"
name="SampeBatchLOV" usage="Suggestive" isManagedExternally="true">
      <TcLOVValue description="sample" value="Sweden"/>
      <TcLOVValue description="sample" value="India"/>
      <TcLOVValue description="sample" value="UK"/>
      <TcLOVValue description="sample" value="USA"/>
      <TcLOVValue description="sample" value="Germany"/>
      <TcLOVValue description="sample" value="South Africa"/>
      <TcLOVValue description="sample" value="Australia"/>
      <TcLOVValue description="sample" value="New Zealand"/>
      <TcLOVValue description="sample" value="UAE"/>
      <TcLOVValue description="sample" value="Pakistan"/>
    </TcLOV>
  </Change>
</TcBusinessData>

```

- To extract all the LOV values, sub-LOV attachments, and localizations from the database, enter a command like the following on a single line:

```

bmide_manage_batch_lovs -u=username -p=password -g=dba
-option=extract -file=BatchLOV_LOV1_extracted.xml

```

Do this when you want to check the current values on any of the LOVs or to use the extracted file as the basis for the next set of changes if the original source input file is no longer available.

- To extract LOV values, sub-LOV attachments, and localizations for specific LOVs in the database, enter a command like the following on a single line:


```
bmid_manage_batch_lovs -u=username -p=password -g=dba  
-option=extract -lovs=BatchLOV_LOV1,BatchLOV_LOV2  
-file=BatchLOV_extracted.xml
```

bmide_manage_templates

Adds solution templates to the database table.

SYNTAX

```
bmide_manage_templates -u=user-id {-p=password |  
pf=password-file} [-g=group]  
-option=option-type -templates=template-names [-h]
```

ARGUMENTS**-u**

Specifies the user ID.

This is generally **infodba** or another user with administration privileges. If this argument is used without a value, the operating system user name is used.

Note

If Security Services single sign-on (SSO) is enabled for your server, the **-u** and **-p** arguments are authenticated externally through SSO rather than being authenticated against the Teamcenter database. If you do not supply these arguments, the utility attempts to join an existing SSO session. If no session is found, you are prompted to enter a user ID and password.

-p

Specifies the password.

If used without a value, the system assumes a null value. This argument is mutually exclusive with the **-pf** argument.

If this argument is not used, the system assumes the *user-ID* value to be the password.

-pf

Specifies the password file. If used without a value, the system assumes a null value. If this argument is not used, the system assumes the *user-ID* value to be the password.

For more information about managing password files, see [Manage password files](#).

This argument is mutually exclusive with the **-p** argument.

-g

Specifies the group associated with the user.

If used without a value, the user's default group is assumed.

-option

Specifies whether to add templates to the database or list the existing templates in the database. Valid values are **add** or **list**.

-templates

Specifies the name of the template to add to the database. Multiple templates can be added by separating the names by a comma.

-h

Displays help for this utility.

ENVIRONMENT

As specified in *Manually configuring your environment for Teamcenter utilities*.

FILES

As specified in *Log files produced by Teamcenter*.

RESTRICTIONS

None.

bmide_postupgradetotc.sh/.bat

Extracts the site's customizations from the database. This utility is executed after upgrading the database to Teamcenter 10.1.

SYNTAX

```
bmide_postupgradetotc.sh -u=user-id {-p=password |  
-pf=password-file} [-g=group]  
-sol_name=solution-name -sol_disp_name=display-name -log=log-file [-h]
```

ARGUMENTS**-u**

Specifies the user ID.

This is generally **infodba** or another user with administration privileges. If this argument is used without a value, the operating system user name is used.

Note

If Security Services single sign-on (SSO) is enabled for your server, the **-u** and **-p** arguments are authenticated externally through SSO rather than being authenticated against the Teamcenter database. If you do not supply these arguments, the utility attempts to join an existing SSO session. If no session is found, you are prompted to enter a user ID and password.

-p

Specifies the password.

If used without a value, the system assumes a null value. This argument is mutually exclusive with the **-pf** argument.

If this argument is not used, the system assumes the *user-ID* value to be the password.

-pf

Specifies the password file. If used without a value, the system assumes a null value. If this argument is not used, the system assumes the *user-ID* value to be the password.

For more information about managing password files, see [Manage password files](#).

This argument is mutually exclusive with the **-p** argument.

-g

Specifies the group associated with the user.

If used without a value, the user's default group is assumed.

-sol_name

Specifies the name of the solution to be created.

-sol_disp_name

Specifies the display name of the solution to be created.

-log

Specifies file path and name of the log file that contains the results of this execution.

-h

Displays help for this utility.

ENVIRONMENT

As specified in *Manually configuring your environment for Teamcenter utilities*.

FILES

As specified in *Log files produced by Teamcenter*.

RESTRICTIONS

None.

bmide_setupknowledgebase

Generates the CLIPS (C Language Integrated Production System) rule file and uploads it to the database. It should only be run after upgrading the database from a pre-Teamcenter 2007 version to Teamcenter 2007 or later version.

SYNTAX

```
bmide_setupknowledgebase [-u=user-id {-p=password |  
-pf=password-file} -g=group]  
-regen=true/false -log=output-file-for-the-log-file [-h]
```

ARGUMENTS**-u**

Specifies the user ID.

This is generally **infodba** or another user with administration privileges. If this argument is used without a value, the operating system user name is used.

Note

If Security Services single sign-on (SSO) is enabled for your server, the **-u** and **-p** arguments are authenticated externally through SSO rather than being authenticated against the Teamcenter database. If you do not supply these arguments, the utility attempts to join an existing SSO session. If no session is found, you are prompted to enter a user ID and password.

-p

Specifies the password.

If used without a value, the system assumes a null value. This argument is mutually exclusive with the **-pf** argument.

If this argument is not used, the system assumes the *user-ID* value to be the password.

-pf

Specifies the password file. If used without a value, the system assumes a null value. If this argument is not used, the system assumes the *user-ID* value to be the password.

For more information about managing password files, see [Manage password files](#).

This argument is mutually exclusive with the **-p** argument.

-g

Specifies the group associated with the user.

If used without a value, the user's default group is assumed.

-regen

Regenerates the CLIPS file. Values can be **true** or **false**.

-log

Specifies file path and name of the log file that contains the results of this execution. This argument is optional.

-h

Displays help for this utility.

ENVIRONMENT

As specified in *Manually configuring your environment for Teamcenter utilities*.

FILES

As specified in *Log files produced by Teamcenter*.

RESTRICTIONS

None.

business_model_extractor

Extracts the data model definitions in the database into a XML file.

SYNTAX

business_model_extractor
[-u=*user-id* {-p=*password* | -pf=*password-file*} -g=*group*]
-outfile=*output-file*
-mode=[all | schema | localization]
-log=*log-file*
-stats
[-h]

ARGUMENTS**-u**

Specifies the user ID.

This is generally **infodba** or another user with administration privileges. If this argument is used without a value, the operating system user name is used.

Note

If Security Services single sign-on (SSO) is enabled for your server, the **-u** and **-p** arguments are authenticated externally through SSO rather than being authenticated against the Teamcenter database. If you do not supply these arguments, the utility attempts to join an existing SSO session. If no session is found, you are prompted to enter a user ID and password.

-p

Specifies the password.

If used without a value, the system assumes a null value. This argument is mutually exclusive with the **-pf** argument.

If this argument is not used, the system assumes the *user-ID* value to be the password.

-pf

Specifies the password file. If used without a value, the system assumes a null value. If this argument is not used, the system assumes the *user-ID* value to be the password.

For more information about managing password files, see [Manage password files](#).

This argument is mutually exclusive with the **-p** argument.

-g

Specifies the group associated with the user.

If used without a value, the user's default group is assumed.

-outfile

Specifies file path and name of the XML file to be created with business data.

-mode

Specifies the extraction mode:

- **all**

Extracts the entire model. This is the default mode.

- **schema**

Extracts the classes and attributes.

- **localization**

Extracts all localizations in the localization's format.

-log

Specifies file path and name of the log file that contains the results of this execution.

-h

Displays help for this utility.

ENVIRONMENT

As specified in [Manually configuring your environment for Teamcenter utilities](#).

FILES

As specified in [Log files produced by Teamcenter](#).

RESTRICTIONS

None.

EXAMPLES

To extract the entire datamodel in the database and output to the **db_extract.xml** file:

```
business_model_extractor -u=infodba -p=infodba -g=dba -mode=all  
-outfile=c:\temp\db_extract.xml
```

business_model_updater

Deploys **tcschema**, types, and business rules.

Note Database statistics must be updated following any **business_model_updater** actions that cause indexes to be created or updated. Without updated statistics, the indexes are not utilized correctly by the database. For example:

```
exec dbms_stats.gather_schema_stats
(ownname=>'INFODBA',estimate_percent=>100,method_opt=>'FOR
ALL COLUMNS SIZE 1',degree=>8,cascade=>true,no_invalidate=>FALSE);
```

For information about updating database statistics, see the [Site Consolidation Guide](#).

SYNTAX

```
business_model_updater [-u=user-id {-p=password |
-pf=password-file} -g=group]
-mode= [install | upgrade]
-update= [ all | non_schema | schema | main_types | types | rules |
bmf_rules_skip_missing_types | bmf_rules | lovs | lov_attaches |
non_schema_ignore_lov_attach | convert_to_primary]
-process= [add | delete | change | all]
-file=XML-file-path-name
-log=log-file-path-name
[-mergedatasettype] [-h]
```

ARGUMENTS**-u**

Specifies the user ID.

This is generally **infodba** or another user with administration privileges. If this argument is used without a value, the operating system user name is used.

Note If Security Services single sign-on (SSO) is enabled for your server, the **-u** and **-p** arguments are authenticated externally through SSO rather than being authenticated against the Teamcenter database. If you do not supply these arguments, the utility attempts to join an existing SSO session. If no session is found, you are prompted to enter a user ID and password.

-p

Specifies the password.

If used without a value, the system assumes a null value. This argument is mutually exclusive with the **-pf** argument.

If this argument is not used, the system assumes the *user-ID* value to be the password.

-pf

Specifies the password file. If used without a value, the system assumes a null value. If this argument is not used, the system assumes the *user-ID* value to be the password.

For more information about managing password files, see [Manage password files](#).

This argument is mutually exclusive with the **-p** argument.

-g

Specifies the group associated with the user.

If used without a value, the user's default group is assumed.

-mode

Specifies the update method:

install

Specifies the update is for an installation of a new database.

upgrade

Specifies the update is for upgrading from one version to another version.

-update

Specifies which Business Modeler IDE object to update.

If you specify **non_schema_ignore_lov_attach**, the utility skips **TcLOVAttach** objects and processes all other **non_schema** objects.

-process

Specifies which Business Modeler IDE process to update.

-file

Specifies the path name to the XML file generated by Business Modeler IDE. The file contains Business Modeler IDE object definition data such as **Class**, **Type**, and **LOV** attachments.

-log

Specifies the path name to the log file.

-mergedatasettype

Specifies that dataset type definitions must be merged with existing dataset type definitions.

-h

Displays help for this utility.

ENVIRONMENT

As specified in [Manually configuring your environment for Teamcenter utilities](#).

FILES

As specified in [Log files produced by Teamcenter](#).

RESTRICTIONS

None.

change_datasets

Modifies the dataset reference name and the associated tool. This utility is run when resolving dataset type name collisions.

For more information, see the [Business Modeler IDE Guide](#).

SYNTAX

change_datasets [-u=*user-id* {-p=*password* | -pf=*password-file*} -g=*group*]
-ds_info=*change-dataset-information-xml-file-location*
-keep_definition
-dry
-bulk
[-h]

ARGUMENTS**-u**

Specifies the user ID.

This is generally **infodba** or another user with administration privileges. If this argument is used without a value, the operating system user name is used.

Note

If Security Services single sign-on (SSO) is enabled for your server, the **-u** and **-p** arguments are authenticated externally through SSO rather than being authenticated against the Teamcenter database. If you do not supply these arguments, the utility attempts to join an existing SSO session. If no session is found, you are prompted to enter a user ID and password.

-p

Specifies the password.

If used without a value, the system assumes a null value. This argument is mutually exclusive with the **-pf** argument.

If this argument is not used, the system assumes the *user-ID* value to be the password.

-pf

Specifies the password file. If used without a value, the system assumes a null value. If this argument is not used, the system assumes the *user-ID* value to be the password.

For more information about managing password files, see [Manage password files](#).

This argument is mutually exclusive with the **-p** argument.

-g

Specifies the group associated with the user.

If used without a value, the user's default group is assumed.

-ds_info

Specifies the full path to the **ChangeDatasetInfo.xml** file generated by the utility.

-keep_definition

Specifies to keep the original tool and reference definition.

-dry

Logs information about modifications. No actual modifications are done with this argument. This is an optional argument.

-bulk

Runs a bulk update. This is an optional argument.

-h

Displays help for this utility.

ENVIRONMENT

As specified in [Manually configuring your environment for Teamcenter utilities](#).

FILES

As specified in [Log files produced by Teamcenter](#).

RESTRICTIONS

None.

clips_dataset_upload

Stores both the ASCII (text) version and binary version of the CLIPS knowledge base files into the CLIPS (singleton) dataset in Teamcenter. This utility is called during Business Modeler IDE deployment activities when changes are identified to condition objects or rules based framework objects (application extension point and application extension rules). Only administrative users are allowed to run this utility.

SYNTAX

clips_dataset_upload [-u=*user-id* {**-p**=*password* | **-pf**=*password-file*} **-g**=*group*]
-file=*path-to-upload-file*
-log=*output-file-for-the-log-file*
-force_uncheckout=*true/false*
[-h]

ARGUMENTS**-u**

Specifies the user ID.

This is generally **infodba** or another user with administration privileges. If this argument is used without a value, the operating system user name is used.

Note

If Security Services single sign-on (SSO) is enabled for your server, the **-u** and **-p** arguments are authenticated externally through SSO rather than being authenticated against the Teamcenter database. If you do not supply these arguments, the utility attempts to join an existing SSO session. If no session is found, you are prompted to enter a user ID and password.

-p

Specifies the password.

If used without a value, the system assumes a null value. This argument is mutually exclusive with the **-pf** argument.

If this argument is not used, the system assumes the *user-ID* value to be the password.

-pf

Specifies the password file. If used without a value, the system assumes a null value. If this argument is not used, the system assumes the *user-ID* value to be the password.

For more information about managing password files, see [Manage password files](#).

This argument is mutually exclusive with the **-p** argument.

-g

Specifies the group associated with the user.

If used without a value, the user's default group is assumed.

-file

Specifies the full path to the file specified for upload. Read access must be allowed on the directory.

-log

Specifies file path and name of the log file that contains the results of this execution. This argument is optional.

-force_uncheckout

Cancel checkout of the CLIPS rules dataset and continue processing if set to **true**. If set to **false**, return an error if the CLIPS rules dataset is checked out. Values can be **true** or **false**.

This is typically used to restore the CLIPS rules dataset if left checked out in error.

-h

Displays help for this utility.

ENVIRONMENT

As specified in [Manually configuring your environment for Teamcenter utilities](#).

FILES

As specified in [Log files produced by Teamcenter](#).

RESTRICTIONS

None.

convert_forms

Allows a user with DBA privileges to convert legacy file-based forms to storage-based forms. You can manage the conversion process, as follows:

1. Determine whether a given form type or all form types to be converted.
2. Define attribute mapping between the file-based and storage-based forms.
3. Control the conversion process by specifying the number of forms to be converted during each run of the utility. Before performing the conversion, you can run the utility to generate an output file containing the UIDs of the file-based forms. This allows you to formulate a plan for performing the conversion by distributing the workload between multiple runs on multiple machines, if necessary.
4. Run the utility in batch mode without user intervention.
5. Restart the process without data corruption in the event that the process is stopped or terminates abnormally.
6. Generate log files listing information about forms that were successfully converted, failed to convert, and which attribute values are dropped or truncated. These files are retained if the process is terminated before completion.

**UPGRADING
FORMS**

Updating file-based forms to storage-based forms involves the following steps:

1. Generate a file containing the list of forms to be converted, by running the **convert_forms** utility, as follows:

```
convert_forms  
-identify -output_file=file-name [-type=type-name]
```

This produces an output file containing the UIDs of file-based forms, one per line. The output file must be opened in **append** mode. This allows multiple lists of form type information to be contained in the same file. If the **-type** argument is not specified, all file-based forms are included in the output file.

2. Run the utility, as required, from one or multiple machines, as follows:

```
convert_forms  
-convert -process_file=file-name [-input_options=file-name]
```

3. If errors occur during the conversion process, the UIDs of the forms that were not converted are listed in the error file. After identifying and correcting the errors, you can use the **ErrorFile** file as the input file when rerunning the utility to convert the forms.

The **-process_file** argument specifies a file containing information that could be specific to each job. The **-input_options** argument specifies a file that is common to all jobs.

PROCESS OPTIONS FILE

The **-process_file** argument specifies a file containing information related to specific runs of the utility. You can copy the following example, paste it into a text editor, and use it as a starting point for your process options file:

```
<?xml version="1.0" encoding="iso-8859-1"?>
<ProcessInfo xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:noNamespaceSchemaLocation="convertFormProcessInfo.xsd">
  <InputFile>C:\\temp\\form_uids.txt</InputFile>
  <StartLine>1</StartLine>
  <EndLine>100000</EndLine>
  <LogFile>C:\\temp\\log.txt</LogFile>
  <ErrorFile>C:\\temp\\error.txt</ErrorFile>
  <SuccessFile>C:\\temp\\success.txt</SuccessFile>
</ProcessInfo>
```

InputFile	Specifies either the file generated as output when the utility is run with the -identify option or the ErrorFile file generated during a conversion run.
StartLine	The line number in the InputFile that specifies the beginning of the block of forms to be converted. If the line number is not specified, the default value is 1 .
EndLine	The line number in the InputFile file that specifies the end of the block of forms to be converted. If the line number is not specified, the default end line is the end of the file.
LogFile	Specifies the name of the file that logs information about dropped or truncated attribute values.
ErrorFile	Specifies the name of the file containing the UIDs of forms that were not converted due to errors. Errors encountered during conversion do not stop the process. When the reasons for the failure have been identified and corrected, the ErrorFile file can be used as the input file when the utility is rerun to convert those forms.
SuccessFile	Contains UIDs of forms that were successfully converted. This file can be useful for multi-site conversions.

All three files, **LogFile**, **ErrorFile**, and **SuccessFile**, are optional. If not specified, the corresponding file is not generated.

INPUT OPTIONS FILE

Unlike the process options file, which is specific to a particular run of the utility, the file specified by the **-input_options** argument contains information that is common to all runs of the utility. The following example illustrates the format of an input options file:

```
<FormTypes>
  <Type name=<name>>
    <DropAttrs action=none | all | unmapped | DropList>
      <ImanFileAttr name=<name> log=no | yes />
      .....
    </DropAttrs>
    <MapAttrs>
      <Map ImanFileAttr=<name> POMAttr=<name> truncate=no | yes | log />
      .....
    </MapAttrs>
    <KeepLastModified action=no | yes />
    <DeleteImanFile action=yes | no />
  </Type>
  .....
</FormTypes>
```

All element attributes in the file have default values, indicated in *italics* in the example. If all default values are assumed, the input options file can be omitted.

Type	<p>One or more Type elements can exist in the mapping file.</p> <p>Forms to be converted can be of different form types, all of which are listed in the file. If a form has no corresponding Type element, all form ImanFile attributes are mapped to the corresponding POM storage class attributes with the same names.</p>
DropAttrs	<p>Each DropAttrs element can contain zero or more DropList elements.</p> <p>DropAttrs action=<i>none</i> indicates that no attributes are dropped, all indicates that all attributes are dropped and no storage object needs to be created. unmapped indicates that all unmapped attributes are dropped. DropList indicates that a list is created of attributes that are dropped.</p>
ImanFileAttr	<p>To use the ImanFileAttr element, the DropList action must be used for the DropAttrs element.</p> <p>The ImanFileAttr element has a log attribute. If the value of this attribute is yes, the dropped attribute name and value are written to the log file.</p>

- MapAttrs** Each **MapAttrs** element can contain one or more **Map** elements. The **Map** element allows an **ImanFile** attribute to be mapped to a POM storage class attribute with a different name. If an **ImanFile** attribute is not included in this list, it is mapped to the storage class attribute with the same name.
- The **Map** element allows you to truncate the string data on conversion. If the **truncate** attribute value is **no** an error occurs and the form is not converted. If the value of the **truncate** attribute is **yes**, the data is truncated and not logged on the log file, while the **log** attribute will truncate the data and log it in the log file.
- Only primitive attribute types are supported by this utility. Date, typed, and untyped references are not supported.
- Do not create empty storage objects. Create an object only if one or more values are copied from the **ImanFile** properties.
- Form attributes are case sensitive.
- KeepLastModified** The **KeepLastModified** element specifies whether the last modified date must be updated to reflect the time of conversion. The default value is to update the last-modified date.
- DeleteImanFile** The **DeleteImanFile** element specifies whether to delete the **ImanFile** after conversion. The default value is to delete the file.

You can copy the following example, paste it into a text editor, and use it as a starting point for your input options file:

```
<?xml version="1.0" encoding="iso-8859-1"?>
<FormTypes xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:noNamespaceSchemaLocation="convertFormInputOptions.xsd">
  <Type name="UGPartAttr">
    <DropAttrs action="DropList">
      <ImanFileAttr name="ASSEMB_NO" log="yes" />
      <ImanFileAttr name="MODEL" log="yes" />
    </DropAttrs>
    <MapAttrs>
      <Map ImanFileAttr="CREATOR" POMAttr="CREATOR" truncate="log" />
    </MapAttrs>
    <KeepLastModified action="no" />
    <DeleteImanFile action="no" />
  </Type>
</FormTypes>
```

SYNTAX

```
convert_forms [-u=user-id {-p=password | -pf=password-file} -g=group]
-identify -output_file=file-name [-type=form-type] -convert
-process_file=file-name [-input_options=file-name]
```

ARGUMENTS**-u**

Specifies the user ID.

This is generally **infodba** or another user with administration privileges. If this argument is used without a value, the operating system user name is used.

Note

If Security Services single sign-on (SSO) is enabled for your server, the **-u** and **-p** arguments are authenticated externally through SSO rather than being authenticated against the Teamcenter database. If you do not supply these arguments, the utility attempts to join an existing SSO session. If no session is found, you are prompted to enter a user ID and password.

-p

Specifies the password.

If used without a value, the system assumes a null value.

If this argument is not used, the system assumes the *user-ID* value to be the password.

This argument is mutually exclusive with the **-pf** argument.

-pf

Specifies the password file. If used without a value, the system assumes a null value. If this argument is not used, the system assumes the *user-ID* value to be the password.

For more information about managing password files, see [Manage password files](#).

This argument is mutually exclusive with the **-p** argument.

-g

Specifies the group associated with the user.

If used without a value, the user's default group is assumed.

-identify

Generates an output file containing the UIDs of file-based forms. This argument is used in conjunction with the **-output_file** argument.

-output_file

Specifies the name of the output file.

-type

Specifies the type of the forms to be converted. If not specified, all file-based forms are converted.

-convert

Converts file-based forms that are read from a previous run of the utility using the **-identify** option. The **-convert** argument is used with the **-process_file** and **-input_options** arguments.

-process_file

Specifies the process file containing names of the input file, log file, error file, and success file, as well as the start line number and end line number. For more information about this file, see [Process options file](#).

-input_options

Specifies the name of an input file containing information about how to convert forms. For more information about this file, see [Input options file](#). This argument is optional.

-h

Displays help for this utility.

ENVIRONMENT

As specified in [Manually configuring your environment for Teamcenter utilities](#).

FILES

The **convertFormProcessInfo.xsd** and **convertFormInputOptions.xsd** XML schema files are delivered as part of your Teamcenter installation and are located in the **imandata** directory. You must use these schema files to process the XML files that you generate.

RESTRICTIONS

None.

EXAMPLES

- To output the UIDs of all file-based forms of **UGPartAttr** type into the **C:\temp\form_uids.txt** file, enter the following command on a single line:

```
convert_forms -u=infodba -p=infodba -g=dba -identify
               -output_file=C:\temp\form_uids.txt -type=UGPartAttr
```

The **C:\temp\form_uids.txt** file is used in the example in [Process options file](#).

- To convert file-based forms, reading in the **C:\temp\process_info.txt** file containing process information and the **C:\temp\options_file.txt** containing conversion information, enter the following command on a single line:

```
convert_forms -u=infodba -p=infodba -g=dba -convert
               -process_file=C:\temp\process_info.txt
               -input_options=C:\temp\input_options.txt
```

deploy_archive

Archives deployment files for both Teamcenter Environment Manager (TEM) and the Business Modeler IDE. An administrator can run this utility from the command line in **retrieve** mode to retrieve the last set of deployment archive files.

This utility is automatically run at deployment to archive files, but you can turn off this functionality by setting the **DEPLOY_ARCHIVE_ACTIVE** environment variable to **false** (or **off**, **no**, or **0**).

The deployment archival process is as follows:

1. A set of extractor and updater utility log files (specified by an input directory) are zipped into an archive file.
2. A set of compare history files associated to the specific deployment are zipped into an archive file.
3. For every deployment, the zipped archive files are uploaded into a new deploy archive dataset (**Fnd0BMIDEDeployArchive** dataset type).

The deployment archive file retrieval process is as follows:

1. Locate the archive datasets (**Fnd0BMIDEDeployArchive** dataset type) from the Teamcenter database using the **BMIDEDeployArchiveRecovery** saved query.
2. A subdirectory using the dataset name with the create date/time stamp is created under a specified target directory.
3. Each named reference (archive zip file) for the archive dataset is exported to that subdirectory.

For more information, see the [Business Modeler IDE Guide](#).

SYNTAX

```
deploy_archive -u=user-id {-p=password | -pf=password-file} [-g=group]
-action=[deploy | retrieve]
-dir=source-or-target-file-path
-deploySource=[tem | bmide]
-retrieveNum=number-of-archives
-afterDate=date-and-time
-log=log-file-path
[-h]
```

ARGUMENTS

-u

Specifies the user ID.

This is generally **infodba** or another user with administration privileges. If this argument is used without a value, the operating system user name is used.

Note If Security Services single sign-on (SSO) is enabled for your server, the **-u** and **-p** arguments are authenticated externally through SSO rather than being authenticated against the Teamcenter database. If you do not supply these arguments, the utility attempts to join an existing SSO session. If no session is found, you are prompted to enter a user ID and password.

-p

Specifies the password.

If used without a value, the system assumes a null value. This argument is mutually exclusive with the **-pf** argument.

If this argument is not used, the system assumes the *user-ID* value to be the password.

-pf

Specifies the password file. The file must be a single-line ASCII file containing the password in clear text. Teamcenter Environment Manager prompts you for a password and creates the password file during installation.

If used without a value, the system assumes a null value. This argument is mutually exclusive with the **-p** argument.

If this argument is not used, the system assumes the *user-ID* value to be the password.

-g

Specifies the group associated with the user.

If used without a value, the user's default group is assumed.

-action

Creates or retrieves the archive. You must specify one of these options:

- **deploy**

Collects and creates the deploy archive ZIP files and upload them into a deploy archive dataset. Requires the **-deploySource** argument to be specified.

- **retrieve**

Extracts the deploy archive ZIP files from the specified deploy archive datasets. The dataset name is used to create a subdirectory in the OS under the directory name given by the **-dir** parameter. The named references (archive ZIP files) are exported to that subdirectory.

-dir

Specifies the full path to the source directory (when using the **-action=deploy** argument) or to the target directory (when using the **-action=retrieve** argument). The source directory must exist and have read access and contains files to archive for the deploy action. The target directory must exist and have write access and contains subdirectories (by dataset name) that each contain ZIP archive files from the retrieve action.

-deploySource

Identifies the source of the deployment activity when using the **-action=deploy** argument. (Ignored when the action is **retrieve**.) The dataset name format is **deploy_tem | bmid_user-id_date-time-stamp**. Valid values are:

- **tem**
Specifies that Teamcenter Environment Manager is the source of the deployment.
- **bmide**
Specifies that the Business Modeler IDE is the source of the deployment.

-retrieveNum

Qualifies the number of archive datasets (from latest to earliest) to return from the saved query execution when the **-action=retrieve** argument is used. (Ignored when the action is **deploy**.) The default is **3**. If not specified, 3 datasets are retrieved. If **0** is set, then all datasets are retrieved. The actual number of datasets exported can be different than this value based on the total number of deploy archive datasets.

-afterDate

Specifies the time after which to collect archive log files when using the **-action=deploy** argument. (Ignored when the action is **retrieve**.) The required format is *yyyy-mm-dd hh:mm:ss*.

-log

Specifies file path and name of the log file that contains the results of this execution. This argument is optional.

-h

Displays help for this utility.

ENVIRONMENT

As specified in [Manually configuring your environment for Teamcenter utilities](#).

FILES

As specified in [Log files produced by Teamcenter](#).

RESTRICTIONS

None.

execute_rbf_rules

Executes the specified application extension rule. The utility validates the application extension rules are deployed and functioning correctly. Input parameters for the rule execution are specified in a file. Output is written to a specified file or displayed on the console. Execution details are written to a specified log file or displayed on the console.

DESCRIPTION SYNTAX

execute_rbf_rules **-u**=*user-id* {**-p**=*password* | **-pf**=*password-file*} [**-g**=*group*]
-id=*application-extension-point-id* **-inputfile**=*input-file-name*
-outputfile=*output-file-name* [**-log**=*log-file-name*] [**-h**]

ARGUMENTS

-u

Specifies the user ID.

This is generally **infodba** or another user with administration privileges. If this argument is used without a value, the operating system user name is used.

Note

If Security Services single sign-on (SSO) is enabled for your server, the **-u** and **-p** arguments are authenticated externally through SSO rather than being authenticated against the Teamcenter database. If you do not supply these arguments, the utility attempts to join an existing SSO session. If no session is found, you are prompted to enter a user ID and password.

-p

Specifies the password.

If used without a value, the system assumes a null value. This argument is mutually exclusive with the **-pf** argument.

If this argument is not used, the system assumes the *user-ID* value to be the password.

-pf

Specifies the password file. If used without a value, the system assumes a null value. If this argument is not used, the system assumes the *user-ID* value to be the password.

For more information about managing password files, see [Manage password files](#).

This argument is mutually exclusive with the **-p** argument.

-g

Specifies the group associated with the user.

If used without a value, the user's default group is assumed.

-id

Specifies the ID of the application extension point.

-inputfile

Specifies the location of the input file. The input file format is:

Column Name | Datatype | Value

There is one line for each input parameter.

Datatype is one of the following:

String
Integer
Double
Float
Boolean
Date
Tag

See *Examples* for an example of the input file.

-outputfile

Specifies the location of the output file. The output file format is the same as the input file format. Datatype for a output parameter is one of the following:

String
Integer
Double
Float
Boolean
Date

-log

Specifies the location of the log file.

-h

Displays help for this utility.

ENVIRONMENT

As specified in [Manually configuring your environment for Teamcenter utilities](#).

FILES

As specified in [Log files produced by Teamcenter](#).

RESTRICTIONS

None.

EXAMPLES

- The following is an example of the utility:

```
execute_rbf_rules -u=infodba -p=infodba -g=dba
-id=tc.core.pse.icon -inputfile="d:\in1.txt"
-outputfile="d:\out1.txt" -log="d:\rbf1.log"
```

- The following input file example specifies that there are two input columns, **Material** and **Pressure**, with corresponding datatypes of **String** and **Integer**. The value of **Material** is **Steel**; the value of **Pressure** is **10**.

```
Material | String | Steel
Pressure | Integer | 10
```

getglobalconstantvalue

Returns the value of a particular global constant in the database. The utility provides help in troubleshooting issues on the server once the global constants are deployed. The utility accepts the name of a global constant and outputs the value of the constant, if present.

SYNTAX

getglobalconstantvalue [-u=*user-id* {-p=*password* | -pf=*password-file*} -g=*group*] -constantname=*global-constant-name* [-v= **on** | **off**] [-h]

ARGUMENTS

-u

Specifies the user ID.

This is generally **infodba** or another user with administration privileges. If this argument is used without a value, the operating system user name is used.

Note If Security Services single sign-on (SSO) is enabled for your server, the **-u** and **-p** arguments are authenticated externally through SSO rather than being authenticated against the Teamcenter database. If you do not supply these arguments, the utility attempts to join an existing SSO session. If no session is found, you are prompted to enter a user ID and password.

-p

Specifies the password.

If used without a value, the system assumes a null value. This argument is mutually exclusive with the **-pf** argument.

If this argument is not used, the system assumes the *user-ID* value to be the password.

-pf

Specifies the password file. If used without a value, the system assumes a null value. If this argument is not used, the system assumes the *user-ID* value to be the password.

For more information about managing password files, see [Manage password files](#).

This argument is mutually exclusive with the **-p** argument.

-g

Specifies the group associated with the user.

If used without a value, the user's default group is assumed.

-constantname

Specifies the name of the global constant.

-v

Specifies verbose mode. Specify **on** to display a detailed message. Specify **off** to display only the value. Default value is **on**.

-h

Displays help for this utility.

ENVIRONMENT

As specified in *Manually configuring your environment for Teamcenter utilities*.

FILES

As specified in *Log files produced by Teamcenter*.

RESTRICTIONS

None.

EXAMPLES

Entering the following command:

```
getglobalconstantvalue -u=infodba -p=infodba -g=dba  
-constantname=SampleConstant
```

returns the following:

```
Value of the global constant SampleConstant = XYZ
```

get_key_definition

Gets the multifold key definition for a business object type. *Multifold keys* are identifiers assigned to each object to ensure their uniqueness in the database.

At a command prompt, type:

```
get_key_definition -class=business-object-type
```

The results are displayed as:

```
Key Definition for business-object-type is multifoldkey-definition
```

Note There are other utilities that you can use to work with keys. Use the [get_key_string](#) utility to get the multifold key value of an item instance as a string containing property names and values, and use the [mfk_update](#) utility to update multifold key definitions in the key table in the database.

For more information, see the [Business Modeler IDE Guide](#).

SYNTAX

```
get_key_definition  
-class=business-object-type  
[-h]
```

ARGUMENTS

-class

The name of the business object type for which you want to get the multifold key definition.

-h

Displays help for this utility.

ENVIRONMENT

As specified in [Manually configuring your environment for Teamcenter utilities](#).

FILES

As specified in [Log files produced by Teamcenter](#).

RESTRICTIONS

None.

EXAMPLES

- To get the multifold key definition for a business object type named **T5_MyItem**:

```
get_key_definition -class=T5_MyItem
```

The results are displayed as follows:

```
Key Definition for T5_MyItem is T5_MyItem{item_id,object_type}
```

get_key_string

Gets the multifold key value of an item instance as a string containing property names and values. This utility can be run on the **Item** business object or any of its children. *Multifold keys* are identifiers assigned to each object to ensure their uniqueness in the database.

If the **-item** or **-key** argument input corresponds to more than one item, the key values for each of the items found is displayed. If the **-key** argument is used, partial keys are allowed. For example, if the key for an item is defined as **item_id,prop1,prop2**, inputs such as **-key=item_id=123456,prop1=abcd** are allowed, which may result in multiple items being found and displayed.

To use the **-item** argument to get key values for an item, use the following command:

```
get_key_string -item=item-ID
```

The results are displayed as:

```
Item Type is business-object-name, Key String is
property1=property1-value,property2=property2-value,etc
```

If other properties are added to the multifold key definition for an item business object, they are displayed separated by commas. For example, if the **item_id** and **object_type** properties comprise the multifold key for a custom item business object named **T5_MyItem**, the output of the command is:

```
Item Type is T5_MyItem, Key String is item_id=000016,object_type=T5_MyItem
```

To use the **-key** argument to get the business object type for an item, use the following command:

```
get_key_string -key=property=property-value
```

The results are displayed as:

```
Item Type is business-object-name, Key String is property=property-value
```

- Note**
- There are other utilities that you can use to work with keys. Use the [get_key_definition](#) utility to get the key definition for a business object type, and use the [mfk_update](#) utility to update multifold key definitions in the key table in the database.
 - To change the delimiters used in the input for the **get_key_string** utility, create the [TC_KEY_STRING_DELIMITER_VALUES](#) preference.

For more information, see the [Preferences and Environment Variables Reference](#).

For more information, see the [Business Modeler IDE Guide](#).

SYNTAX

```
get_key_string
-item=business-object-item-ID
-key=key-string
[-h]
```

ARGUMENTS**-item**

Specifies the ID of items for which key values are to be displayed.

-key

Specifies the key value of items for which item types are to be displayed.

-h

Displays help for this utility.

ENVIRONMENT

As specified in [Manually configuring your environment for Teamcenter utilities](#).

FILES

As specified in [Log files produced by Teamcenter](#).

RESTRICTIONS

None.

EXAMPLES

- To get the values of the multifield key properties on an **Item** business object instance whose item ID is **000016**:

```
get_key_string -item=000016
```

The results are displayed as follows:

```
Item Type is Item, Key String is item_id=000016,object_type=Item
```

- To get the values of the multifield key properties on an instance of a custom item business object (**T5_MyItem**) whose item ID is **000019**:

```
get_key_string -item=000019
```

The results are displayed as follows:

```
Item Type is T5_MyItem, Key String is item_id=000019,object_type=T5_MyItem
```

- To get the item type of an instance whose key is **item_id=000016**:

```
get_key_string -key=item_id=000016
```

The results are displayed as follows:

```
Item Type is Item, Key String is item_id=000016
```

getpropertyconstantvalue

Returns the value of a property constant on a particular business object and property in the database. The utility provides help in troubleshooting issues on the server once the property constants are deployed to the database. The utility also applies the inheritance and scope of the property constants while fetching the value. Therefore, it removes the burden of manually analyzing the extracted model to troubleshoot a property constants value on a given business object and property. The utility accepts the name of a property constant, business object and property and outputs the value of the constant, if present.

SYNTAX

```
getpropertyconstantvalue [-u=user-id {-p=password |  
-pf=password-file} -g=group]  
-constantname=constant-property-name -typename=type-object-name  
-propertyname=property-name [-v= on | off] [-h]
```

ARGUMENTS**-u**

Specifies the user ID.

This is generally **infodba** or another user with administration privileges. If this argument is used without a value, the operating system user name is used.

Note

If Security Services single sign-on (SSO) is enabled for your server, the **-u** and **-p** arguments are authenticated externally through SSO rather than being authenticated against the Teamcenter database. If you do not supply these arguments, the utility attempts to join an existing SSO session. If no session is found, you are prompted to enter a user ID and password.

-p

Specifies the password.

If used without a value, the system assumes a null value. This argument is mutually exclusive with the **-pf** argument.

If this argument is not used, the system assumes the *user-ID* value to be the password.

-pf

Specifies the password file. If used without a value, the system assumes a null value. If this argument is not used, the system assumes the *user-ID* value to be the password.

For more information about managing password files, see [Manage password files](#).

This argument is mutually exclusive with the **-p** argument.

-g

Specifies the group associated with the user.

If used without a value, the user's default group is assumed.

-constantname

Specifies the constant property name.

-typename

Specifies the name of the business object.

-propertyname

Specifies the name of a property on the business object specified by the **-typename** argument.

-v

Specifies verbose mode. Specify **on** to display a detailed message. Specify **off** to display only the value. Default value is **on**.

-h

Displays help for this utility.

ENVIRONMENT

As specified in [Manually configuring your environment for Teamcenter utilities](#).

FILES

As specified in [Log files produced by Teamcenter](#).

RESTRICTIONS

None.

EXAMPLES

Entering the following command:

```
propertyconstantvalue -u=infodba -p=infodba -g=dba  
-constantname=Visible -typename=Item -property=item_id
```

returns the following:

```
Value of the property constant Visible on Item.item_id = true
```

gettypeconstantvalue

Returns the value of type constant, that is, a business object constant, on a particular business object in the database. The utility provides help in troubleshooting issues on the server once the type constants are deployed to the database. The utility also applies the inheritance and scope of the type constant while fetching the value. Therefore, it removes the burden of manually analyzing the extracted model to troubleshoot a type constants value on a given business object. The utility accepts the name of a type constant and business object and outputs the value of the constant, if present.

SYNTAX

gettypeconstantvalue [-u=*user-id* {-p=*password* | -pf=*password-file*} -g=*group*]
-constantname=*type-constant-name* -typename=*type-name* [-v= on | off] [-h]

ARGUMENTS**-u**

Specifies the user ID.

This is generally **infodba** or another user with administration privileges. If this argument is used without a value, the operating system user name is used.

Note

If Security Services single sign-on (SSO) is enabled for your server, the **-u** and **-p** arguments are authenticated externally through SSO rather than being authenticated against the Teamcenter database. If you do not supply these arguments, the utility attempts to join an existing SSO session. If no session is found, you are prompted to enter a user ID and password.

-p

Specifies the password.

If used without a value, the system assumes a null value. This argument is mutually exclusive with the **-pf** argument.

If this argument is not used, the system assumes the *user-ID* value to be the password.

-pf

Specifies the password file. If used without a value, the system assumes a null value. If this argument is not used, the system assumes the *user-ID* value to be the password.

For more information about managing password files, see [Manage password files](#).

This argument is mutually exclusive with the **-p** argument.

-g

Specifies the group associated with the user.

If used without a value, the user's default group is assumed.

-constantname

Specifies the name of the type constant.

-typename

Specifies the name of the business object.

-v

Specifies verbose mode. Specify **on** to display a detailed message. Specify **off** to display only the value. Default value is **on**.

-h

Displays help for this utility.

ENVIRONMENT

As specified in [Manually configuring your environment for Teamcenter utilities](#).

FILES

As specified in [Log files produced by Teamcenter](#).

RESTRICTIONS

None.

EXAMPLES

Entering the following command:

```
gettypeconstantvalue -u=infodba -p=infodba -g=dba  
-constantname=SampleConstant -typename=WorkspaceObject
```

returns the following:

```
Value of the type constant SampleConstant on type  
WorkspaceObject = QWERTY
```

manage_icon_files

Manages uploading icon files to the database. *Icons* are images placed on business object instances to identify them in the user interface.

When templates are deployed to a server from the Business Modeler IDE or installed using Teamcenter Environment Manager, the template's zipped icon files are placed into the *TC_DATA/model/icons* (**TC_DATA_MODEL**) directory. At deployment or installation time, the **manage_icon_files** utility is used internally by Teamcenter to upload the icons into **Fnd0IconResource** dataset instances in the database. To search for these datasets using the rich client, search for the **Icon Resource Dataset** type.

For more information, see the [Business Modeler IDE Guide](#).

Warning This utility is used internally only. Do not run this utility yourself.

SYNTAX

```
manage_icon_files -u=user-id {-p=password | -pf=password-file} [-g=group]
-option=upload
-template=template-name
[-h]
```

ARGUMENTS**-u**

Specifies the user ID.

This is generally **infodba** or another user with administration privileges. If this argument is used without a value, the operating system user name is used.

Note If Security Services single sign-on (SSO) is enabled for your server, the user and password arguments are authenticated externally through SSO rather than being authenticated against the Teamcenter database. If you do not supply these arguments, the utility attempts to join an existing SSO session. If no session is found, you are prompted to enter a user ID and password.

-p

Specifies the user's password.

If used without a value, the system assumes a null value. If this argument is not used, the system assumes the *user-ID* value to be the password.

This argument is mutually exclusive with the **-pf** argument.

-pf

Specifies the password file. If used without a value, the system assumes a null value. If this argument is not used, the system assumes the *user-ID* value to be the password.

For more information about managing password files, see [Manage password files](#).

This argument is mutually exclusive with the **-p** argument.

-g

Specifies the group associated with the user.

If used without a value, the user's default group is assumed.

-option=upload

Uploads all icon files from the *TC_DATA_MODEL/icons* directory to datasets for the templates specified in the **-template** argument.

-template

Specifies the templates to upload. Multiple template are specified in a comma-separated list, for example, **foundation,tcae**.

-h

Displays help for this utility.

ENVIRONMENT

As specified in *Manually configuring your environment for Teamcenter utilities*.

FILES

As specified in *Log files produced by Teamcenter*.

RESTRICTIONS

None.

EXAMPLES

```
manage_icon_files -u=infodba -p=administrator-password -g=dba  
-option=upload -template=bmideproject
```

manage_model_files

Manages upload and download of files in the *TC_DATA_MODEL* directory into **Fnd0BMIDEResource** datasets. These Business Modeler IDE datasets are stored in the database in a **Fnd0BMIDEResource** folder. You can query for this folder in the rich client for quick reference to the Business Modeler IDE resource files that are maintained in the database. You can use these files to restore your Business Modeler IDE environment.

For more information, see the [Business Modeler IDE Guide](#).

SYNTAX

```
manage_model_files -u=user-id {-p=password | -pf=password-file} [-g=group]
-option=[list | upload | download] [-dir=directory -template=template
-release=Teamcenter-release -resource=project ]
-syncToDb
[-h]
```

ARGUMENTS**-u**

Specifies the user ID.

This is generally **infodba** or another user with administration privileges. If this argument is used without a value, the operating system user name is used.

Note

If Security Services single sign-on (SSO) is enabled for your server, the user and password arguments are authenticated externally through SSO rather than being authenticated against the Teamcenter database. If you do not supply these arguments, the utility attempts to join an existing SSO session. If no session is found, you are prompted to enter a user ID and password.

-p

Specifies the user's password.

If used without a value, the system assumes a null value. If this argument is not used, the system assumes the *user-ID* value to be the password.

This argument is mutually exclusive with the **-pf** argument.

-pf

Specifies the password file. If used without a value, the system assumes a null value. If this argument is not used, the system assumes the *user-ID* value to be the password.

For more information about managing password files, see [Manage password files](#).

This argument is mutually exclusive with the **-p** argument.

-g

Specifies the group associated with the user.

If used without a value, the user's default group is assumed.

-option

Valid values for option are:

- **list**

Lists the version of all templates, language, and model files in the utility output by querying the versions from the datasets.

- **upload**

Uploads all template-related files from the *TC_DATA_MODEL* directory to datasets. If **-template** is specified, files specific to the given template are uploaded from the *TC_DATA_MODEL* directory to datasets.

- **download**

Downloads all template-related files from datasets to the *TC_DATA_MODEL* directory. If **-template** is specified, files specific to the given template are downloaded from datasets to the *TC_DATA_MODEL* directory. If **-dir** is specified, files are downloaded to the given directory instead of the *TC_DATA_MODEL* directory.

-dir

Absolute directory path where templates should be downloaded. This can be used only with the **-option=download** option.

-template

List of comma-separated template file names, for example, **foundation,tcae**.

-release

Teamcenter version, for example, **tc9000.1.0**. If the **-release** argument is specified, the template project backups for that release are retrieved. This argument can be used with only the **-resource=project** argument.

-resource=project

Downloads the template project backup to the directory specified by the **-dir** argument. The **-dir**, **-release**, and **-template** arguments must be used in conjunction with the **-resource=project** argument.

-syncToDb

Runs the extractor and uploads the extracted model file into the database. Run the utility with this option only if the data in the database is modified and the model file in the database is not synchronized.

-h

Displays help for this utility.

ENVIRONMENT

As specified in [Manually configuring your environment for Teamcenter utilities](#).

FILES

As specified in [Log files produced by Teamcenter](#).

RESTRICTIONS

None.

EXAMPLES

- To retrieve the latest backup of the **customer** template project to the local **D:\BackupDownloadDir** directory:

```
manage_model_files -u=username -p=password -g=dba
-option=download -template=customer -dir=D:\BackupDownloadDir
-resource=project -release=tc9000.1.0
```

mfk_update

Updates multifield key definitions in the database. *Multifield keys* are IDs assigned to each object to ensure their uniqueness in the database.

Normally, this utility is run automatically by the system when upgrading to update business object instances on the server to the new multifield key definitions. As an administrator, you can also run this utility manually to evaluate proposed multifield key definitions in a template before installing the template to the production server in order to avoid any potential key collisions during installation. You can also use this utility to analyze the multifield key definitions on the server, and if there are corrupt or inconsistent key definitions, rebuild the key table on the database.

Note There are other utilities that you can use to work with keys. Use the [get_key_definition](#) utility to get the multifield key definition for a business object type, and use the [get_key_string](#) utility to get the multifield key value of an item instance as a string containing property names and values.

For more information, see the [Business Modeler IDE Guide](#).

SYNTAX

```
mfk_update -u=user-id {-p=password | -pf=password-file} [-g=group]
-check
-file=template-file-path
-rebuild
[-log=log-file-path] [-h]
```

ARGUMENTS**-u**

Specifies the user ID.

This is generally **infodba** or another user with administration privileges. If this argument is used without a value, the operating system user name is used.

Note If Security Services single sign-on (SSO) is enabled for your server, the **-u** and **-p** arguments are authenticated externally through SSO rather than being authenticated against the Teamcenter database. If you do not supply these arguments, the utility attempts to join an existing SSO session. If no session is found, you are prompted to enter a user ID and password.

-p

Specifies the password.

If used without a value, the system assumes a null value. This argument is mutually exclusive with the **-pf** argument.

If this argument is not used, the system assumes the *user-ID* value to be the password.

-pf

Specifies the password file. The file must be a single-line ASCII file containing the password in clear text. Teamcenter Environment Manager prompts you for a password and creates the password file during installation.

If used without a value, the system assumes a null value. This argument is mutually exclusive with the **-p** argument.

If this argument is not used, the system assumes the *user-ID* value to be the password.

-g

Specifies the group associated with the user.

If used without a value, the user's default group is assumed.

-check

Runs the utility in analysis mode. When analyzing proposed multifield key definitions, the **-file** option must specify the file name.

-file

Specifies the path of the XML file that contains multifield key definitions to be analyzed. If **-file** option is not specified, the utility analyzes the multifield key definitions in the system.

-rebuild

Rebuilds all multifield key entries. This argument is mutually exclusive with the **-check** argument.

-log

Specifies the path of the log file that contains results of the multifield key update. The default is the *TC_TMP_DIR/mfk_updater_date-and-time.log* file. If the *TC_TMP_DIR* environment variable is not set, the log file is created in the system temporary directory (**C:\TEMP** on Windows or **/tmp** on UNIX).

-h

Displays help for this utility.

ENVIRONMENT

As specified in [Manually configuring your environment for Teamcenter utilities](#).

FILES

As specified in [Log files produced by Teamcenter](#).

RESTRICTIONS

None.

EXAMPLES

- To perform an analysis of proposed keys in a custom Business Modeler IDE template:

```
mfk_update -u=infodba -p=infodba -g=dba -check -file=C:\delta.xml
-log=C:\mfk_check_template.log
```

- To perform an analysis of the keys in the database:

```
mfk_update -u=infodba -p=infodba -g=dba -check -log=C:\mfk_check_database.log
```

- To rebuild the key table for all multifield key definitions in the database:

```
mfk_update -u=infodba -p=infodba -g=dba -rebuild -log=C:\mfk_rebuild.log
```

package_live_updates

Packages live update templates stored in the database. Only templates that are enabled for live updates can be packaged using this utility.

For more information, see the [Business Modeler IDE Guide](#).

SYNTAX

```
package_live_updates -u=user-id {-p=password | -pf=password-file} [-g=group]  
-template=template-name  
-dir=directory  
[-log=log-file-path] [-h]
```

ARGUMENTS**-u**

Specifies the user ID.

This is generally **infodba** or another user with administration privileges. If this argument is used without a value, the operating system user name is used.

Note

If Security Services single sign-on (SSO) is enabled for your server, the **-u** and **-p** arguments are authenticated externally through SSO rather than being authenticated against the Teamcenter database. If you do not supply these arguments, the utility attempts to join an existing SSO session. If no session is found, you are prompted to enter a user ID and password.

-p

Specifies the password.

If used without a value, the system assumes a null value. This argument is mutually exclusive with the **-pf** argument.

If this argument is not used, the system assumes the *user-ID* value to be the password.

-pf

Specifies the password file. The file must be a single-line ASCII file containing the password in clear text. Teamcenter Environment Manager prompts you for a password and creates the password file during installation.

If used without a value, the system assumes a null value. This argument is mutually exclusive with the **-p** argument.

If this argument is not used, the system assumes the *user-ID* value to be the password.

-g

Specifies the group associated with the user.

If used without a value, the user's default group is assumed.

-template

Specifies the live update template to package.

-dir

Specifies the full path to the directory where the template package is created. If the **-dir** argument is not provided, the package is created in the

TC_DATA_MODEL directory. The name of the generated package template file is *template-name_template_extracted_from_live_update_site-name*.

-log

Specifies file path and name of the log file that contains the results of this execution. This argument is optional.

-h

Displays help for this utility.

ENVIRONMENT

As specified in *Manually configuring your environment for Teamcenter utilities*.

FILES

As specified in *Log files produced by Teamcenter*.

RESTRICTIONS

None.

process_action_rules

Lists or deletes all the existing action rules in the database.

SYNTAX

process_action_rules [-u=*user-id* {-p=*password* | -pf=*password-file*} -g=*group*]
[-delete | -list] [-h]

ARGUMENTS**-u**

Specifies the user ID.

This is generally **infodba** or another user with administration privileges. If this argument is used without a value, the operating system user name is used.

Note

If Security Services single sign-on (SSO) is enabled for your server, the **-u** and **-p** arguments are authenticated externally through SSO rather than being authenticated against the Teamcenter database. If you do not supply these arguments, the utility attempts to join an existing SSO session. If no session is found, you are prompted to enter a user ID and password.

-p

Specifies the password.

If used without a value, the system assumes a null value. This argument is mutually exclusive with the **-pf** argument.

If this argument is not used, the system assumes the *user-ID* value to be the password.

-pf

Specifies the password file. If used without a value, the system assumes a null value. If this argument is not used, the system assumes the *user-ID* value to be the password.

For more information about managing password files, see [Manage password files](#).

This argument is mutually exclusive with the **-p** argument.

-g

Specifies the group associated with the user.

If used without a value, the user's default group is assumed.

-delete

Deletes all the existing action rules.

-list

Lists the actions rules configured for a site. This is the default mode if delete is not specified.

-h

Displays help for this utility.

ENVIRONMENT

As specified in [Manually configuring your environment for Teamcenter utilities](#).

FILES

As specified in *Log files produced by Teamcenter*.

RESTRICTIONS

None.

EXAMPLES

- To list all action rules in the database:

```
process_action_rules -u=infodba -p=infodba -g=dba -list
```

- To delete all action rules in the database:

```
process_action_rules -u=infodba -p=infodba -g=dba -delete
```

taxonomy

Generates a character-mode summary of the POM class hierarchy. The taxonomy summary is provided in one of two formats: brief summary or full summary. The brief summary provides a single-line description of each POM class; the full summary every attribute of every each POM class.

SYNTAX

taxonomy [-u=*user-id* {-p=*password* | -pf=*password-file*}
-g=*group*] [-b] [-f=*file-name*]

ARGUMENTS**-u**

Specifies the user ID.

This is generally **infodba** or another user with administration privileges. If this argument is used without a value, the operating system user name is used.

Note

If Security Services single sign-on (SSO) is enabled for your server, the **-u** and **-p** arguments are authenticated externally through SSO rather than being authenticated against the Teamcenter database. If you do not supply these arguments, the utility attempts to join an existing SSO session. If no session is found, you are prompted to enter a user ID and password.

-p

Specifies the password.

If used without a value, the system assumes a null value.

If this argument is not used, the system assumes the *user-ID* value to be the password.

This argument is mutually exclusive with the **-pf** argument.

-pf

Specifies the password file. If used without a value, the system assumes a null value. If this argument is not used, the system assumes the *user-ID* value to be the password.

For more information about managing password files, see [Manage password files](#).

This argument is mutually exclusive with the **-p** argument.

-g

Specifies the group associated with the user.

If used without a value, the user's default group is assumed.

-b

Specifies a brief summary. Each line of the summary provides the following information about a POM class: class depth in the schema, object class name, maximum size in bytes, minimum size in bytes, and application name.

-f

Specifies the name of the output file.

ENVIRONMENT

As specified in *Manually configuring your environment for Teamcenter utilities*.

FILES

As specified in *Log files produced by Teamcenter*.

RESTRICTIONS

None.

EXAMPLES

None.

Localization

find_all_key_value_pairs

Finds all the key/value pairs related to a given installation of Teamcenter. The results are generated in separate text files named *language_locale.txt*, where *language_locale* is the Java standardized name for each locale supported by the system. The result files contain the information using the format *key;value*, where *value* can span over more than one line.

SYNTAX

find_all_key_value_pairs [-u=*user-id* {-p=*password* | -pf=*password-file*} -g=*group*] -resources_location=*path* -output_directory=*directory* [-h]

ARGUMENTS**-u**

Specifies the user ID.

This is generally **infodba** or another user with administration privileges. If this argument is used without a value, the operating system user name is used.

Note

If Security Services single sign-on (SSO) is enabled for your server, the **-u** and **-p** arguments are authenticated externally through SSO rather than being authenticated against the Teamcenter database. If you do not supply these arguments, the utility attempts to join an existing SSO session. If no session is found, you are prompted to enter a user ID and password.

-p

Specifies the password.

If used without a value, the system assumes a null value. This argument is mutually exclusive with the **-pf** argument.

If this argument is not used, the system assumes the *user-ID* value to be the password.

-pf

Specifies the password file. If used without a value, the system assumes a null value. If this argument is not used, the system assumes the *user-ID* value to be the password.

For more information about managing password files, see [Manage password files](#).

This argument is mutually exclusive with the **-p** argument.

-g

Specifies the group associated with the user.

If used without a value, the user's default group is assumed.

-resources_location

Specifies the full path to the directory where the Teamcenter localization is located. This value is provided by the **TC_MSG_ROOT** environment variable.

-output_directory

Specifies the directory where the output files are generated.

-h

Displays help for this utility.

ENVIRONMENT

As specified in [Manually configuring your environment for Teamcenter utilities](#).

FILES

As specified in [Log files produced by Teamcenter](#).

RESTRICTIONS

None.

EXAMPLES

None.

ics_localize_class_attributes

Marks class attributes as localizable or nonlocalizable. When marking attributes as nonlocalizable, Teamcenter removes the translations of referenced ICO object properties.

SYNTAX

```
ics_localize_class_attributes [-u=user-id {-p=password |  
-pf=password-file} -g=group]  
-file=file_name -localizable=true | false [-continueOnError] [-h]
```

ARGUMENTS**-u**

Specifies the user ID.

This is generally **infodba** or another user with administration privileges. If this argument is used without a value, the operating system user name is used.

Note

If Security Services single sign-on (SSO) is enabled for your server, the **-u** and **-p** arguments are authenticated externally through SSO rather than being authenticated against the Teamcenter database. If you do not supply these arguments, the utility attempts to join an existing SSO session. If no session is found, you are prompted to enter a user ID and password.

-p

Specifies the password.

If used without a value, the system assumes a null value. This argument is mutually exclusive with the **-pf** argument.

If this argument is not used, the system assumes the *user-ID* value to be the password.

-pf

Specifies the password file. If used without a value, the system assumes a null value. If this argument is not used, the system assumes the *user-ID* value to be the password.

For more information about managing password files, see [Manage password files](#).

This argument is mutually exclusive with the **-p** argument.

-g

Specifies the group associated with the user.

If used without a value, the user's default group is assumed.

-file

Specifies a file name in text format. The input file contains the class ID and the attribute ID of the attributes to be marked as localized or nonlocalized. The format of this text file must be as follows:

```
Class ID | AttributeID_1,AttributeID_2, AttributeID_3...
```

-localizable

If set to **true**, specifies that the class attributes listed in the input file are marked as localizable. If set to **false**, it marks the specified attributes as nonlocalized and removes the translations of referenced ICO object properties.

-continueOnError

Specifies that the utility continues processing after encountering an error.

-h

Displays help for this utility.

ENVIRONMENT

As specified in [Manually configuring your environment for Teamcenter utilities](#).

FILES

As specified in [Log files produced by Teamcenter](#).

RESTRICTIONS

None.

EXAMPLES

```
ics_localize_class_attributes -u=infodba -p=infodba -g=infodba
-file=my_class_attributes.txt -localizable=true -continueOnError
```

This example marks all the attributes found in **my_class_attributes.txt** as localizable. **my_class_attributes.txt** must contain the class and attribute IDs of the attributes to be localized. For example, to mark the **Description** (ID -1200 and **Comments** (ID -1210) attributes contained within the **Components** class (ID **FIXT02**) as localizable, the contents of the **my_class_attributes.txt** file are:

```
FIXT02|-1200,-1210
```

l10n_import_export

Exports property values of objects from the database to an XML file for translation purposes. It is also used to import the translated property values from the XML file to the database.

SYNTAX

```
l10n_import_export [-u=user-id {-p=password | -pf=password-file} -g=group]
[-mode= export | import]
-classificationObjectId=class-ID -transferMode=transfer-mode
-type=type-name -properties=list-of-property-names -file=file-name
-noTransExist -locale=locale-name -master -status=localization-status
[-startDate="DD-MMM-YYYY HH:MM"-endDate="DD-MMM-YYYY HH:MM"]
[-savedQueryName=saved-query-name -entryCount=entry-count -entryNames=
entry-names -entryValues=entry-values]
[-h]
```

ARGUMENTS**-u**

Specifies the user ID.

This is generally **infodba** or another user with administration privileges. If this argument is used without a value, the operating system user name is used.

Note

If Security Services single sign-on (SSO) is enabled for your server, the **-u** and **-p** arguments are authenticated externally through SSO rather than being authenticated against the Teamcenter database. If you do not supply these arguments, the utility attempts to join an existing SSO session. If no session is found, you are prompted to enter a user ID and password.

-p

Specifies the password.

If used without a value, the system assumes a null value. This argument is mutually exclusive with the **-pf** argument.

If this argument is not used, the system assumes the *user-ID* value to be the password.

-pf

Specifies the password file. If used without a value, the system assumes a null value. If this argument is not used, the system assumes the *user-ID* value to be the password.

For more information about managing password files, see [Manage password files](#).

This argument is mutually exclusive with the **-p** argument.

-g

Specifies the group associated with the user.

If used without a value, the user's default group is assumed.

-mode

Specifies whether the user is performing an import or export operation. It takes one of the two values, **-import** or **-export**.

-transfermode

Specifies the name of the transfer mode used to export the objects. This transfer mode specifies the traversal rules, filter rules, and property sets to be used for export. It determines what is exported from the system. If not specified, a default transfer mode is used (**TIEUnconfiguredExportDefault**).

-type

Specifies the type for which the instances have to be exported from the database: **bldb0** (view), **icm0** (classification type), **smlb0** (class), **stxt** (key LOV type), **unct_dict** (dictionary attribute).

-properties

Specifies the list of properties (separated by a comma) for which the values must be exported from the database.

-file

Specifies the file name. It acts as an output XML file while exporting objects from the database and acts as an input XML file while importing objects to the database.

-noTransExist

Specifies that no translations exist. This option should only be used with the **-mode=export** option. This option should be used along with the **-locale** option to export objects for a given type that do not have translations for the specified locale.

-locale

Specifies the locale. When this option is used with the **-mode=export** and **-noTransExist** options, it exports all objects for a given type that have no translations for the specified locale. When this option is used with the **-mode=export** and **-master** options, it exports all objects for a given type that have the specified locale as the master locale.

-master

Exports master values to XML file. This option should only be used with the **-mode=export** option.

-status

Specifies the status of the localization. This option is used with the **-mode=export** option to specify the localization status of the property values that must be exported to the XML file for translations. This option is used with the **-mode=import** option to specify the localization status that needs to be set on the property values that are being imported to the database. This option takes one of the following localization statuses: **A** (approved), **P** (pending), **R** (in review), **I** (invalid), or **M** (master).

-startDate

Specifies the start date in the date range using format *DD-MMM-YYYY HH:MM*, for example, **28-OCT-2010 12:01**.

-endDate

Specifies the end date in the date range using format *DD-MMM-YYYY HH:MM*, for example, **28-OCT-2010 12:01**.

-savedQueryName

Specifies the saved query name that already exists in the database. This option can be used with the **-mode=export** option to export the specified list of localizable properties on objects returned by the saved query.

-entryCount

Specifies the entry count of an input to the saved query.

-entryNames

Specifies the list of entry names separated by commas.

-entryValues

Specifies the list of entry values separated by commas.

-classificationObjectId

Specifies the ID of the classification object that needs to be exported.

-h

Displays help for this utility.

ENVIRONMENT

As specified in [Manually configuring your environment for Teamcenter utilities](#).

FILES

As specified in [Log files produced by Teamcenter](#).

RESTRICTIONS

None.

EXAMPLES

- To search for **Item** objects that have translations so that translations can be done for a new language (for example, Japanese):

```
l10n_import_export -mode=export -type=item -locale=ja_JP -notrans -file=abc
```

- To find all **Item** objects that were created in the month of January 2010:

```
l10n_import_export -mode=export -type=Item -file=abc -startDate=01-01-2010
-endDate=01-31-2010
```

- To import the objects in the Japanese localized file to the database and set the status to pending (**P**):

```
l10n_import_export -mode=import -file=xyz_ja_JP.xml -status=P
```

- To localize the **object_name** property and description for new items, first create a saved query in the rich client to search for the new items (for example, **findNewItem**) and then export the items using the saved query as an option:

```
l10n_import_export -mode=export -savedQueryName=findNewItem -type=Item
-properties=object_name,description -file=abc
```

- To export **Item** objects where the master locale of the **object_name** property is Japanese:

```
l10n_import_export -mode=export -type=Item -properties=object_name
-locale=jp_JP -master -file=abc
```

Only one XML file (for the Japanese locale) is created as **output - abc_jp_JP.xml**. All item instances that have the master locale as Japanese for the **object_name** property are included in this file.

- To export **Item** objects where the localization status of the **object_name** property is pending (**P**):

```
l10n_import_export -mode=export -status=P -type=Item
-properties=object_name -file=abc
```

Only one XML file (for the Japanese locale) is created as output (**abc_jp_JP.xml**). All the item instances that have the master locale as Japanese for the **object_name** property are included in this file.

l10n_purge_translations

Purges the property translations on instances of a given business object. This utility is typically used to clean up the instances of a business object whose property was once marked as localizable with the **Localizable** property constant in the Business Modeler IDE.

For more information, see the [Business Modeler IDE Guide](#).

For example, if you add the **Localization** button to a property by setting the **Localizable** property constant to **true**, and then later decide to remove the button by setting the constant to **false**, you must run the **l10n_purge_translations** utility to remove translations that were entered using the **Localization** button on that property. The utility must be executed after the Business Modeler IDE template is deployed.

The **l10n_purge_translations** utility is necessary only if the **Localizable** constant on a property is moved or deleted from one level, but still exists at another level of the hierarchy. For example, if the **Localizable** property constant is set to **false** on a property on a business object, and there are no sub-business objects that need to be set to **true**, then the utility does not need to be executed. The Business Modeler IDE deploy automatically drops all instances of the **Localization** button translations on the changed property.

SYNTAX

```
l10n_purge_translations [-u=user-id {-p=password | -pf=password-file} -g=group]  
-properties=business-object:property  
-file=file-containing-list-of-properties-to-be-purged  
-purge_lot_size=number  
-log=file-name  
[-h]
```

ARGUMENTS

-u

Specifies the user ID.

This is generally **infodba** or another user with administration privileges. If this argument is used without a value, the operating system user name is used.

Note

If Security Services single sign-on (SSO) is enabled for your server, the user and password arguments are authenticated externally through SSO rather than being authenticated against the Teamcenter database. If you do not supply these arguments, the utility attempts to join an existing SSO session. If no session is found, you are prompted to enter a user ID and password.

-p

Specifies the user's password.

If used without a value, the system assumes a null value. If this argument is not used, the system assumes the *user-ID* value to be the password.

This argument is mutually exclusive with the **-pf** argument.

-pf

Specifies the password file. If used without a value, the system assumes a null value. If this argument is not used, the system assumes the *user-ID* value to be the password.

For more information about managing password files, see [Manage password files](#).

This argument is mutually exclusive with the **-p** argument.

-g

Specifies the group associated with the user.

If used without a value, the user's default group is assumed.

-properties

Specifies properties to be purged. The format of the property is *business-object:property*, for example, **Item:object_name**. Multiple business object/property combinations can be supplied as a comma-separated string.

-file

Specifies the file to read the *business-object:property* to be purged. Each line must contain a business object/property combination.

-purge_lot_size

Specifies the batch size of instances to be loaded for purging. The default value is 10000.

-log

Specifies the file name to report any failures. The default is the **TC_TMP_DIR/10n_purge_translations_date.log**. If the **TC_TMP_DIR** environment variable is not set, the log file is created in the **/tmp** or **%TEMP%** directory.

-h

Displays help for this utility.

ENVIRONMENT

As specified in [Manually configuring your environment for Teamcenter utilities](#).

FILES

As specified in [Log files produced by Teamcenter](#).

RESTRICTIONS

None.

Attribute sharing

tc_config_attr_mapping

Enables an administrative user to create or delete the mapping between Teamcenter properties and external attributes. The utility reads the **attribute_sharing_config.xml** configuration file in the **TC_DATA** directory to create the mapping. If the properties are already mapped to an LOV, the properties are not remapped to the external attributes when the utility is run.

Note As of Teamcenter 9.1, this utility is deprecated and the [bmide_manage_batch_lovs](#) utility should be used instead.

SYNTAX

tc_config_attr_mapping [-u=*user-id* {-p=*password* | -pf=*password-file*} -g=*group*] -action=create_mapping | delete_mapping=*type-name.property-name*] -h

ARGUMENTS**-u**

Specifies the user ID.

This is generally **infodba** or another user with administration privileges. If this argument is used without a value, the operating system user name is used.

Note If Security Services single sign-on (SSO) is enabled for your server, the **-u** and **-p** arguments are authenticated externally through SSO rather than being authenticated against the Teamcenter database. If you do not supply these arguments, the utility attempts to join an existing SSO session. If no session is found, you are prompted to enter a user ID and password.

-p

Specifies the password.

If used without a value, the system assumes a null value. This argument is mutually exclusive with the **-pf** argument.

If this argument is not used, the system assumes the *user-ID* value to be the password.

-pf

Specifies the password file. If used without a value, the system assumes a null value. If this argument is not used, the system assumes the *user-ID* value to be the password.

For more information about managing password files, see [Manage password files](#).

This argument is mutually exclusive with the **-p** argument.

-g

Specifies the group associated with the user.

If used without a value, the user's default group is assumed.

-create_mapping

Reads the configuration file and creates **ExternalAttributes** objects by establishing a connection to the external data source. In addition, the utility creates **MappedProperty** objects using the Teamcenter type property and the external

attributes as well as creating a List of Values (LOV) based on the type of Teamcenter property. The LOV type may be any of the following, based on the value type:

- **ListOfValuesExternalStringExtent**
- **ListOfValuesExternalIntegerExtent**
- **ListOfValuesExternalDoubleExtent**
- **ListOfValuesExternalFloatExtent**
- **ListOfValuesExternalDateExtent**
- **ListOfValuesCharExtent**

The LOV is not populated with any values at this time, the population occurs at runtime when the user requests the values for the LOV. The values are fetched by connecting to the external data source and executing the external query.

-delete_mapping

Deletes the mapping between the specified property and the external attribute. To delete specific mappings, specify the type and property names in the following format:

```
type-name.property-name,type-name.property-name...
```

The attached object mapping and saved query are deleted when the mapping is deleted.

-h

Displays help for this utility.

ENVIRONMENT

As specified in [Manually configuring your environment for Teamcenter utilities](#).

FILES

As specified in [Log files produced by Teamcenter](#).

RESTRICTIONS

The configuration file must be prepared and validated using the [xml_validator](#) utility prior to running the **tc_config_attr_mapping** utility.

EXAMPLES

- To display help for this utility, enter the following command on a single line:

```
tc_config_attr_mapping -h
```

- To create the attribute mapping, enter the following command on a single line:

```
tc_config_attr_mapping -u=infodba -p=password -g=dba
-action=create_mapping
```

- To delete mapping for a specific Teamcenter type property, enter the following command on a single line:

```
tc_config_attr_mapping -u=infodba -p=password -g=dba
-delete_mapping=type1.property1 type2.property2...
```

This command also deletes the associated query. If the LOV that is attached to the property is not referenced by another property in the database, it is deleted when this command is run.

Organization

ldapsync

Compares data in an LDAP directory server with the user data in the Teamcenter database and adds user and person entries that are missing from the Teamcenter database. If the LDAP information is more recent than the Teamcenter information, the tool synchronizes the Teamcenter definitions with the LDAP data. In addition, if user data in Teamcenter does not have a corresponding entry in the LDAP directory, the Teamcenter user is deactivated.

Preferences must be set to enable this feature. For more information, see the [Preferences and Environment Variables Reference](#).

SYNTAX

ldapsync [-u=*user-id* {-p=*password* | -pf=*password-file*}
-g=*group*] -M=*runMode* -l=*LDAP-password* [-t] [-v] [-h]

ARGUMENTS

-u

Specifies the user ID.

This is generally **infodba** or another user with administration privileges. If this argument is used without a value, the operating system user name is used.

Note

If Security Services single sign-on (SSO) is enabled for your server, the **-u** and **-p** arguments are authenticated externally through SSO rather than being authenticated against the Teamcenter database. If you do not supply these arguments, the utility attempts to join an existing SSO session. If no session is found, you are prompted to enter a user ID and password.

-p

Specifies the password.

If used without a value, the system assumes a null value. This argument is mutually exclusive with the **-pf** argument.

If this argument is not used, the system assumes the *user-ID* value to be the password.

-pf

Specifies the password file. If used without a value, the system assumes a null value. If this argument is not used, the system assumes the *user-ID* value to be the password.

For more information about managing password files, see [Manage password files](#).

This argument is mutually exclusive with the **-p** argument.

-g

Specifies the group associated with the user.

If used without a value, the user's default group is assumed.

-M

Specifies the mode in which to run the **sync** tool.

-l

Specifies the LDAP user password. This value overrides the value of the **LDAP_admin_pw** preference.

-t

Specifies trace mode. This value runs the utility in trace mode for debugging or obtaining extra information.

-v

Specifies verbose mode.

-h

Displays help for this utility.

RESTRICTIONS

This utility cannot be used to update replicated organization objects, but can be used to update master organization objects.

EXAMPLES

To update existing user data and add new user entries, enter the following command on a single line:

```
$TC_ROOT/bin/ldapsync -u=administrative-user  
-p=administrative-password -g=dba -l=ldap-password
```

make_user

Creates new users, groups, persons, roles, and volumes outside of a Teamcenter session. This utility also allows you to modify properties of existing user, group, and role objects. The **make_user** utility supports batch mode processing using an input file.

The **make_user** utility creates each of the objects specified by the command line arguments. If the minimum arguments are specified, the utility creates a person and an associated user. If the **-group** argument is supplied, a group is created and the user becomes a member of the group. If the **-role** argument is supplied, a role is created and assigned to that user.

Note If a user is created without specifying a role, the user assumes the default role of the group to which they belong. Because a single user can have multiple roles, it is possible for a user to be a member of the same group multiple times, once for each role. Therefore, although the **make_user** utility does not require that a role be specified, it is recommended that one be specified, particularly if there is more than one role associated with the group.

In addition, if a user is created without specifying a password, Teamcenter assigns the user ID as the password.

More than one user can be created at a time. All of the users created become members of the specified group. If both the **-volume** and **-group** arguments are supplied, the group will have a default volume or be granted access to the volume. If any of the specified objects already exist, this utility does not attempt to create them.

You can modify an existing group, role, person, or user using command line options. Use the **-update** argument to modify the properties of user, person, group, or role objects. Use the **-rename** argument to rename an existing user, person, group, or role. For a user, the **-rename** argument specifies the user ID; for a person, group and role objects, the **-rename** argument specifies the group name and role name, respectively.

Use the **-description** argument to set or modify the description of the group and/or role. When using the **-description** argument with the **-role** and **-group** arguments, the same description is set for both the group and role.

Use the **-volume** argument to create volumes. Before creating volumes, you must have an FMS server cache (FSC) installed and running, and you must set the **FMS_BootStap_Urls** preference with the FSC host and port information.

Use the **-licenselevel** argument to set the license level when creating a user.

Use the **-datasource** argument to fix an object incorrectly configured as internally or externally managed via LDAP synchronization. Use this argument *only* to correct incorrect synchronization settings. Set this argument to **0** reset the object as internally managed. Set this argument to **1** to reset the object as externally managed. Set this argument to **2** to reset the object as remotely managed.

Use the **-citizenships** argument to set the citizenships of the user. You can define multiple nationalities, for example, Great Britain and the United States.

SYNTAX

```

make_user [-u=user-id {-p=password | -pf=password-file} -g=group]
[-update] -user=user-id [-password=password] [-OSuser=name]
-person=name] [-status=0 | 1] [-defaultgroup=default-group]
[-group=group-name] [-parent=parent] [-privilege=0 | 1]
[-description=description] [-security=security] [-defaultrole=default-role]
[-defaultvolume=default-volume] [-defaultlocalvolume=default-local-volume]
[-role=name] [-rename=user-id | group-name | role-name | person-name] [-os]
[-volume=name]
[-node=name] [-path=name] [-file=file]
[-filestoregroupid=filestore-group-ID] [-loadbalancerid=load-balancer-ID]
[-licenselevel= author | consumer | occasionaluser ]1
[-licensebundle=license-bundle]
[-nationality=nationality] [-geography=geography] [-ip_clearance=ip_clearance]
[-gov_clearance=government_clearance]
[-PA1=person-PA1-attribute(address OOTB)]
[-PA2=person-PA2-attribute(city OOTB)]
[-PA3=person-PA3-attribute(state OOTB)]
[-PA4=person-PA4-attribute(zip code OOTB)]
[-PA5=person-PA5-attribute(country OOTB)]
[-PA6=person-PA6-attribute(organization OOTB)]
[-PA7=person-PA7-attribute(GID OOTB)]
[-PA8=person-PA8-attribute(mail code OOTB)]
[-PA9=person-PA9-attribute(e-mail address OOTB)]
[-PA10=person-PA10-attribute(phone number OOTB)]
[-ga=0 | 1 | false | true] [-gm_status=0 | 1 | false | true]
[-citizenships=user_citizenship-list]
[-v] [-datasource=0 | 1 | 2] [-h]

```

ARGUMENTS

-u

Specifies the user ID.

This is generally **infodba** or another user with administration privileges. If this argument is used without a value, the operating system user name is used.

Note

If Security Services single sign-on (SSO) is enabled for your server, the **-u** and **-p** arguments are authenticated externally through SSO rather than being authenticated against the Teamcenter database. If you do not supply these arguments, the utility attempts to join an existing SSO session. If no session is found, you are prompted to enter a user ID and password.

-p

Specifies the password.

If used without a value, the system assumes a null value.

If this argument is not used, the system assumes the *user-ID* value to be the password.

This argument is mutually exclusive with the **-pf** argument.

1. The types of licenses available depends on your license agreement. For descriptions of the available license levels, see your license agreement documentation.

-pf

Specifies the password file. If used without a value, the system assumes a null value. If this argument is not used, the system assumes the *user-ID* value to be the password.

For more information about managing password files, see [Manage password files](#).

This argument is mutually exclusive with the **-p** argument.

-g

Specifies the group associated with the user.

If used without a value, the user's default group is assumed.

-update

Use when modifying any of the existing user, group, or role objects. See restriction #5.

-user

Creates a new Teamcenter user.

-password

Creates a password for the new user.

Note

If a password is not specified for the new user, Teamcenter assigns the user ID as the password.

-OSuser

Specifies the operating system user name for the new user. If this argument is not supplied, the operating system user name defaults to the value specified in the **-user** argument.

-group

Specifies the group to which the new user is added. See restriction #1.

-person

Specifies the person associated with the new user. See restriction #2.

-status

Specifies a user's status. See restriction #8.

-defaultgroup

Specifies a user's default group. See restriction #7.

-parent

Specifies a group's parent group. See restriction #7.

-privilege

Specifies a group's privilege. See restriction #6.

-description

Specifies the description of a group and/or role.

-security

Specifies a group's security.

-defaultrole

Specifies default role for a group. See restriction #7.

-defaultvolume

Specifies default volume for group and/or user. If you specify this argument, you must also specify the **-user**, **-password**, and **-group** arguments. See restriction #7.

-defaultlocalvolume

Specifies the default local volume on the user object. See restriction #9.

-role

Specifies the role to which the new user is assigned.

-rename

Specifies a new name for an existing user, group, role, or person.

-os

Specifies that user names and groups specified in the operating system **/etc/passwd** and **/etc/group** files are used to create new users. See restriction #3.

-volume

Specifies the new volume to be created. See restrictions #4 and #10.

-node

Specifies the network node where the new volume is located. See restriction #4.

-path

Specifies the full path to the location of the new volume. See restriction #4.

-file

Specifies that the input file is read to create users or to modify existing users, groups and roles after other arguments are processed. Each record in the file contains the following information:

```
person|user|password|group|role|option_name1|option_value1
|option_name2|option_value2|...|update
```

option_name is any command line argument and **option_value** is a valid value for **option_name**.

Each field is delimited by the (|) character. The password and role fields can be null (|). The role defaults to the last value specified in either the file or on the command line using the **-role** argument.

Note

If a password is not specified for the new user, Teamcenter assigns the user ID as the password.

When modifying an existing user, group, or role, specify the properties to be modified by *option_name|option_value* pairs followed by the **-update** argument.

-licenselevel

Specifies the license level of the user. The default value is **author**. For information on setting license levels, see the [Organization Guide](#).

The types of licenses available depends on your license agreement. For descriptions of the available license levels, see your license agreement documentation.

-licensebundle

Specifies a license bundle for the user.

Note The following four arguments are required if you want the FMS master configuration updated when a volume is created. Do not include these arguments if you intend to update the FMS configuration manually. If supplied, only one of the **fscid**, **filestoregroupid**, or **loadbalancerid** arguments is permitted.

-fscid

Specifies the FSC ID in the FMS configuration to which the volume element will be added. The maximum length of this argument is 32 characters.

-filestoregroupid

Specifies the Filestore Group ID in the FMS configuration to which the volume element will be added. The maximum length of this argument is 32 characters.

-loadbalancerid

Specifies the load balancer ID in the FMS configuration to which the volume element will be added. The maximum length of this argument is 32 characters.

-nationality

Specifies the nationality of the user.

-citizenships

Specifies the citizenship of the user. Citizenships are a two-letter country code from ISO 3166. A user can have multiple citizenships. Multiple citizenships are specified with a comma delimiter, such as:

```
-citizenships=US,GB
```

-geography

Specifies the geographical location of the user.

-ip_clearance

Specifies the intellectual property (IP) clearance level of the user. This is the level of access the user has to sensitive (classified) information.

-gov_clearance

Specifies the level of clearance the user has to classified data.

-PA1

Specifies the person attribute: address.

-PA2

Specifies the person attribute: city.

-PA3

Specifies the person attribute: state.

-PA4

Specifies the person attribute: zip code.

-PA5

Specifies the person attribute: country.

-PA6

Specifies the person attribute: organization.

-PA7

Specifies the person attribute: GID (group ID).

-PA8

Specifies the person attribute: mail code.

-PA9

Specifies the person attribute: e-mail address.

-PA10

Specifies the person attribute: phone number.

-ga

Specifies the group member admin privilege. If the user is an administrator, set this value to either **1** or **true**. If the user is not an administrator, set this value to either **0** or **false**.

-gm_status

Specifies the group member status. If the group member is inactive, set this value to either **1** or **true**. If the group member is active, set this value to either **0** or **false**.

-v

Runs the utility in verbose mode to display the maximum amount of information. Typically, nonverbose utility sessions only display error messages.

-datasource

Determines whether the synchronization of the specified object (via LDAP synchronization) is reset. Use this argument *only* to correct incorrect synchronization settings.

Set this argument to **0** reset the object as internally managed. Set this argument to **1** to reset the object as externally managed. Set this argument to **2** to reset the object as remotely managed.

-h

Displays help for this utility.

ENVIRONMENT

As specified in [Manually configuring your environment for Teamcenter utilities](#).

FILES

As specified in [Log files produced by Teamcenter](#).

RESTRICTIONS

1. If the command line argument parameter contains spaces, it must be enclosed in quotes, for example, "**product validation**". This restriction does not apply to arguments from a file using the **-file** argument because quotes are automatically added using this argument.
2. The **make_user** utility does not assign person attributes, such as address and phone number.
3. The **-os** argument is only valid on UNIX platforms.
4. The **-volume**, **-node**, and **-path** arguments must not be separated by other arguments.

5. Only one object (user, group, or role) can be updated at a time.

If the **-user**, **-group**, or **-role** arguments are specified when using the **-update** argument, the user object is assumed as the target object of the update.

If the **-group** or **-role** arguments are specified without the **-user** argument, the group object is updated.

If the **-role** argument is specified without the **-user** or **-group** arguments, the role is updated. The object to be updated must already exist.

6. The privilege setting for a group can be either **0** or **1**. **1** implies that the group has DBA privileges. **0** implies a non-DBA group. The default value is **0**.
7. The objects specified for the **-defaultgroup**, **-defaultrole**, **-defaultvolume**, and **-parent** arguments must be existing objects.
8. The valid values for the **-status** argument are **0** (active) and **1** (inactive). If these values are not specified, the default status is active.
9. If both **defaultlocalvolume** and **defaultvolume** arguments are specified, their values must be different. The default local volume must be configured as a valid FMS volume.
10. To create volumes, an FSC must be installed and running, and the **FMS_BootStap_Urls** preference must be set with the FSC host and port information.

RETURN VALUES

Return value upon success 0

Return value upon failure 1

EXAMPLES

- To create three new users (**tom**, **dan**, and **bob**) and assign them to the **london.dev** group, enter the following commands, each on a single line:

```
$TC_ROOT/bin/make_user -u=infodba -p=password -g=dba
-user=tm -group=london.dev -person=tom
$TC_ROOT/bin/make_user -u=infodba -p=password -g=dba
-user=dm -group=london.dev -person=dan
$TC_ROOT/bin/make_user -u=infodba -p=password -g=dba
-user=bp -group=london.dev -person=bob
```
- To assign the role of **planner** to **london.dev** member **tm** and assign the role of **qc** to **london.dev** members **dm** and **bp**, enter the following commands, each on a single line:

```
$TC_ROOT/bin/make_user -u=infodba -p=password -g=dba
-user=tm -group=london.dev -role=planner
$TC_ROOT/bin/make_user -u=infodba -p=password -g=dba
-user=dm -group=london.dev -role=qc
$TC_ROOT/bin/make_user -u=infodba -p=password -g=dba
-user=bp -group=london.dev -role=qc
```
- To add all **london.dev** group members **tm**, **dm**, and **bp** to the **qa** group, enter the following commands, each on a single line:

```
$TC_ROOT/bin/make_user -u=infodba -p=password -g=dba
-user=tm -group=qa
$TC_ROOT/bin/make_user -u=infodba -p=password -g=dba
-user=dm -group=qa
$TC_ROOT/bin/make_user -u=infodba -p=password -g=dba
-user=bp -group=qa
```
- To create a volume test on network node **svr1**, enter the following command on a single line:

```
$TC_ROOT/bin/make_user -u=infodba -p=password -g=dba
-volume=test -node=svr1 -path=/user/volumes/test
```
- To modify the default volume for an existing user (**tm**), enter the following command on a single line:

```
make_user -u=infodba -p=password -g=dba
-update -user=tm -defaultvolume=test1
```
- To create a new group **dev2**, another new group **hongkong.dev2** (a subgroup of the new group **dev2**) and assign to the latter group the default volume test, enter the following command on a single line:

```
$TC_ROOT/bin/make_user -u=infodba -p=password -g=dba
-volume=test -node=svr1 -path=/user/volumes/test -group=hongkong.dev2
```
- To create a new group (**Test**), add a role (**Test Engineer**) to the group and also define a common description for the group and role, enter the following command on a single line:

```
make_user -u=infodba -p=password -g=dba
-group=Test -role="Test Engineer"
-description="Common description for both the Group and Role"
```

- To *rename* a group (**hongkong**), rename a user (**tm**) and modify the user's status to **inactive**, enter the following commands:

```
make_user -u=infodba -p=password -g=dba
          -update -group=hongkong -rename=hk
make_user -u=infodba -p=password -g=dba
          -update -user=tm -rename=tm_new -status=1
```

- To *create* three new users (**tom**, **dan**, and **bob**), whose user IDs are (**tm**, **dm**, and **bp**) and assign them to the **london.dev** group, create a file named **user.lst** containing the following data:

```
tom|tm||london.dev|
dan|dm||london.dev|
bob|bp||london.dev|
```

Then enter the following command on a single line:

```
$TC_ROOT/bin/make_user -u=infodba -p=password -g=dba -file=user.lst
```

- To *rename* the three users (**tom**, **dan**, and **bob**), whose user IDs are (**tm**, **dm**, and **bp**), create a file named **user.lst** containing the following data:

```
|tm||||rename|tm_new|update
|dm||||rename|dm_new|update
|bp||||rename|bp_new|update
```

Then enter the following command on a single line:

```
make_user -u=infodba -p=password -g=dba -file=user.lst
```

- To *inactivate* a user with a user ID of **tom**, create a **user.lst** file containing the following data:

```
|tom||||status|1|update
```

Then enter the following command on a single line:

```
make_user -u=infodba -p=password -g=dba -file=user.lst
```

- To *rename* user **bob**, whose user ID is **bp**, and change the default volume, create a file named **user.lst** containing the following data:

```
|bp||||rename|bp_new|defaultvolume|new_volume|update
```

Then enter the following command on a single line:

```
make_user -u=infodba -p=password -g=dba -file=user.lst
```

- To set the default local volume for an existing user:

```
make_user -u=infodba -p=password -g=dba
          -update -user=mrd -defaultlocalvolume=milfordtempvol
```

- To create a new user (**user_id3**) who is a citizen of the United States:

```
Person_name3|user_id3||PGroup|PRole|citizenships|US
```

- To create a new user (**user_id4**) who is a citizen of both the United States and Japan:

```
Person_name4|user_id4||PGroup|PRole|citizenships|US,JP
```

- To modify the citizenship of an existing (**user_id3**) who is a citizen of both the United States and Great Britain:

```
|user_id3||PGroup|PRole|citizenships|US,GB|update
```
- To modify the citizenship of an existing (**user_id4**) who is a citizen of the United States:

```
|user_id4||PGroup|PRole|citizenships|US|update
```
- To create user **user1** with a license level of **consumer** and user **user2** with a license level of **author**, create a file named **user.lst** containing the following data:

```
user1person|user1|user1|group1|Consumer|licenselevel|consumer
user2person|user2|user2|group1|Author|licenselevel|author
```

Then enter the following command on a single line:

```
make_user -u=infodba -p=password -g=dba -file=user.lst
```

Note The types of licenses available depends on your license agreement. For descriptions of the available license levels, see your license agreement documentation.

- To create user **user1** with a license level of **consumer** and user **user2** with a license level of **author**, and assign **user1** to the **LBundle1** license bundle and **user2** to the **LBundle2** license bundle, create a file named **user.lst** containing the following data:

```
user1person|user1|user1|group1|role1|licenselevel|consumer|licensebundle|LBundle1
user2person|user2|user2|group1|role1|licenselevel|author|licensebundle|LBundle2
```

Then enter the following command on a single line:

```
make_user -u=infodba -p=password -g=dba -file=user.lst
```

Note The types of licenses available depends on your license agreement. For descriptions of the available license levels, see your license agreement documentation.

- To *update* user **user1** with a license level of **consumer** and user **user2** with a license level of **author**, and assign **user1** to the **LBundle1** license bundle and **user2** to the **LBundle2** license bundle, create a file named **user.lst** containing the following data:

```
user1person|user1|user1|group1|role1|licenselevel|consumer|licensebundle|
LBundle1|update
user2person|user2|user2|group1|role1|licenselevel|author|licensebundle|
LBundle2|update
```

Then enter the following command on a single line:

```
make_user -u=infodba -p=password -g=dba -file=user.lst
```

Note The types of licenses available depends on your license agreement. For descriptions of the available license levels, see your license agreement documentation.

Object validation

create_validationdata

Creates validation data or validation agent objects in Teamcenter.

Before creating validation data, ensure the validation agent exists in the database. If the validation agent does not exist, the utility issues an error message and terminates. If the validation data already exists for the given validation agent, the utility issues an error message and terminates.

SYNTAX

The following syntax applies to creating validation data:

```
create_validationdata[-u=user-id {-p=password | -pf=password-file} -g=group]
  -v_name=validation-name
  -v_agent_name=agent-name
  [-clr_name=closure-rule]
  [-v_desc=validation-data-description]
  [-v_category=category]
  [-v_ext_rule_file=external-rule-file-path]
  [-log]
  [-h]
```

The following syntax applies to creating validation agent:

```
create_validationdata[-u=user-id {-p=password | -pf=password-file} -g=group]
  -create_agent
  -v_agent_name=agent-name
  -v_util_cmd=utility-command
  -clr_name=closure-rule
  [-v_desc=validation-agent-description]
  [-v_args=validation-arguments]
  [-log]
  [-h]
```

ARGUMENTS

-u

Specifies the user ID.

This is generally **infodba** or another user with administration privileges. If this argument is used without a value, the operating system user name is used.

Note If Security Services single sign-on (SSO) is enabled for your server, the **-u** and **-p** arguments are authenticated externally through SSO rather than being authenticated against the Teamcenter database. If you do not supply these arguments, the utility attempts to join an existing SSO session. If no session is found, you are prompted to enter a user ID and password.

-p

Specifies the password.

If used without a value, the system assumes a null value.

If this argument is not used, the system assumes the *user-ID* value to be the password.

This argument is mutually exclusive with the **-pf** argument.

-pf

Specifies the password file. If used without a value, the system assumes a null value. If this argument is not used, the system assumes the *user-ID* value to be the password.

For more information about managing password files, see [Manage password files](#).

This argument is mutually exclusive with the **-p** argument.

-g

Specifies the group associated with the user.

If used without a value, the user's default group is assumed.

-create_agent

Indicates the utility is to create a validation agent.

-v_name

Specifies the name of the validation data, for example, **mgc_exam_geometry_combo** in **NX Check-Mate**.

-v_agent_name

Specifies the name of the validation agent in the database.

-v_desc

Describes the validation data or validation agent, for example, **"Test geometry"**.

-v_category

Specifies the category of the validation data, for example, **template.modeling**.

-v_util_cmd

Specifies the utility command string for the validation agent, for example, **ug_check_part**.

-clr_name

Specifies the name of the closure rule. Closure rule names must be unique. If multiple closure rules are encountered, the utility considers only the first rule. If the closure rule name does not exist in the database, the utility issues an error message and returns a list of closure rules in the database available to the user. An example of a closure rule is **NX**.

-v_args

Specifies the arguments associated with the validation agent, for example, **-pim=yes**.

-v_ext_rule_file

Specifies the external rule file path for validation data, for example, **D:\Temp\ExtRuleFile.xls**.

-log

Specifies the output of this utility to be written to the log file.

-h

Displays help for this utility.

ENVIRONMENT

Requires no special environment. Teamcenter environment is sufficient to run this utility.

RESTRICTIONS

None.

EXAMPLES

The following example creates a validation agent for the **NX Check-Mate** utility for the **mqc_exam_geometry** checker.

```
create_validationdata.exe -create_agent -v_agent_name= "NX CheckMate"  
-v_desc= "testing create agent" -v_util_cmd= "ug_check_part"  
-clr_name= "NX" -v_args= "-pim=yes"  
  
create_validationdata.exe -v_name= "mqc_exam_geometry"  
-v_agent_name= "NX CheckMate" -v_desc= "testing create checker"  
-v_category= "modeling"
```

Teamcenter reporting

install_default_report_designs

Creates the default report designs contained in the **default_report_design.xml** file as part of the Teamcenter installation process. Other report designs can be imported into the database by adding them to the **default_report_design.xml** file and rerunning the utility.

SYNTAX

```
install_default_report_designs [-u=user-id {-p=password |  
-pf=password-file} -g=group]  
-file=xml-file-list -design_name=report-design-name -create_new [-update_all |  
-update_query | -update_pff | -update_formatter [-v] [-h]
```

ARGUMENTS**-u**

Specifies the user ID.

This is generally **infodba** or another user with administration privileges. If this argument is used without a value, the operating system user name is used.

Note

If Security Services single sign-on (SSO) is enabled for your server, the **-u** and **-p** arguments are authenticated externally through SSO rather than being authenticated against the Teamcenter database. If you do not supply these arguments, the utility attempts to join an existing SSO session. If no session is found, you are prompted to enter a user ID and password.

-p

Specifies the password.

If used without a value, the system assumes a null value. This argument is mutually exclusive with the **-pf** argument.

If this argument is not used, the system assumes the *user-ID* value to be the password.

-pf

Specifies the password file. If used without a value, the system assumes a null value. If this argument is not used, the system assumes the *user-ID* value to be the password.

For more information about managing password files, see [Manage password files](#).

This argument is mutually exclusive with the **-p** argument.

-g

Specifies the group associated with the user.

If used without a value, the user's default group is assumed.

-file

Specifies the name of the XML file containing report design definitions.

-design_name

Specifies the name of the report design.

-create_new

Creates new objects.

This cannot be used with update modes.

-update_all

Updates all existing objects (query, pff, and formatter).

-update_query

Updates existing query objects.

-update_pff

Updates existing pff objects.

-update_formatter

Updates existing formatter objects.

-v

Displays detailed status information.

-h

Displays help for this utility.

ENVIRONMENT

As specified in [Manually configuring your environment for Teamcenter utilities](#).

FILES

As specified in [Log files produced by Teamcenter](#).

RESTRICTIONS

None.

**RETURN
VALUES**

Return value 0
upon success

Return value >1, -1
upon failure

EXAMPLES

To create default report enter the following command on a single line:

```
install_default_report_designs -u=infodba -p=infodba -g=dba
-file=$TC_DATA\report_writer\default_report_designs.xml
```

**XML FILE
FORMAT**

The format required for the XML file is as follows.

```
<ReportDesign>
  <DesignName>Admin - Objects By Status</DesignName>
  <DesignDesc>This report returns objects of a specified type released to a
specified status.</DesignDesc>
  <Query>
    <QueryName>Admin - Objects By Status</QueryName>
    <QueryDesc>This query was created to support the Admin -
Objects By Status Report.</QueryDesc>
    <QueryClass>WorkspaceObject</QueryClass>
    <QueryClause>SELECT qid FROM WorkspaceObject WHERE
      "object_type" = "${Type = ItemRevision}" AND
      "release_status_list.name" = "${Release Status = }"
    </QueryClause>
    <DomainFlag>QRY_DOMAIN_LOCAL</DomainFlag>
  </Query>
  <Pff>
    <PffName>Admin - Objects By Status</PffName>
    <PffDesc>This PFF was created to support the Admin -
Objects By Status Report.</PffDesc>
    <PffClass>WorkspaceObject</PffClass>
    <PffClause>
      WorkspaceObject.object_name;Object Name,
      WorkspaceObject.object_type;Object Type,
      WorkspaceObject.release_status_list.name;Release Status,
      WorkspaceObject.date_released;Date Released,
      WorkspaceObject.owning_user.user_id;User Name,
      WorkspaceObject.owning_group.name;Group Name
    </PffClause>
  </Pff>
  <Formatter>
    <Filename>default_xml_template.xml</Filename>
    <Datasettype>XMLReportFormatter</Datasettype>
  </Formatter>
  <Formatter>
    <Filename>default_excel_template.xlt</Filename>
    <Datasettype>ExcelReportFormatter</Datasettype>
  </Formatter>
</ReportDesign>
```

XML file format

rep_batch_report

Used in batch or shell script files to generate reports in batch mode when the user selects the **Batch mode** option in the **Generate ME Report** dialog box.

The following administrative tasks must be performed to enable batch reporting:

1. Create a report request flat file containing default values and details of item ID, revision ID, operating system report location, Teamcenter report location, report format, revision rule, variant rule, transfer mode, and status. The format of the file is as follows:

```

_____Start of format_____
start_global_definitions
User_ID    infodba
Password   infodba
TC_ROOT    w:\iman_wnti
TC_DATA    w:\src\iman\data
Default Report Location Teamcenter    # possible values are OS or Teamcenter
Default OS Location    C:\temp\Reports
Default Revision Rule   Latest Working
Default Saved Variant Rule DemoVariantRule
Default Transfer Mode   web_reports
Default Formatter      : Product_Structure.xml
Delimiter            ~
end_global_definitions
start_data_definition
#ItemID~Revision~Report Location~OS Location~Revision Rule~Saved Variant
Rule~Transfer
Mode~Formatter~Status
000234~A~OS~~~MyVariantRule~~Standard Product~
end_data_definition

start_data_definition
#ItemID~Revision~Report Location~OS Location~Revision Rule~Saved Variant
Rule~Transfer Mode~Formatter~Root Product Tag~ (line continued)
Root Product Rev rule~Root Product Var rule~Root process Tag ~Hierarchy
UID tags~level~Root PlantTag~Root Plt Rev Rule~Root Plt Var Rule~Status
GM00071~001~OS~~Latest Working~~web_reports~Product_Structure.xml~~~~~
ABC000007~A~OS~~Latest Working~~web_reports~station_weld.xml~SvNJ1puVl9P7VD~
(line continued)
Latest Working~~htNJmI1819P7VD~x5FJmI1819P7VD,BKKJmI1819P7VD,RaDJmI1819P7VD
~3~zwDJ2_1$19P7VD~Latest Working~~
end_data_definition
_____End of format_____

```

Note

A sample batch file and shell script file are located in the **TC_ROOTweb/htdocs/web_reports/data** directory. In addition, you can manually append data to an existing **batch_request** file and run the utility.

2. Schedule a task in the operating system.
3. Select the program and specify the execution date and time.

The utility performs the following activities:

1. Reads the location of the report request flat file from the value of the **Batch_Report_Request_File** preference.

For more information, see the [Preferences and Environment Variables Reference](#).

2. Reads the global definition values.
3. Parses each line in the flat file, checks the status field of the line, and processes the line if the status is not **success**.
4. An XML file is generated for each line in the file, using the revision rule, variant rule, and closure rule associated with the transfer mode. If the rules are not specified in the line, default rules are used.
5. The report format (style sheet) is applied on the XML file and report HTML files are generated. The datasets are exported during the transformation.
6. If the report location is specified as Teamcenter, the dataset is created and the files generated are attached to the item revision, including the exported dataset files.
7. If the report location is specified as OS, the reports are saved at the OS location specified in the file.
8. Create or update the log file for the process.
9. Update the status field once the line is processed.

SYNTAX

rep_batch_report [-u=*user-id* {-p=*password* | -pf=*password-file*} -g=*group*] -h

ARGUMENTS**-u**

Specifies the user ID.

This is generally **infodba** or another user with administration privileges. If this argument is used without a value, the operating system user name is used.

Note

If Security Services single sign-on (SSO) is enabled for your server, the **-u** and **-p** arguments are authenticated externally through SSO rather than being authenticated against the Teamcenter database. If you do not supply these arguments, the utility attempts to join an existing SSO session. If no session is found, you are prompted to enter a user ID and password.

-p

Specifies the password.

If used without a value, the system assumes a null value. This argument is mutually exclusive with the **-pf** argument.

If this argument is not used, the system assumes the *user-ID* value to be the password.

-pf

Specifies the password file. If used without a value, the system assumes a null value. If this argument is not used, the system assumes the *user-ID* value to be the password.

For more information about managing password files, see [Manage password files](#).

This argument is mutually exclusive with the **-p** argument.

-g

Specifies the group associated with the user.

If used without a value, the user's default group is assumed.

-h

Displays help for this utility.

ENVIRONMENT

As specified in *Manually configuring your environment for Teamcenter utilities*.

FILES

As specified in *Log files produced by Teamcenter*.

RESTRICTIONS

None.

import_export_reports

Allows report definitions, their dependent data (for example, saved query definitions and property set definitions), and their associated style sheets to be exported from one Teamcenter server and imported to another Teamcenter server.

SYNTAX

```
import_export_reports {-import | -export | -execute}  
[-u=user-id -p=password | -pf=password-file -g=group]  
-stageDir=directory -reportId=report-identifier  
[-reportFile=xml-output-file] [-f=xml-file] [-h]
```

ARGUMENTS

-import

Specifies the report definitions are to be imported to the Teamcenter server.

-export

Specifies the report definitions are to be exported from the Teamcenter server.

-execute

Generates a report in the command line. This argument requires the **-f** argument.

-u

Specifies the user ID.

This is generally **infodba** or another user with administration privileges. If this argument is used without a value, the operating system user name is used.

Note

If Security Services single sign-on (SSO) is enabled for your server, the **-u** and **-p** arguments are authenticated externally through SSO rather than being authenticated against the Teamcenter database. If you do not supply these arguments, the utility attempts to join an existing SSO session. If no session is found, you are prompted to enter a user ID and password.

-p

Specifies the password.

If used without a value, the system assumes a null value.

If this argument is not used, the system assumes the *user-ID* value to be the password.

This argument is mutually exclusive with the **-pf** argument.

-pf

Specifies the password file. If used without a value, the system assumes a null value. If this argument is not used, the system assumes the *user-ID* value to be the password.

For more information about managing password files, see [Manage password files](#).

This argument is mutually exclusive with the **-p** argument.

-g

Specifies the group associated with the user.

If used without a value, the user's default group is assumed.

-stageDir

Specifies the fully qualified name of the directory that contains all of the report definitions and its associated data in predefined format.

-reportId

Specifies the ID of the report definition.

-reportFile

Specifies the name of the XML containing the list of report templates.

-f

Specifies the name of an XML file containing report parameters. This argument is used with the **-execute** arguments.

-h

Displays help for this utility.

ENVIRONMENT

As specified in [Manually configuring your environment for Teamcenter utilities](#).

FILES

As specified in [Log files produced by Teamcenter](#).

RESTRICTIONS

None.

EXAMPLES

- The following command exports report definitions and associated data (style sheets) to the file system pointed by the **stageDir** argument:

```
import_export_reports -export -u=<username> -p=<password> -g=<group>
    -stageDir=<data directory> -reportId=<reportname>
```

- The following command imports report definitions from a Teamcenter server:

```
import_export_reports -import -u=<username> -p=<password> -g=<group>
    -stageDir=<data directory> -reportId=<reports>
```

Teamcenter interface

AppRegUtil

Communicates with the application registry, either as a stand-alone program or within an application (such as an installation program), to:

- Check the existence or availability of an application registry.
- Register an application instance with the application registry.
- Unregister an application instance from the application registry.

SYNTAX

```
AppRegUtil -mode={verify | register | unregister | list | help}  
{-import | -export | -execute}  
[-u=user-id -p=password | -pf=password-file -g=group]  
[-file=default | data-file] [-appRegUrl=application-registry-URL]  
[-guid=guid-of-chooser-application-instance] [-file=file-name] [-h]
```

ARGUMENTS**-p**

Specifies the password.

If used without a value, the system assumes a null value.

If this argument is not used, the system assumes the *user-ID* value to be the password.

This argument is mutually exclusive with the **-pf** argument.

-pf

Specifies the password file. If used without a value, the system assumes a null value. If this argument is not used, the system assumes the *user-ID* value to be the password.

For more information about managing password files, see [Manage password files](#).

This argument is mutually exclusive with the **-p** argument.

-g

Specifies the group associated with the user.

If used without a value, the user's default group is assumed.

-mode

Specifies the task to perform. The value can be **verify**, **register**, **unregister**, **list**, or **help**. If the mode is not specified, the program exits with an error.

verify

Checks whether the given application registry URL is running and also provides the registry information file for the URL.

register

Registers an application with the application registry with the details from the specified registry file, based on the data in the file specified by the **-file** argument. If the **appRegUrl** argument is used with this argument, it overrides the URL specified in this file.

unregister

Unregisters the application identified by the **guid** argument from the application registry identified with the **appRegUrl** argument. It also provides the registry file for this information.

list

Lists all entries in the application registry.

help

Displays help for this argument.

-file

Specifies the file name of the configuration file containing the application details and the application registry URL. Examples of application details are **AppGUID**, **launcher URL**, **portal launcher URL**, and **webService URL**.

The file format must be *Name=Value* pairs, separated by the equal sign (=). For formatting examples, see the **AppRegUtil.data.default** file in the **\$TC_DATA** directory.

If you specify **default**, rather than a file name, the utility uses the **AppRegUtil.data.default** file in the **\$TC_DATA** directory. This is a template file you can either edit, or save and modify.

-appRegUrl

Specifies the URL of the application registry. The value can be passed as an argument or as a property in the configuration file provided as an argument. For more information, see the description of the **-file** argument.

-guid

Specifies the **guid** of the Teamcenter application instance.

-h

Displays help for this utility.

ENVIRONMENT

If the **-file** option is used with a value of **default**, the **TC_DATA** environment must be available.

RESTRICTIONS

The file format of the configuration file specified by the **-file** argument must be *Name=Value* pairs, separated by the equal sign (=). For more information about formatting this file, see the **AppRegUtil.data.default** file in the **\$TC_DATA** directory. This is a template file you can either edit, or save and modify.

EXAMPLES

The following examples illustrate use of the **AppRegUtil** utility.

Verifying the existence of the Application Registry

- To verify the existence of the application registry:

```
AppRegUtil -mode=verify -appRegUrl=application-registry-URL
```

Tests whether the given application registry is running. Returns **true** if the application registry is not running.

- To test whether the application registry identified by the **ApplicationRegistryUrl** property in the **config** file is active:

```
AppRegUtil -mode=verify -file=absolute-path-of-config-file
```

Returns a failure if the application registry is not active or is unreachable.

- To test whether the application registry identified by the **ApplicationRegistryUrl** property in the default **config** file in the **\$TC_DATA/AppRegUtil.data** directory is active:

```
AppRegUtil -mode=verify -file=
```

Returns a failure if the application registry is not active or is unreachable.

Registering a Teamcenter application instance with the Application Registry

- To register an application instance defined in the given **config** file:

```
AppRegUtil -mode=register -file=absolute-path-of-config-file
```

or

```
AppRegUtil -mode=register -file=default
```

The data in the file must be *Name=Value* pairs. If the value of the **-file** option is defined as **default**, the utility uses the **AppRegUtil.data.default** file in the **\$TC_DATA** directory. This is a template file you can either edit, or save and modify.

The configuration contains the **ApplicationRegistryURL** property, which provides the application registry information. The value of this property can be overridden using the **-appRegUrl=application-registry-URL** on the command line.

Unregistering a Teamcenter application instance with the Application Registry

- To unregister an application instance defined in the given **config** file:

```
AppRegUtil -mode=unregister -file=absolute-path-of-config-file
```

or

```
AppRegUtil -mode=unregister -file=default
```

The data in the file must be *Name=Value* pairs. If the value of the **-file** option is defined as **default**, the utility uses the **AppRegUtil.data.default** file in the **\$TC_DATA** directory. This is a template file you can either edit, or save and modify.

The configuration contains the **ApplicationRegistryURL** property, which provides the application registry information. The value of this property can be overridden using the **-appRegUrl=application-registry-URL** on the command line.

Chapter

3 *Product configuration utilities*

Content management

contmgmt_upgrade_8x

Adds a prefix to DITA object item IDs and updates all content references with the correct extensions. This utility must be run after upgrading from a Teamcenter release prior to 9.0 and after performing the other migration tasks.

For more information, see the [Content Management Guide](#).

SYNTAX

contmgmt_upgrade_8x [-u=*user-id* -p=*password* | -pf=*password-file* -g=*group*]
[-prefix=*DITA-prefix*]
[-dryrun] [-h]

ARGUMENTS**-u**

Specifies the user ID.

This is generally **infodba** or another user with administration privileges.

-p

Specifies the password.

-pf

Specifies the password file. If used without a value, the system assumes a null value. If this argument is not used, the system assumes the *user-ID* value to be the password.

For more information about managing password files, see [Manage password files](#).

This argument is mutually exclusive with the **-p** argument.

-g

Specifies the group associated with the user.

-prefix

The prefix that will be added to all DITA objects during upgrade if they do not already have this prefix. This prefix should begin with an alphabetic character.

-dryrun

Generates a report of the changes that will be made but does not update the database. Places a text file that contains the old and new **item_id** values and the old and new XML files that have updated content references in the Teamcenter temporary directory.

-h

Displays help for this utility.

ENVIRONMENT

As specified in [Manually configuring your environment for Teamcenter utilities](#).

FILES

As specified in [Log files produced by Teamcenter](#).

RESTRICTIONS

None.

EXAMPLES

- To run a test to see which DITA objects will have their **item_id** values changed and which content references will be changed:

```
contmgmt_upgrade_8x -u=infodba -p=pswd -g=dba -prefix=X -dryrun
```

- To change all the **item_id** values of DITA topics to begin with the prefix and to update content references:

```
contmgmt_upgrade_8x -u=infodba -p=pswd -g=dba -prefix=X
```

contmgmt_migration_100

In Teamcenter 10 and later, a new relationship exists between the **PSOccurrence** object between two related topics and the reference topic type that is between the two topic types. When upgrading from a prior release, the **contmgmt_migration_100** utility must be run to add this relationship to **PSOccurrence** objects. If more than one reference type exists between the two topic types, the utility selects one and the selection is noted in a log file.

For more information, see the [Content Management Guide](#).

SYNTAX

contmgmt_migration_100 [-u=*user-id* -p=*password* | -pf=*password-file* -g=*group*]
[-clear]
[-dryrun] [-h]

ARGUMENTS**-u**

Specifies the user ID.

This is generally **infodba** or another user with administration privileges.

-p

Specifies the password.

-pf

Specifies the password file. If used without a value, the system assumes a null value. If this argument is not used, the system assumes the *user-ID* value to be the password.

For more information about managing password files, see [Manage password files](#).

This argument is mutually exclusive with the **-p** argument.

-g

Specifies the group associated with the user.

-clear

Clears any existing **Ctm0RefTopicTypeR** relations before processing.

-dryrun

Generates a report of the changes to be made but does not update the database.

-h

Displays help for this utility.

ENVIRONMENT

As specified in [Manually configuring your environment for Teamcenter utilities](#).

FILES

As specified in [Log files produced by Teamcenter](#).

RESTRICTIONS

None.

EXAMPLES

- To run a test to see which **PSOccurrences** objects will have relationships added:

```
contmgmt_migration_100 -u=infodba -p=pswd -g=dba -dryrun
```

- To update all the **PSOccurrences** objects with the new relationship:

```
contmgmt_migration_100 -u=infodba -p=pswd -g=dba
```

Product structure maintenance

bom_expand

DESCRIPTION

Expands the bill of materials (BOM) and generates a report on the BOM structure and expansion statistics.

You can use this utility to find the total size and depth of a BOM structure as well as the number of lines at each level of the structure.

This utility provides a **bomset** mode that can help you expand the BOM in equal-sized sets without encountering memory issues.

SYNTAX

```
bom_expand [-u=user-id -p=password | -pf=password-file -g=group]  
-item=item | -key=item-key  
[-revision_rule=revision-rule] [-rev=revision]  
[-saved_variant_rule=saved-variant-rule]  
[-use_packing] [-show_unconfigured] [-show_unconfigured_variants]  
[-props=comma-separated-list-of-property-names] [-level_props]  
[-output=output-filename] [-props= | -props_output=  
comma-separated-list-of-property-names]  
[-bom_report_file=report-file-name] [-report_mode= [TOP | LEVEL | FULL]]  
[-no_verbose] [-mem_debug] [-bom_closure_rule=closure-rule]  
[-bomset] [-bomset_criteria=[MEMORY | NUMLINES | PERCENT]]  
[-bomset_help]  
[-bomset_criteria_data=[Number-of-Lines | PERCENT]]  
[-bomset_override_data_complexity_preference=[true | false]]  
[-bomset_use_mem_percentage_preference=percent]  
[-create_absoccs]  
[-create_absocc_ids] [-h]
```

ARGUMENTS**-u**

Specifies the user ID.

This is generally **infodba** or another user with or without administration privileges. If this argument is used without a value, the operating system user name is used.

Note

If Security Services single sign-on (SSO) is enabled for your server, the **-u** and **-p** arguments are authenticated externally through SSO rather than being authenticated against the Teamcenter database. If you do not supply these arguments, the utility attempts to join an existing SSO session. If no session is found, you are prompted to enter a user ID and password.

-p

Specifies the password.

If used without a value, the system assumes a null value. This argument is mutually exclusive with the **-pf** argument.

If this argument is not used, the system assumes the *user-ID* value to be the password.

-pf

Specifies the password file. If used without a value, the system assumes a null value. If this argument is not used, the system assumes the *user-ID* value to be the password.

For more information about managing password files, see [Manage password files](#).

This argument is mutually exclusive with the **-p** argument.

-g

Specifies the group associated with the user.

If used without a value, the user's default group is assumed.

-item

Specifies the item ID for which the associated BOM view revision (BVR) is traversed. (BVRs are associated with revisions corresponding to the specified item.)

-key

Specifies the key for which the associated BVR is traversed. (BVRs are associated with revisions corresponding to the specified item.)

[keyAttr1=keyVal1] [,keyAttr2=keyVal2]...[,keyAttrN=keyValN]

To find the key of an object, use the [get_key_string](#) utility.

For more information, see the [Business Modeler IDE Guide](#).

-revision_rule

Specifies the revision rule to use when configuring the BOM window.

If this argument is omitted, uses the default revision rule, **Latest Working**.

-rev

Specifies the item revision specified by the **-item** argument. The revision must have an associated BVR.

-saved_variant_rule

Specifies the variant rule to apply to the BOM window when configuring the BOM.

-use_packing

Indicates whether the packed lines in the BOM window should be packed or unpacked.

-show_unconfigured

Indicates whether unconfigured BOM lines are considered for BOM expansion.

-show_unconfigured_variants

Indicates whether to consider unconfigured BOM lines due to variant conditions for BOM expansion.

-props

Specifies the set of properties to fetch for BOM structure lines. If the value specified is **rich client** (RAC), fetches all properties required by the rich client. You can also specify a comma-separated list of properties. For example, if you want properties **bl_formula** and **bl_variant_state**, specify this argument as **-props= bl_formula, bl_variant_state**.

-level_props

Indicates whether to fetch all properties together for a BOM level during expansion. If this flag is absent, fetches properties once for the entire BOM after expansion.

-output

Directs the output to a specific file by specifying a file name.

-props_output

Lists BOM line properties separated by a comma that should be written to output. The **-props** and **-props_output** arguments are mutually exclusive.

-bom_report_file

Specifies the report file name, for example, **report.csv** (CSV file type is preferred).

-report_mode

Specifies report generation mode. Valid values are:

- **TOP** prints expansion report for top node of BOM only.
- **LEVEL** prints summary of expansion for all levels in BOM.
- **FULL** prints detailed expansion report for each node in BOM.

The default mode is **LEVEL**.

-no_verbose

Indicates whether to write detailed output to the console.

-mem_debug

Indicates whether to print additional debug information related to memory usage during BOM expansion.

-bom_closure_rule

Specifies BOM closure rule to apply to BOM window during BOM expansion.

-bomset

Indicates whether expansion should be done in equal-sized sets of BOM lines.

-bomset_criteria

Specifies creation criteria of BOM sets for expansion. Valid values are:

- **MEMORY**
The BOM is divided and expanded in sets that occupy the same memory. This memory is calculated based on the available system physical memory and BOM size.
- **NUMLINES**
The BOM is divided and expanded in sets that contain the same number of BOM lines. Use the argument **-bomset_criteria_data** to specify the number of lines for each BOM set.
- **PERCENT**
The BOM is divided and expanded in sets that contain the number of lines which are the same percent as the total number of BOM lines in the BOM. Use the argument **-bomset_criteria_display** to specify this percent.

The default is **MEMORY**.

-bomset_help

Prints examples for using **bomset** options.

-bomset_criteria_data

Specifies data for BOM set creation to use with **-bomset_criteria**. Valid values are **NUMLINES** or **PERCENT**.

This argument should be used when **-bomset_criteria_data=NUMLINES** or **-bomset_criteria_data=PERCENT**.

For example, if you have a 10,000 item BOM and you want it to be divided and expanded in sets each having the size 1,000 items, include the following arguments in your command:

- **-bomset -bomset_criteria=NUMLINES**
- **-bomset_criteria_data=1000**

If you want a 10,000 item BOM to be divided and expanded in sets that are 10 percent of its size, include the following arguments in your command:

- **-bomset -bomset_criteria=PERCENT**
- **-bomset_criteria_data=10**

-bomset_override_data_complexity_preference

Specifies whether BOM lines have complex data, such as incremental changes and absolute occurrences, which can decide the size of the BOM set when the **-bomset_criteria** value is set to **MEMORY**. The valid values are **true** or **false**. The default value is **false**.

-bomset_use_mem_percentage_preference

Specifies the percent of system memory available to use for expansion. Valid values must be set between **0** and **100**.

-create_absoccs

Creates absolute occurrence appearance path node (APN) objects on a BOM structure during expansion.

-create_absocc_ids

Creates or uses existing absolute occurrence IDs on a BOM structure during expansion.

-h

Displays help for this utility.

ENVIRONMENT

As specified in [Manually configuring your environment for Teamcenter utilities](#).

FILES

As specified in [Log files produced by Teamcenter](#).

This utility can generate different output files based on the **[-output=output-filename]** or the **-bom_report_file** options.

RESTRICTIONS

This is a debug utility and should only be used for debugging purposes.

EXAMPLES

- BOM expansion with properties, using revision rules and variant rules.

```
bom_expand -u=user-id -p=password -pf=password-file -item=item
-rev=revision -revision_rule=revision-rule -saved_variant_rule=
saved-variant-rule -bom_closure_rule=closure-rule -props=RAC-level_props
```

- BOM expansion with out properties, using revision rule and variant rule.

```
bom_expand
-u=user-id -p=password -pf=password-file -item=item -rev=revision
-revision_rule=revision-rule -saved_variant_rule=
saved-variant-rule
```

- BOM expansion with properties, using revision rule and variant rule and closure rule.

```
bom_expand -u=user-id -p=password -pf=password-file -item=item
-rev=revision -revision_rule=revision-rule -saved_variant_rule=
saved-variant-rule -props=RAC-level_props
bom_report_file=report-file-name -report_mode=FULL
```

- BOM expansion with properties, using revision rule and variant rule and providing output in a file (txt).

```
bom_expand -u=user-id -p=password -pf=password-file -item=item
-rev=revision -revision_rule=revision-rule -saved_variant_rule=
saved-variant-rule -props=RAC-level_props -output=output-filename
```

- BOM expansion with properties, using revision rule and variant rule and closure rule with properties, using revision rule and variant rule and providing BOM report file (csv) (BOM data).

```
bom_expand -u=user-id -p=password -pf=password-file -item=item
-rev=revision -revision_rule=revision-rule -saved_variant_rule=
saved-variant-rule -props=RAC-level_props
bom_report_file=report-file-name -report_mode=FULL
```

- BOM expand with BOM set.

```
bom_expand -u=user-id -p=password -item=item_id -bomset
bom_expand -u=user-id -p=password -item=item_id -bomset
-bomset_criteria=PERCENT -bomset_criteria_data=% of bomlines in a set.
Value between 0-100>
bom_expand -u=user-id -p=password -item=item_id -bomset
-bomset_criteria=NUMLINES -bomset_criteria_data=num of lines in a set
bom_expand -u=user-id -p=password -item=item_id -bomset
-bomset_override_data_complexity_preference=true/false
-bomset_use_mem_percentage_preference=% of system mem
which can be used for bomset/expansion. Value between 0 -100.
Only valid when -bomset_criteria=Empty value of MEMORY
```

bom_roll_up_report

Creates BOM properties rollup reports. The reports can be created systematically by a task scheduler to generate weekly or daily reports to track property changes of assembly structures.

SYNTAX

```
bom_roll_up_report [-u=user-id -p=password | -pf=password-file -g=group]
{[-item=item | [-key=[keyAttr1=keyVal1][,keyAttr2=keyVal2...,keyAttrN=keyValN]]}
[-rev=revision] [-revrule=revision-rule]}
[-effdate=mm:dd:yyyy:HH:MM:SS | now | today]
[-varrule=variant-option] [-name=name] [-desc=description]
-template=name:scope-context [-folder] [-h]
```

ARGUMENTS**-u**

Specifies the user ID.

This is generally **infodba** or another user with administration privileges. If this argument is used without a value, the operating system user name is used.

Note

If Security Services single sign-on (SSO) is enabled for your server, the **-u** and **-p** arguments are authenticated externally through SSO rather than being authenticated against the Teamcenter database. If you do not supply these arguments, the utility attempts to join an existing SSO session. If no session is found, you are prompted to enter a user ID and password.

-p

Specifies the password.

If used without a value, the system assumes a null value. This argument is mutually exclusive with the **-pf** argument.

If this argument is not used, the system assumes the *user-ID* value to be the password.

-pf

Specifies the password file. If used without a value, the system assumes a null value. If this argument is not used, the system assumes the *user-ID* value to be the password.

For more information about managing password files, see [Manage password files](#).

This argument is mutually exclusive with the **-p** argument.

-g

Specifies the group associated with the user.

If used without a value, the user's default group is assumed.

-item

Specifies the item to be used as the root line for the BOM properties rollup report.

-key

Specifies the key to be used as the root line for the BOM properties rollup report. Use the following format:

```
[keyAttr1=keyVal1] [,keyAttr2=keyVal2]...[,keyAttrN=keyValN]
```

To find the key of an object, use the **get_key_string** utility.

For more information, see the *Business Modeler IDE Guide*.

-rev

Specifies the item revision to be used as the root line for the BOM properties rollup report. If this argument is omitted, the report is based on the latest revision of the item.

-revrule

Specifies the revision rule to use when creating the BOM properties rollup report. If this argument is omitted, the default revision rule, **LATEST_WORKING**, is used.

-effdate

Specifies the date to use when configuring the effectivity of the assembly structure. If this argument is omitted, no effectivity date is set.

- *mm* specifies the month (01–12)
- *dd* specifies the day (01–31)
- *yyyy* specifies the year (0001–9999)
- *HH* specifies the hour (00–23)
- *MM* specifies the minute (00–59)
- *SS* specifies the second (00–59)

-varrule

Specifies the variant option set to use when setting the variant options of the assembly structure. If this argument is omitted, default variant options are used or no options are set.

-name

Specifies the name of the BOM properties rollup report. If this argument is omitted, an auto-generated name is created.

-desc

Contains a description of the BOM properties rollup report. If this argument is omitted, an autogenerated description is created. The autogeneration occurs only if a default description is defined in the BOM properties rollup template that was used to create the report.

-template

Specifies the name and scope context of the BOM properties rollup template to use when creating the BOM properties rollup report. Scope context is the user, group, or site scope identifier.

-folder

Indicates that the system is to place the new BOM properties rollup report dataset into the users **Newstuff** folder. If user privileges do not allow for BOM properties rollup report datasets to be attached to item revisions, the report is placed in the user's **Newstuff** folder.

-h

Displays help for this utility.

ENVIRONMENT

As specified in [Manually configuring your environment for Teamcenter utilities](#).

FILES

As specified in [Log files produced by Teamcenter](#).

RESTRICTIONS

None.

EXAMPLES

- The following example creates a BOM properties rollup report with autologin, autogeneration options, and attaches the report to the item revision:

```
bom_roll_up_report -item=12345678 -template="masstemplate:engineering"
```

- The following example creates a BOM properties rollup report with autologin, autogeneration options, and attaches the report to the **Newstuff** folder:

```
bom_roll_up_report -item=12345678 -template="masstemplate:engineering" -folder
```

- The following example creates a BOM properties rollup report with autologin, but with no autogeneration options:

```
bom_roll_up_report -item=12345678 -rev=B -name="My Report"
-desc="Validating mass values." -template="masstemplate:engineering"
```

- The following example creates a BOM properties rollup report with autologin and configures the assembly:

```
bom_roll_up_report -item=12345678 -rev=B -revrule="Released"
-effdate=02:20:2006 -varrule="High performance option set" -name="My Report"
-desc="Validating mass values." -template="masstemplate:engineering"
```

create_or_update_bbox_and_tso

This utility performs the following tasks:

- Creates, updates or deletes bounding box data from NX (**UGMASTER**) and JT (**DirectModel**) datasets.
- Creates or updates TruShape data (**.tso** files) for JT files.
- Generates reports of:
 - o NX or JT datasets not having a bounding box object.
 - o NX datasets not having a **UGPartBBox** form.
 - o JT datasets not having a **.tso** file. Using this report, the same utility can generate Dispatcher requests so the translation (conversion to the bounding box and TSO) occurs on the dedicated Dispatcher machine.

SYNTAX

```
create_or_update_bbox_and_tso [-u=user-id -p=password | -pf=password-file
-g=group] -mode=usermode -translation_mode=operatingmode -generate_ets_request
-delete_all_bboxes -dataset=dataset_uids -dataset_list=filename
-output_dir=dirname [-h]
```

ARGUMENTS**-u**

Specifies the user ID.

This is generally **infodba** or another user with administration privileges. If this argument is used without a value, the operating system user name is used.

Note

If Security Services single sign-on (SSO) is enabled for your server, the **-u** and **-p** arguments are authenticated externally through SSO rather than being authenticated against the Teamcenter database. If you do not supply these arguments, the utility attempts to join an existing SSO session. If no session is found, you are prompted to enter a user ID and password.

-p

Specifies the password.

If used without a value, the system assumes a null value. This argument is mutually exclusive with the **-pf** argument.

If this argument is not used, the system assumes the *user-ID* value to be the password.

-pf

Specifies the password file. If used without a value, the system assumes a null value. If this argument is not used, the system assumes the *user-ID* value to be the password.

For more information about managing password files, see [Manage password files](#).

This argument is mutually exclusive with the **-p** argument.

-g

Specifies the group associated with the user.

If used without a value, the user's default group is assumed.

-mode

Specifies one of the following modes:

- **query**

Generates a report of NX or JT datasets that require updates to the bounding boxes, TSO files (JT files only), or missing **UGPartBBox** forms (NX datasets only).

- **process**

Creates or updates bounding boxes, TSO files (for JT datasets only) or both for a set of NX or JT datasets.

- **query+process**

Generates a report of NX or JT datasets that require updates to the bounding boxes or TSO files (JT files only). It then creates or updates bounding boxes or TSO files (for JT datasets only) for the identified datasets.

- **delete**

Deletes the specified datasets.

-translation_mode

Specifies one of the following translation modes:

- **JTTOBBOX**

Creates or updates the bounding boxes in JT datasets.

- **JTTOTSO**

Creates or updates the TSO files in JT datasets.

- **JTTOBBOX+JTTOTSO**

Creates or updates the bounding boxes and TSO files in JT datasets.

- **NXBBOXTOBBOX**

Creates or updates the bounding boxes in NX datasets.

- **processAll**

Use with the **NXBBOXTOBBOX** mode to force creation of bounding boxes for all NX datasets.

- **NXBBOXFORM**

Creates a report listing all NX datasets for which the associated **UGPART-BBOX** form is not available. You can use this report file with the **ugmanager_refile** utility to generate **UGPART-BBOX** form data.

-generate_ets_request

Specify this argument when you specify **query** mode and are working only with JT datasets. Creates a Dispatcher request in the database for each JT dataset that needs updates to bounding boxes or TSO files. Before you use this argument, ensure the Dispatcher translation service is configured and running, as described in *Getting Started with Dispatcher (Translation Management)*.

-delete_all_bboxes

Deletes all the bounding boxes, multibounding boxes and relations between them from the database. (It does not delete TruShape data and you should re-create the TSO files if necessary.)

-dataset

Specifies one or more dataset UIDs as a string separated with commas, in the format:

```
ds1, ds2, ....., dsn
```

This argument is valid only if you specify **process** mode.

-dataset_list

Specifies the absolute path to an input file that contains a list of dataset UIDs to process. Each dataset must appear on a new line of this file. This argument is valid only if you specify **process** mode.

-output-dir

Specifies the absolute path to the directory where the report file is generated. If no path is specified, the report is generated in **/output_dir**. This argument is valid only if you specify **query** mode.

-h

Displays help for this utility.

ENVIRONMENT

As specified in [Manually configuring your environment for Teamcenter utilities](#).

FILES

As specified in [Log files produced by Teamcenter](#).

RESTRICTIONS

None.

EXAMPLES

- Create bounding box and TruShape data for all the JT datasets in the database. You can do this in one of two ways:

- o Create a report file of the datasets that do not have bounding box information as a log file. Process the log file and create the missing bounding box information on the dataset. This method is suitable if you have a large quantity of data to process as you can split the report log file can be split into multiple files, allowing the utility to process fewer datasets in each execution:

```
create_or_update_bbox_and_tso -u=user> -p=password> -g=group>
-mode=query -translation_mode=JTTOBBOX+JTTOTSO -output_dir=c:\temp
```

```
create_or_update_bbox_and_tso - u=user> -p=password> -g=group>
-mode=process -translation_mode=JTTOBBOX+JTTOTSO
-dataset_list=c:\temp\file generated by above command>
```

- o Query and process the datasets with a single command:

```
create_or_update_bbox_and_tso -u=<user> -p=<password> -g=<group>
```

```
-mode=query+process -translation_mode=JTTOBBOX+JTTOTSO
```

- Create bounding boxes for all NX datasets in the database. You can do this in one of two ways:
 - o Create a report file of the datasets that do not have bounding box information as a log file. Process the log file and create the missing bounding box information on the dataset. This method is suitable if you have a large quantity of data to process as you can split the report log file can be split into multiple files, allowing the utility to process fewer datasets in each execution:

```
create_or_update_bbox_and_tso -u=user> -p=password> -g=group>
-mode=query -translation_mode=NXBBOXTOBBOX -output_dir=c:\temp
```

```
create_or_update_bbox_and_tso -u=user> -p=password> -g=group>
-mode=process -translation_mode=NXBBOXTOBBOX
-dataset_list=c:\temp\file generated by above command>
```

- o Query and process the datasets with a single command:

```
create_or_update_bbox_and_tso -u=<user> -p=<password> -g=<group>
-mode=query+process -translation_mode=NXBBOXTOBBOX
```

- Create a report file of all NX datasets in a log file that do not have an associated **UGPART-BBOX** form:

```
create_or_update_bbox_and_tso -u=<user> -p=<password> -g=<group>
-mode=query -translation_mode=NXBBOXTOBBOX
-output_dir=c:\temp\report.txt
```

- Delete all bounding box and TruShape data associated with a specified item revision:

```
create_or_update_bbox_and_tso -u=<user> -p=<password> -g=<group>
-mode=delete -dataset=uid_of_item_rev1 or
absoccl[,uid_of_item_rev2 or absocc2, ...]
```

- Delete all bounding box and TruShape data associated with the parts specified in an input file:

```
create_or_update_bbox_and_tso -u=<user> -p=<password> -g=<group>
-mode=delete -dataset_list=<path of file containing a list of uids of item
revisions or absolute occurrences separated by new lines>
```

- Delete all bounding box data in the database:

```
create_or_update_bbox_and_tso -u=<user> -p=<password> -g=<group>
-delete_all_bboxes
```

generate_tc_ps_path

Runs an ITK program that accepts the changed part list and generates the paths up to the top assembly for each item revision.

SYNTAX

```
generate_tc_ps_path [-u=user-id -p=password | -pf=password-file -g=group]  
[-revision_rule=rule-name] -item_rev_list=rev-file-name  
[-item_type=item-type] [-out_file=output-file-name]  
-output_format= new | old [-configure_top_level_revs= yes | no]  
-rev_full_file=rule-file-name [-h]
```

ARGUMENTS**-u**

Specifies the user ID.

This is generally **infodba** or another user with administration privileges. If this argument is used without a value, the operating system user name is used.

Note

If Security Services single sign-on (SSO) is enabled for your server, the **-u** and **-p** arguments are authenticated externally through SSO rather than being authenticated against the Teamcenter database. If you do not supply these arguments, the utility attempts to join an existing SSO session. If no session is found, you are prompted to enter a user ID and password.

-p

Specifies the password.

If used without a value, the system assumes a null value. This argument is mutually exclusive with the **-pf** argument.

If this argument is not used, the system assumes the *user-ID* value to be the password.

-pf

Specifies the password file. If used without a value, the system assumes a null value. If this argument is not used, the system assumes the *user-ID* value to be the password.

For more information about managing password files, see [Manage password files](#).

This argument is mutually exclusive with the **-p** argument.

-g

Specifies the group associated with the user.

If used without a value, the user's default group is assumed.

-item_rev_list

Defines the file containing the list of item revisions. Typically, this file is the output from the [find_released_item_rev](#) utility or the [find_recently_saved_item_rev](#) utility. This file can also be custom made.

-revision_rule

Defines the revision rule on the basis of which the configured parent is returned. These revision rules are used to determine whether they configure the item revisions specified in the **item_rev_list** file.

This argument may be repeated for various revisions rules.

This argument is not allowed if the **rev_rule_file** argument is defined, specifying the name of a file containing the list of revision rules. Otherwise, this argument is mandatory.

-out_file

Specifies the file to be used to write the utility output. If not defined, the output is written in the standard output.

-output_format

Specifies whether the utility output is in new format or old format. This argument is optional. If it is not defined, the new format is used. Possible values for this option are **new** and **old**.

The old format is as follows:

```
@DB/VEH0001/004,@DB/VPPS0002/001,@DB/VPPS0006/
001,@DBIA0009/002:Precise;Aplp2 Best w/PDI
```

The new format is as follows:

```
PathPartRev@DBseparatorItem IDseparatorRevID/PartRev
[PartRev@DBseparatorItemIDseparatorRevID]/PartRev]
RevisionRuleRevisionrulename/RevisionRule/Path
```

-item_type

Specifies the item type of the top-level assembly. The utility lists only those paths with top-level assemblies of this type. In cases where such assemblies have parents, the defined path begins at the item with the given type. This argument is optional.

-rev_rule_file

Lists all the revisions rules to be used for the found item revisions. This argument is optional only if the revision rule arguments are defined; otherwise, this argument is required.

If both the **revision_rule** argument and the **rev_rule_file** argument are defined, the **rev_rule_file** argument takes precedence.

-configure_top_level_revs

Specifies whether to configure the top-level revision. This argument is optional.

If **Yes** is specified, the top-level item of the changed item revision is configured and the changed item revision is checked to determine if it uses the top-level item revision.

If **No** is specified, or the argument is not specified, the top levels are not configured.

-h

Displays help for this utility.

ENVIRONMENT

This utility should be run from a shell where the Teamcenter environment is set.

FILES

As specified in *Log files produced by Teamcenter*.

RESTRICTIONS

None.

EXAMPLES

To generate a path for the item revisions specified in a text file:

```
generate_tc_ps_path -item_rev_list=released_items.txt  
-revision_rule=revision-rule-name -output_file=path_list.txt  
-output_format=new -configure_top_level_revs=no
```

identify_non_structure_edges

Marks parent and child items that are ignored when adding, listing, or removing entries in spatial indexes. There are four ways to use this utility.

- Adding and specifying what to add using parallel lists of item IDs or UIDs. For example, in a parallel list of parents and children, the first parent goes with the first child, the second parent goes with the second child, and so on.
- Adding and specifying what to add by a parent item type and child item type.
- Listing all elements or optionally specifying that only elements relevant to a particular product or particular parent or child objects be listed.
- Removing and specifying what to remove by giving parallel lists of item IDs or UIDs. For example, in a parallel list of parents and children, the first parent goes with the first child, the second parent goes with the second child, and so on.

SYNTAX

```
identify_non_structure_edges [-u=user-ID [-p=password | -pf=password-file]
[-g=group] ]
[-add {-parent_item_id={parent-object1, parent-object2, ... | @filename}
-child_item_id={child-object1, child-object2, ... | @filename} |
-parent_uid={parent-uid1, parent-uid2, ... | @filename}
-child_uid={child-uid1, child-uid2, ... | @filename} |
-parent_item_type=parent-type -child_item_type=child-type} ]
[-remove {-parent_item_id={parent-object1, parent-object2, ... | @filename}
-child_item_id={child-object1, child-object2, ... | @filename} |
-parent_uid={parent-uid1, parent-uid2, ... | @filename}
-child_uid={child-uid1, child-uid2, ... | @filename} } ]
[-list {-all [parent1, parent2, ... | @filename] |
[child1, child2, ... | @filename] }
{product={product1, product2, ...}
[-parent_item_id={parent1, parent2, ... | @filename}
[-child_item_id={child1, child2, ... | @filename} ] |
[-parent_uid={parent1, parent2, ... | @filename}
[-child_uid={child1, child2, ... | @filename} ] }
[-h]
```

ARGUMENTS

-u

Specifies the user ID. The user must have administrative privileges.

If this argument is used without a value, the operating system user name is used.

Note

If your Teamcenter server uses Security Services single sign-on, see [Before you begin](#) for additional information.

-p

Specifies the user's password.

If used without a value, the system assumes a null value. If this argument is not used, the system assumes the *user-ID* value to be the password.

This argument is mutually exclusive with the **-pf** argument.

-pf

Specifies the password file. If used without a value, the system assumes a null value. If this argument is not used, the system assumes the *user-ID* value to be the password.

For more information about managing password files, see [Manage password files](#).

This argument is mutually exclusive with the **-p** argument.

-g

Specifies the group associated with the user.

-add

Adds entries for parent and child pairs. You can specify a parent and child object pair using item IDs or UIDs. You enter the identifiers as parallel, comma-separated lists for the parent and child flags.

For long lists of values, you can create parent and child object files that contain a separate identifier on each line. Use the *@file-name* values to supply the file names on the command line in place of the comma-separated lists.

Alternatively, you can use the **-parent_item_type=parent-type** and **-child_item_type=child-type** flags to specify a parent item and child item type pair.

-remove

Removes entries for parent and child pairs. You can specify a parent and child object pair using item IDs or UIDs. You enter the identifiers as parallel comma-separated lists for the parent and child flags.

For long lists of values, you can create parent and child object files that contain a separate identifier on each line. Use the *@file-name* values to supply the file names on the command line in place of the comma-separated lists.

-list

Lists all entries when used with the **-all** argument without the optional parent or child object specifications. You can filter the listed entries by specifying parent or child objects using item IDs or UIDs. You can narrow the filter by specifying both parent and child objects to get a list of entries containing the parent objects that have at least one of the specified child objects. You enter the identifiers as comma-separated lists for the parent and child arguments.

Optionally, you can list only those entries associated with one or more specified products using the **-product=product-name** argument. You can filter the product entries by specifying parent or child objects or both parent and child objects in the same way as the **-all** argument.

For long lists of values, you can create parent and child object files that contain a separate identifier on each line. Use *@file-name* values to supply the file names on the command line in place of the comma-separated lists.

EXAMPLES

- To add parent-item and child-item pairs **pitem1**, **pitem2** and **citem1**, **citem2**, enter the following command on a single line:

```
identify_non_structure_edges -u=infodba -p=infodba -g=dba
-add -parent_item_id=pitem1,pitem2,pitem3
    -child_item_id=citem1,citem2,citem3
```

- To add parent-item and child-item pairs **puid1**, **puid2** and **cuid1**, **cuid2**, enter the following command on a single line:

```
identify_non_structure_edges -u=infodba -p=infodba -g=dba
-add -parent_uid=puid1,puid2,puid3,puid4
-child_uid=cuid1,cuid2,cuid3,cuid4
```

- To add parent-item and child-item pairs where the parent item is of type **ItemType1** and the child item is of type **ItemType2**, enter the following command on a single line:

```
identify_non_structure_edges -u=infodba -p=infodba -g=dba
-add -parent_item_type=pitemtype
-child_item_type=citemtype
```

Note This example uses the parent and child item type, rather than the parent and child item ID.

- To list all parent-item and child-item pairs, enter the following command on a single line:

```
identify_non_structure_edges -u=infodba -p=infodba -g=dba -list
-all
```

- To list parent-item and child-item pairs by product, enter the following command on a single line:

```
identify_non_structure_edges -u=infodba -p=infodba -g=dba
-list -product=product1
```

- To remove parent-item and child-item pairs using item IDs, enter the following command on a single line:

```
identify_non_structure_edges -u=infodba -p=infodba -g=db
-remove -parent_item_id=pitem1,pitem2,pitem3
-child_item_id=citem1,citem2,citem3
```

- To remove parent-item and child-item pairs using UIDs, enter the following command on a single line:

```
identify_non_structure_edges -u=infodba -p=infodba -g=dba
-remove -parent_uid=puid1,puid2,puid3,puid4
-child_uid=cuid1,cuid2,cuid3,cuid4
```

- To remove parent-item and child-item pairs using a file containing the item IDs, create a parent and child file containing a single ID on each line. The pairs are determined by the order of the values in the file, for example:

Create a parentList.txt file with the following contents:

```
pitem1
pitem2
:
pitemN
```

Create the childList.txt file with the following contents:

```
citem1
citem2
:
citemN
```

Enter the following command on a single line:

```
identify_non_structure_edges -u=infodba -p=infodba -g=dba  
-remove -parent_item_id=@parentList.txt -child_item_id=@childList.txt
```

item_to_part_design

Converts instances of the **Item** class and its subtypes to specified instances of the **Part** and **Design** classes or their subclasses or subtypes. Use this utility if you have existing product structure containing items that you want to convert to separate part and design (CAD) objects. You must create a text file that provides the necessary input for the conversion process.

Note You must restart the Teamcenter server after running this utility for the new objects to be visible in the rich client.

The utility allows you to:

- Convert all objects in the system or a specified set of objects of a given source type to given target types.
- Specify the source as the **Item** class or its subtypes and specify the target as the **Part** and **Design** classes, their subclasses, or subtypes.

Caution After you run this utility, you cannot revert the changes.

You must analyze the impact on business rules (for example, property rules and GRM rules) of the conversion and then plan accordingly. The utility does not check the validity of the business rules.

Performance may be poor if you try to convert the entire database or a large structure. If possible, limit the structure size or the number of objects to convert.

The following conversions are supported by this utility:

- **Item** subtypes to **Part** subtypes
- **Item** subtypes to **Design** subtypes
- **Item** subtypes to **Part** class
- **Item** subtypes to **Part** subclass
- **Item** subtypes to **Design** class
- **Item** subtypes to **Design** subclass
- **Item** class to **Part** class
- **Item** class to **Part** subclass
- **Item** class to **Design** class
- **Item** class to **Design** subclass

The utility operates in three modes:

- **type_based**

Input from the user is an item class or its subtype, which is a source type. The user specifies a target type. All objects in the database of source type are converted to the target type.

- **item_id_based**

In this mode, the user specifies a comma-separated list of item IDs, together with the source and the target types. Only the *valid* item IDs listed in the input file are converted from the source type to the target type.

- **structure_based**

The user specifies the item identifier of the top-level item in the structure, together with the source and the target types. This mode converts all the valid children of the top-level item to the target type.

The user must create a text file that contains input to the entire conversion process.

SYNTAX

```
item_to_part_design [-u=user-id -p=password | -pf=password-file -g=group]
{-mode=type_based | item_id_based [-itemid=item-id | -key_property_list=key-id-list]
| structure_based [-itemid=item-id | -itemkey=key-id] }
-file=input-file-name
[-rf=report-file-name] [-dryRun=true/on | false/off]
[-h]
```

ARGUMENTS

-u

Specifies the user ID.

This is generally **infodba** or another user with administration privileges. If this argument is used without a value, the operating system user name is used.

Note

If Security Services single sign-on (SSO) is enabled for your server, the **-u** and **-p** arguments are authenticated externally through SSO rather than being authenticated against the Teamcenter database. If you do not supply these arguments, the utility attempts to join an existing SSO session. If no session is found, you are prompted to enter a user ID and password.

-p

Specifies the password.

If used without a value, the system assumes a null value. This argument is mutually exclusive with the **-pf** argument.

If this argument is not used, the system assumes the *user-ID* value to be the password.

-pf

Specifies the password file. If used without a value, the system assumes a null value. If this argument is not used, the system assumes the *user-ID* value to be the password.

For more information about managing password files, see [Manage password files](#).

This argument is mutually exclusive with the **-p** argument.

-g

Specifies the group associated with the user.

If used without a value, the user's default group is assumed.

-mode

Specifies **type_based**, **item_id_based**, or **structure_based**.

-itemid

Specifies the ID of the item to be exported. Valid only if no input file is specified using the **-i** option.

-itemkey

Specifies the key of the object. You can use the **-itemid** argument or the **-itemkey** argument.

To find the key of an object, use the [get_key_string](#) utility.

For more information, see the [Business Modeler IDE Guide](#).

-key_property_list

Specifies the key attributes of the object except the item ID.

To find the key of an object, use the [get_key_string](#) utility.

For more information, see the [Business Modeler IDE Guide](#).

-file

This argument is mandatory for all the conversion modes.

Specifies the name of the input file containing conversion parameters, including the full path. The input file is a list of key-value pairs. The keys indicate the source and target item types. The input file has the following format:

```
source_class_name=Item class | SubType
target_class_name=Part/Design class | SubClass | SubType
item_id_list=Name of input file
```

Ensure you use internal names for the source and target.

Note

The list of item IDs must end with a comma (,).

-rf

This optional argument specifies the name and location of the report file. If you do not specify a name and location, the system writes the file to *C:\temp* on Microsoft Windows systems or */tmp* on other systems. It is not applicable in **type_based** mode.

-dryRun

This optional argument causes the utility to validate if the specified items or structure can be validate, but does not perform the conversion. Valid values are **true** or **on** and **false** or **off** (case sensitive). It is not applicable in **type_based** mode.

-h

Displays help for this utility.

ENVIRONMENT

As specified in [Manually configuring your environment for Teamcenter utilities](#).

FILES

As specified in *Log files produced by Teamcenter*.

RESTRICTIONS

- Execute this utility when no other activity is present on the database.
- This utility changes only the type of item object, its revisions, master form, and revision master form.
- The storage class of source and target master forms is assumed to be same. If the attributes on the source and target form storage class are different, the utility does not know how to map those attributes. Hence the utility does not change the storage class of the form, but converts the form object itself to the target class. For example, while converting the standard **Item** class to **Design**, it performs the following conversion:
 - o Before conversion, the item master form (a form storage class) is an item master.
 - o After conversion, the design master form (a form storage class) is an item master.
- The utility converts objects in a particular database. If the objects being converted are exported to different sites, either the replica of the object must be deleted before conversion or replica objects also need to be converted in a similar fashion. If the replica objects are not converted due to change of type, synchronizing of data does not occur.
- If you use **item_id_based** mode or **structure_based** mode, items with export records or conflicting GRM rules are not converted. They are converted if you use **type_based** mode.
- The conversion of item subclass instances is not supported.
- The utility does not support source and target classes or subtypes other than those listed in the description.
- Any customization of the source object type is not supported after conversion to the target type.
- If the target is a subclass of **Part** or **Design** with custom attributes, those custom attributes of the converted instance or object are populated with default values, if any are defined. If none are defined, the custom attributes are null.

EXAMPLES

- The following example uses **type_based** mode and converts all the objects of a source type to a target type defined in the **Sample.txt** file:

```
item_to_part_design -u=infodba -p=infodba -g=dba
-mode=type_based -file="C:\Sample.txt"
```

The contents of **Sample.txt** are:

```
source_class_name=Item
target_class_name=Part
```

- The following example uses **item_id_based** mode and converts all of the *valid* items listed in the **Sample.txt** file from source type to target type. No dry run is performed, but a report is written to the **ReportFile.txt** file.

```
item_to_part_design -u=infodba -p=infodba -g=dba
  -mode=item_id_based -rf="C:\temp\ReportFile.txt"
  -file="C:\Sample.txt" -dryRun=false
```

The contents of **Sample.txt** are:

```
source_class_name=Item
target_class_name=Part
item_id_list=000001,000004,000009,
```

- The following example uses **item_id_based** mode and converts all of the *valid* items listed in the **Sample.txt** file from source type to target type. A dry run is performed, and a report is written to the default location.

```
item_to_part_design -u=infodba -p=infodba -g=dba
  -mode=item_id_based -file="C:\Sample.txt" -dryRun=true
```

The contents of **Sample.txt** are the same as in the previous example.

- The following example uses **structure_based** mode and converts all the *valid* items and children of the specified item identifier from a source type to a target type defined in the **Sample.txt** file. No dry run is performed, but a report is written to the **ReportFile.txt** file.

```
item_to_part_design -u=infodba -p=infodba -g=dba
  -mode=structure_based
  -rf="C:\temp\ReportFile.txt" -file="C:\Sample.txt"
  -dryRun=false -itemId=000008
```

The contents of **Sample.txt** are:

```
source_class_name=Item
target_class_name=Part
```

Note **item_id_list** is not required.

- The following example uses **item_id_based** mode with the **key_property_list** argument. Because the multifield key is defined as **item_id,object_type**, only the **object_type** property is given (which is **Mfk2Item**).

```
item_to_part_design -u=infodba -p=infodba -mode=item_id_based
  -key_property_list=object_type=Mfk2Item -file=itemlist.txt
```

multiple_svr_variant_configurator

Assists in creating configured structure representations using saved variant rules (SVRs) from unconfigured structure representations.

- Use the **bomwriter** utility to generate an unconfigured PLM XML file containing product structure information, including the variant conditions for child lines.
- Use the **multiple_svr_variant_configurator** utility to read the unconfigured PLM XML file and the specified SVRs.

The utility generates multiple pruned PLM XML files, corresponding to each SVR.

- Optionally, import pruned PLM XML files back into Teamcenter as **DirectModelAssembly** datasets using the **import_file** utility.
- Open the configured PLM XML files or datasets in Lifecycle Visualization.

Note Before running this utility, there must be a **DirectModelAssembly** dataset with a **TCEng_rdv_plmxml_unconfigured** relation placed under the product revision containing the unconfigured PLM XML file. Import the dataset using the **import_file** utility, ensuring that the variant XML preexists as a named reference of the unconfigured dataset.

SYNTAX

```
multiple_svr_variant_configurator [-u=user-id -p=password | -pf=password-file
-g=group] -product_id=product-item-ID -product_rev=product-revision-ID
-dataset_name=unconfigured-dataset [-log_file=file-name]
-directory_name=directory-name
{-svr_input_file=file-name | -process_all_svrs=Y} [-is_import_required=Y]
[-import_utility=path-to-utility]
[-import_utility_parameters=utility-parameters] [-h]
```

ARGUMENTS

-u

Specifies the user ID.

This is generally **infodba** or another user with administration privileges. If this argument is used without a value, the operating system user name is used.

Note If Security Services single sign-on (SSO) is enabled for your server, the **-u** and **-p** arguments are authenticated externally through SSO rather than being authenticated against the Teamcenter database. If you do not supply these arguments, the utility attempts to join an existing SSO session. If no session is found, you are prompted to enter a user ID and password.

-p

Specifies the password.

If used without a value, the system assumes a null value. This argument is mutually exclusive with the **-pf** argument.

If this argument is not used, the system assumes the *user-ID* value to be the password.

-pf

Specifies the password file. If used without a value, the system assumes a null value. If this argument is not used, the system assumes the *user-ID* value to be the password.

For more information about managing password files, see [Manage password files](#).

This argument is mutually exclusive with the **-p** argument.

-g

Specifies the group associated with the user.

If used without a value, the user's default group is assumed.

-product_id

Specifies the product item ID. Specify the top level of the assembly for which you want to generated configured PLM XML files.

-product_rev

Specifies the product revision ID. Specify the top level of the assembly containing the SVRs.

-dataset_name

Specifies the name of the dataset under the product revision containing the unconfigured files. This must be a **DirectModelAssembly** dataset with a **TCEng_rdv_plmxml_unconfigured** relation placed under the product revision containing the unconfigured PLM XML file.

-log_file

Specifies the full path to the log file in which activity is recorded. Use this argument only when importing the pruned PLM XML files.

-directory_name

Specifies the full path to the directory in which the pruned PLM XML files are stored. The user running the utility must have write access to this directory.

-svr_input_file

Specifies the full path to the file containing the SVRs with which to configure the assembly.

You must specify either the **-svr_input_file** or the **-process_all_svrs** argument for the utility to run. If both are specified, **-svr_input_file** takes precedence.

-process_all_svrs

Specifies that all SVRs are processed. The valid value is **Y**.

You must specify either the **-svr_input_file** or the **-process_all_svrs** argument for the utility to run. If both are specified, **-svr_input_file** takes precedence.

-is_import_required

Specifies whether the pruned files are imported. Valid values are **Y** and **N**. The default setting is **N**.

If you set this argument to **Y**, you must set the **-import_utility** and **-import_utility_parameters** arguments.

-import_utility

Specifies the full path to the [import_file](#) utility, used to import the pruned files.

If you set this argument, you must set the **-import_utility_parameters** argument.

-import_utility_parameters

Sets the required parameters of the **import_file** utility. Specify parameters in a single string. Separate each parameter with a hash sign (#). For example:

```
-import_utility_parameters=#-u=userID#-p=password#-g=group>#-d=ConfiguredAll
#-ref=ConfiguredAssembly#-type=DirectModelAssembly#
-relation=TCEng_rdv_plmxml_configured#
-desc=ConfiguredAll#-use_existing=no#-f=#-item=ABC00004#-revision=001#
```

-h

Displays help for this utility.

ENVIRONMENT

As specified in [Manually configuring your environment for Teamcenter utilities](#).

FILES

As specified in [Log files produced by Teamcenter](#).

RESTRICTIONS

Before running this utility, there must be a **DirectModelAssembly** dataset with a **TCEng_rdv_plmxml_unconfigured** relation placed under the product revision containing the unconfigured PLM XML file. Import the dataset using the **import_file** utility, ensuring that the variant XML preexists as a named reference of the unconfigured dataset.

EXAMPLES

- In the following example, all SVRs listed in the **input_svr.txt** file are used to generate multiple pruned PLM XML files, one file for each SVR. The files are not imported. The pruned PLM XML files are generated from the **multi_svr** directory.

```
multiple_svr_variant_configurator -u=userID -p=password -g=admin -product_id=ABC004
-product_rev=001 -dataset_name=ABC00004Unconfigured2
-directory_name=c:\temp\multi_svr -svr_input_file=c:\temp\multi_svr\input_svr.txt
```

- In the following example, all SVRs under the **ABC004/001** product revision are used to generate multiple pruned PLM XML files, one file for each SVR. The files are not imported. The pruned PLM XML files are generated from the **multi_svr** directory.

```
multiple_svr_variant_configurator -u=userID -p=password -g=admin -product_id=ABC004
-product_rev=001 -dataset_name=ABC00004Unconfigured2
-directory_name=c:\temp\multi_svr -process_all_svrs=y
```

- In the following example, all SVRs under the **ABC004/001** product revision are used to generate multiple pruned PLM XML files, one file for each SVR. The files are then imported back into the Teamcenter database.

```
multiple_svr_variant_configurator -u=userID -p=password -g=admin
-product_id=ABC004 product_rev=001 -dataset_name=ABC00004Unconfigured2
-directory_name=c:\temp\multi_svr -process_all_svrs=y -is_import_required=Y
-log_file=c:\temp\multi_svr\log_file.txt
-import_utility={TC_ROOT}\bin\import_file -import_utility_parameters=#-u=userID
#-p=password#-g=admin#-d=ConfiguredAll#-ref=ConfiguredAssembly
#-type=DirectModelAssembly#-relation=TCEng_rdv_plmxml_configured
#-desc=ConfiguredAll
#-use_existing=no#-f=#-item=ABC00004#-revision=001#
```

The parameters for the **import_file** utility are set in a single string, each parameter separated by a hash mark (#). The **-d** parameter specifies the prefix to the dataset names created upon import, followed by an underscore. For example:

```
ConfiguredAll_
```

You must include the **-f** parameter in the parameter string for the **import_file** utility. Do not assign it a value. The import utility automatically supplies the pruned PLM XML files.

ps_exportconfignxassembly

Enables a site to export a configured NX assembly. The assembly is configured using the given revision rule and saved variant rule.

SYNTAX

```
ps_exportconfignxassembly [-u=user-id -p=password |  
-pf=password-file -g=group]  
-item=top-item-id | -key=keyAttr1=keyVal1 [,keyAttr2=keyVal2]...[,keyAttrN=keyValN]  
-rev=top-rev-id [-revrule=revision-rule]  
-variant=saved-variant-rule [-scopeitem=item-id]  
[-scoperev=rev-id] [-display] [-verbose=y | yes | n | no]  
[-exclude=itemid1,itemid2,itemiid3,...  
| -excludekeys=keyAttr1=keyVal1 [,keyAttr2=keyVal2]...[,keyAttrN=keyValN];  
keyAttr1=keyVal1 [,keyAttr2=keyVal2]...[,keyAttrN=keyValN];...] [-h]
```

ARGUMENTS**-u**

Specifies the user ID.

This is a required argument unless the [TC_auto_login](#) site preference is set.

This is generally **infodba** or another user with administration privileges. If this argument is used without a value, the operating system user name is used.

Note

If Security Services single sign-on (SSO) is enabled for your server, the **-u** and **-p** arguments are authenticated externally through SSO rather than being authenticated against the Teamcenter database. If you do not supply these arguments, the utility attempts to join an existing SSO session. If no session is found, you are prompted to enter a user ID and password.

-p

Specifies the password.

This is a required argument unless the [TC_auto_login](#) site preference is set.

If used without a value, the system assumes a null value. This argument is mutually exclusive with the **-pf** argument.

If this argument is not used, the system assumes the *user-ID* value to be the password.

-pf

Specifies the password file. If used without a value, the system assumes a null value. If this argument is not used, the system assumes the *user-ID* value to be the password.

For more information about managing password files, see [Manage password files](#).

This argument is mutually exclusive with the **-p** argument.

-g

Specifies the group associated with the user.

This is a required argument unless the [TC_auto_login](#) site preference is set.

If used without a value, the user's default group is assumed.

-item

Specifies the item ID of the item to be exported. This argument is required unless **-key** is defined. If both **-item** and **-key** are defined, **-key** takes precedence.

-key

Specifies the key of the item to be exported. This argument is required unless **-item** is defined. If both **-item** and **-key** are defined, **-key** takes precedence.

To find the key of an object, use the [get_key_string](#) utility.

For more information, see the [Business Modeler IDE Guide](#).

-rev

Specifies the revision ID of the top line of the structure to be exported. This argument is optional. If specified, the rest of the structure below the top line is configured by the **-revrule** argument. If not specified, the entire exported structure is configured by the **-revrule** argument.

-revrule

Specifies the revision rule to use to configure the structure. This argument is required.

-variant

Specifies the name of the variant rule to apply to configure the structure. This argument is required.

-scopeitem

This is an optional argument. Include this argument if and only if the given variant rule is not attached to the exporting item (top item) but is attached to this scope item's revision. The item revision is specified in the **scoperev** argument.

-scoperev

This is an optional argument. Include this argument if and only if the given variant rule is not attached to the exporting item (top item) but is attached to this **scoperev**.

-display

Displays the output folder to the screen. This argument is optional. If this argument is not specified, the output folder is not displayed at the end of the successful operation.

-verbose

Prints the debug statement. This argument is optional. The default value is **n**.

-exclude

Specifies the list of item IDs to exclude from exporting after the structure is configured using the revision rule and saved variant rule. This argument is optional.

-excludekeys

Specifies the list of keys to exclude from exporting after the structure is configured using the revision rule and saved variant rule. This argument is optional.

-h

Displays help for this utility.

ENVIRONMENT

NX must be installed and configured.

As specified in *Manually configuring your environment for Teamcenter utilities*.

FILES

As specified in *Log files produced by Teamcenter*.

RESTRICTIONS

The files are created in the folder specified by the **TC_TMP_DIR** variable. The value of **TC_TMP_DIR** must be set to the desired destination directory temporarily before running the utility.

EXAMPLES

- Exports **TopAssmRevA** after configuring the structure below the top line with the **Latest Working** revision rule and **Tire200** saved variant rule. **SubAssm1** is excluded from the export even though it was configured.

```
ps_exportconfignxassembly -u=infodba -p=infodba -g=dba
  -item=TopAssm1 -rev =TopAssmRevA -revrule=Latest Working
  -variant=Tire200 -display -verbose=y -exclude=SubAssm1
```

- Exports **TopAssmRevA** after configuring the structure below the top line with the **Latest Working** revision rule and **Tire200** saved variant rule. There is no exclusion list provided in this example.

```
ps_exportconfignxassembly -u=infodba -p=infodba -g=dba
  -item=TopAssm1 -rev =TopAssmRevA -revrule=Latest Working
  -variant=Tire200 -display -verbose=n
```

- Exports **SubAssmRevA** after configuring the structure below the top line with the **Latest Working** revision rule and **Tire200** saved variant rule. The saved variant rule is attached to **TopAssmRevA**.

```
ps_exportconfignxassembly -u=infodba -p=infodba -g=dba
  -item=SubAssm1 -rev =SubAssmRevA -revrule=Latest Working
  -variant=Tire200 -scopeitem=TopAssm1 -scoprev=TopAssmRevA
```

- Exports revision **A** of the item with key of **CarModel**. The structure below the top line is configured with the **Latest Working** revision rule and **Car1** saved variant rule.

```
ps_exportconfignxassembly -u=xxx -p=yyy
  -key=item_id=CarModel -rev=A
  -revrule="Latest Working" -variant=Car1 -verbose=y
```

ps_rename_bvrs

Renames BOM views and BOM view revisions using the current naming scheme. The new name can differ from the old name because the naming scheme has changed or because the name of the view type has changed.

By default, the utility runs on all BOM views and BOM view revisions in the database, but it can accept an item ID argument (including wildcards) or a key argument, defining a set of objects to rename.

SYNTAX

```
ps_rename_bvrs[-u=user-id -p=password | -pf=password-file -g=group]  
[-item=item_pattern]  
| -key=[keyAttr1=keyVal1] [,keyAttr2=keyVal2]...[,keyAttrN=keyValN]  
[-view=view-type] [-h]
```

ARGUMENTS

-u

Specifies the user ID.

This is a required argument unless the [TC_auto_login](#) site preference is set.

This is generally **infodba** or another user with administration privileges. If this argument is used without a value, the operating system user name is used.

Note

If Security Services single sign-on (SSO) is enabled for your server, the **-u** and **-p** arguments are authenticated externally through SSO rather than being authenticated against the Teamcenter database. If you do not supply these arguments, the utility attempts to join an existing SSO session. If no session is found, you are prompted to enter a user ID and password.

-p

Specifies the password.

This is a required argument unless the [TC_auto_login](#) site preference is set.

If used without a value, the system assumes a null value. This argument is mutually exclusive with the **-pf** argument.

If this argument is not used, the system assumes the *user-ID* value to be the password.

-pf

Specifies the password file. If used without a value, the system assumes a null value. If this argument is not used, the system assumes the *user-ID* value to be the password.

For more information about managing password files, see [Manage password files](#).

This argument is mutually exclusive with the **-p** argument.

-g

Specifies the group associated with the user.

This is a required argument unless the [TC_auto_login](#) site preference is set.

If used without a value, the user's default group is assumed.

-item

Specifies which BOM views or BOM view revisions to rename by a pattern match on the parent item ID.

-key

Specifies which BOM views or BOM view revisions to rename by key. Use the following format:

```
[keyAttr1=keyVal1] [,keyAttr2=keyVal2]...[,keyAttrN=keyValN]
```

To find the key of an object, use the [get_key_string](#) utility.

For more information, see the [Business Modeler IDE Guide](#).

-view

Specifies which BOM views or BOM view revisions to rename by BOM view type. The default is to rename BOM views or BOM view revisions of all types.

-v

Verbose mode.

-h

Displays help for this utility.

ENVIRONMENT

As specified in [Manually configuring your environment for Teamcenter utilities](#) and the following preferences:

[TC_ignore_case_on_search](#)

[TC_pattern_match_style](#)

For more information about these preferences, see the [Preferences and Environment Variables Reference](#).

FILES

As specified in [Log files produced by Teamcenter](#).

RESTRICTIONS

Wildcard characters used in the **-item** argument may require enclosure in double quotation marks to prevent the shell from expanding them.

EXAMPLES

- To update names of all BOM views and BOM view revisions (BVRs) in the database, enter the following command:

```
$TC_ROOT/bin/ps_rename_bvrs
```

- Consider a site where the **Manufacturing** BOM view type is renamed to **M Site 1**. To rename all BOM views and BOM view revisions with parent item IDs beginning **pbx** to agree with the new name, enter the following command on a single line:

```
$TC_ROOT/bin/ps_rename_bvrs -view="M Site 1" -item="pbx"
```

ps_traverse

Traverses a product structure and reports BOM line attributes and workspace attribute values in a file in delimited format. It also sets an assembly to precise and releases/transfers ownership of the item revisions that constitute the structure. The inputs for these are taken from a configuration file and the input for product structure configuration are taken from the command line options.

A configuration file for input is mandatory. If the **te.cfg** file exists in the current directory, it is considered. If not, a configuration file must be specified by the **-cfg** options. A sample configuration file is provide in the **\$TC_ROOT** directory.

SYNTAX

```
ps_traverse [-u=user-id -p=password | -pf=password-file -g=group]
-itemid=item-to-traverse
-rev=revision-for-item-to-traverse
[-revrule=revision-rule-to-configure-BOM-window]
[-viewtype=type-of-item-revision-BVR-to-traverse]
[-variant=saved-variant-object-to-configure-BOM-window]
[-log=log-file-for-session-output]
[-packlines= true | false]
[-cfg=configuration-file-for-options]
[-report=report-file-for-output]
[-h]
```

ARGUMENTS

-u

Specifies the user ID.

This is generally **infodba** or another user with administration privileges. If this argument is used without a value, the operating system user name is used.

Note

If Security Services single sign-on (SSO) is enabled for your server, the **-u** and **-p** arguments are authenticated externally through SSO rather than being authenticated against the Teamcenter database. If you do not supply these arguments, the utility attempts to join an existing SSO session. If no session is found, you are prompted to enter a user ID and password.

-p

Specifies the password.

If used without a value, the system assumes a null value. This argument is mutually exclusive with the **-pf** argument.

If this argument is not used, the system assumes the *user-ID* value to be the password.

-pf

Specifies the password file. If used without a value, the system assumes a null value. If this argument is not used, the system assumes the *user-ID* value to be the password.

For more information about managing password files, see [Manage password files](#).

This argument is mutually exclusive with the **-p** argument.

-g

Specifies the group associated with the user.

If used without a value, the user's default group is assumed.

-itemid

Specifies the ID of the item for which the associated BOM view revision (BVR) is traversed. (BOM view revisions are associated with revisions corresponding to the specified item.)

-rev

Specifies the revision of the item specified by the **-itemid** argument. The revision must have an associated BVR.

-viewtype

Specifies the type of the BVR to be traversed.

-revrule

Specifies the revision rule used to configure the BOM window. The default revision rule is **Latest Working**.

-variant

Specifies the saved variant used to configure the BOM window. If more than one saved variant exists, the first one found is considered.

-log

Specifies the log file to which the output is directed. The default file is **te.log**.

-report

Specifies the file to which the report is written. The default file is **tereport.txt**.

-packlines

Indicates whether to pack or unpack BOM lines.

-cfg

Specifies the input configuration file. The default file is **te.cfg**.

-h

Displays help for this utility.

**CONFIGURATION
FILE ENTRIES**

The configuration file is a text file in which entries must be made under the following separate headings:

bomreport=

Valid values for this attribute are the display names of columns in Structure Manager, for example, **Rule configured by** and **Sequence No.**, and are case sensitive. Values listed in the columns are reported for each node in the BOM.

woreport=

The values of these object attributes in workspace listed under this entry are reported.

formattributes=

The form attributes to be reported are listed under this string. These should be in the format **Form Type Name:attribute**. The form values are truncated to 200 characters.

action=

The actions to be performed while traversing product structure are listed under this heading. The following actions are supported:

- **fastrelease**
- **changeowner**
- **setprecise**

Note When the **setprecise** action is specified, other actions and reporting inputs are ignored and the utility makes only the specified assembly precise.

relstat=

Specifies the release status to be applied if the specified action is **fastrelease**.

newowner=

Lists the new owner to whom the object ownership is transferred if the **changeowner** action is specified.

alternate=

Specifies whether alternates are processed. Valid values are **Yes** and **No**.

delimiter=

Specifies the delimiter used to separate attribute values in report generation. The default delimiter is a semicolon (;).

columnwidth=

Specifies the attribute values used In report generation. The default column width is 20 characters.

EXAMPLES

Following is the content of the **ps_traverse.cfg** sample configuration file, which is located in the **\$TC_ROOT\sample\examples** directory.

```
alternate=
yes
bomreport=
BOM Line Name
woreport=
Name
Revision
Owner
formattributes=
delimiter=
#
columnwidth=
25
action=
#fastrelease
#changeowner
#setprecise
relstat=
X
newowner=
infodba
group=
dba
```

Sample configuration file

ps_upload

Creates an imprecise product structure based on an input file.

SYNTAX

ps_upload [-u=*user-id* -p=*password* | -pf=*password-file* -g=*group*] [-o= **yes** | **no**] [-f] [-c= **Item** | **Architecture**] [-t=*type*] [-v] [-i=*input-file*] [-h]

ARGUMENTS

-u

Specifies the user ID.

This is generally **infodba** or another user with administration privileges. If this argument is used without a value, the operating system user name is used.

Note

If Security Services single sign-on (SSO) is enabled for your server, the **-u** and **-p** arguments are authenticated externally through SSO rather than being authenticated against the Teamcenter database. If you do not supply these arguments, the utility attempts to join an existing SSO session. If no session is found, you are prompted to enter a user ID and password.

-p

Specifies the password.

If used without a value, the system assumes a null value. This argument is mutually exclusive with the **-pf** argument.

If this argument is not used, the system assumes the *user-ID* value to be the password.

-pf

Specifies the password file. If used without a value, the system assumes a null value. If this argument is not used, the system assumes the *user-ID* value to be the password.

For more information about managing password files, see [Manage password files](#).

This argument is mutually exclusive with the **-p** argument.

-g

Specifies the group associated with the user.

If used without a value, the user's default group is assumed.

-o

Specifies overwrite mode. Default value is **on**.

-f

Displays a help message about the input file format.

-c

Specifies the item class name to create. Specify either **Item** or **Architecture**.

-t

Specifies the default item type to create.

-v
Displays verbose information.

-i
Specifies the full path to an input file.

-h
Displays help for this utility.

ENVIRONMENT
As specified in [Manually configuring your environment for Teamcenter utilities](#).

FILES
As specified in [Log files produced by Teamcenter](#).

RESTRICTIONS
The default mode, **overwrite**, replaces any existing structures with the information contained in the input file.

**INPUT FILE
FORMAT**
The input file consists of lines of comments, directives, and items. Each item and key line defines an item that the **ps_upload** utility creates. The comments and directive lines only alter the input file parsing behavior. They have no effect on the created items.

The structure hierarchy is determined by the level column (where the top level is level zero). The textual indentation of the item ID in the example file is only to make it more readable. Each time a level zero item is created, a new structure is started. Therefore, many structures (including single items) can be created from a single input file.

The top-level item in each structure is added to the user's **Newstuff** folder.

Note Do not show more than one occurrence of the same expanded assembly or you duplicate its contents.

The **ps_upload** utility assigns **A** as the initial revision ID.

With the exception of lines beginning with **#DELIMITER**, **#SUB_DELIMITER**, or **#COL**, the pound sign (#) in the first column indicates a comment that is ignored. Completely blank lines are also ignored.

#DELIMITER *x* Specifies the delimiter. Default value is a space.

#SUB_DELIMITER *x* Specifies the sub-delimiter. Default value is a semicolon ;.

#COL Specifies the column heading order.
The format is **#COL** *field field field*
The default value is:

item arch_element_id name level seq occs qty uom sub

key Specifies the key. To find the key of an object, use the [get_key_string](#) utility.

For more information, see the [Business Modeler IDE Guide](#).

item Specifies the item ID.

rev Specifies a revision letter other than the default. The default is **A**.

Note You can create multiple revisions of the same item. Do this by first creating the item without specifying a level so that the item is not placed into any structure. Enter a line for each revision required. The **rev** column can then be left blank in the structure lines for items already created.

arch_element_id Specifies the architecture element ID.

option Specifies an option and set of allowed values to be defined and attached to the item being created by that line. The format is:

Option-name;Value;Value...

Note The delimiter must be that defined for substitutes (**#SUB_DELIMITER** directive). In the previous example, the delimiter is a semicolon (;).

loadif Specifies a simple variant condition using an option that has been defined in the previously created items. The variant condition is limited to one option/value expression. The format is:

Owning-ItemID;Option-name== or != Value

Note The delimiter must be that defined for substitutes (**#SUB_DELIMITER** directive). In the previous example, the delimiter was a semicolon (;).

name Specifies the item name.

revname Specifies the item revision name.

level Specifies the structural hierarchy.

seq Specifies the find number of the item in the BOM view.

occs Specifies the number of occurrences of the item.

qty Specifies the quantity of an item in the structure.

uom Specifies the symbol representing the unit of measure.

sub Specifies substitutes.

type Specifies the item type. Note that the type must already exist in the database.

occname Specifies the occurrence path name.

loadifkey Specifies the key.

Substitutes

An entry in the substitutes column should consist of a delimiter-separated list of item IDs, where each substitute has been individually defined as a **level 0** structure. The default column delimiter is a semicolon (;).

The **#SUB_DELIMITER** directive tells the system that the next nonspace character on the line is used as a delimiter. If there is no nonspace character after the **#DELIMITER** directive, the delimiter is set to a blank space (' '). The substitute delimiter cannot be the same as the column delimiter.

EXAMPLES

The following figure illustrates the general file layout and shows the arbitrary use of the **#COL** and **#DELIMITER** directives:

```
# Example File for ps_upload
# The product structure for a bicycle.
# Change the delimiter to a comma, so we can use spaces in the name
#DELIMITER ,
# We start off with the default column order, and define a few simple
# parts that can be used as substitutes later. Note that these parts do not
# have a Level defined,
# and so will not be part of any structure.
#   item          name          Level Seq   Occs   Qty   Uom Sub
#   b100,         Bolt type 100
#   b101,         Bolt type 101
# Now we get on to the main structure
#   b001,         bicycle,      0,
#   b002,         frame,        1,   10
#   b003,         26" Wheel,    1,   20,
#   b004,         Metal Spoke,  2,   10,   20
#   b005,         bolt,         2,   20,   -,   2,   -,   b100 ; b101
#   b003,         26" Wheel,    1,   30
# Change the column order to show how its done!
#COL   Level Seq item      Uom Name          Occs   Sub Qty
#       1,   40, b007,     -,   Handlebars Assy, 2
#       2,   10, b008,     -,   Brake Level Assy, 2
#       2,   20, b009,     -,   Grips, 2
#       2,   30, b010,     -,   Handlebar Frame,
# Change Substitute Delimiter to /
#SUB_DELIMITER /
#       2,   40, b005,     -,   bolt, -,   b101 / b100, 2
#       1,   50, b011,     -,   Saddle,
# Change Delimiter to allow commas in the name
#DELIMITER $
#       1$   60$ 1001$    m1$ Oil, lubricating$ -$ -$ 100
#       1$   70$ 1002$    m$  Paint, red$ -$ -$ 2.4
-----
```

Product structure input file

purge_baselined_item_revisions

Purges baseline revisions when the automatic purge process fails.

SYNTAX

```
purge_baselined_item_revisions [-u=user-id -p=password | -pf=password-file
-g=group]
[-item_id=item_id
| -key=[keyAttr1=keyVal1] [,keyAttr2=keyVal2]...[,keyAttrN=keyValN]
-status=release-status [-date=yyyy MM dd hh mm ss | now | today] [-h]
```

ARGUMENTS

-u

Specifies the user ID.

This is generally **infodba** or another user with administration privileges. If this argument is used without a value, the operating system user name is used.

Note

If Security Services single sign-on (SSO) is enabled for your server, the **-u** and **-p** arguments are authenticated externally through SSO rather than being authenticated against the Teamcenter database. If you do not supply these arguments, the utility attempts to join an existing SSO session. If no session is found, you are prompted to enter a user ID and password.

-p

Specifies the password.

If used without a value, the system assumes a null value. This argument is mutually exclusive with the **-pf** argument.

If this argument is not used, the system assumes the *user-ID* value to be the password.

-pf

Specifies the password file. If used without a value, the system assumes a null value. If this argument is not used, the system assumes the *user-ID* value to be the password.

For more information about managing password files, see [Manage password files](#).

This argument is mutually exclusive with the **-p** argument.

-g

Specifies the group associated with the user.

If used without a value, the user's default group is assumed.

-item_id

Specifies the appearance root item.

-key

Specifies the key of the appearance root item. Use the following format:

```
[keyAttr1=keyVal1] [,keyAttr2=keyVal2]...[,keyAttrN=keyValN]
```

To find the key of an object, use the [get_key_string](#) utility.

For more information, see the [Business Modeler IDE Guide](#).

-status

Purges baselined item revisions of the status specified by this argument.

-date

Purges baselined item revisions created before the date/time specified by this argument.

-h

Displays help for this utility.

ENVIRONMENT

As specified in [*Manually configuring your environment for Teamcenter utilities.*](#)

FILES

None.

RESTRICTIONS

None.

EXAMPLES

None.

qsearch_process_queue

Updates or queries the spatial indexes used by the cacheless search mechanism. You can also use this utility to modify or query the state of the update queue process that updates these indexes.

SYNTAX

```
qsearch_process_queue [-u=user-id -p=password | -pf=password-file -g=group]
{[-list_queue | -list_all_queue]} [-show_queue_oldest_date]
[-process_queue] [-process_queue_repeatedly [-delay=N] [-repeat=M]]
[-clear_queue] [-force_queue_update Objects]
[-force_queue_substructure_update=Objects]
[-force_queue_all_leaf_item_updates] [-force_queue_all_possible_updates]
[-force_queue_all_necessary_updates] [-force_queue_all_inconsistent_updates]
[-tolerance=Percentage] [-ask_global_search_box_delta] [-list_volumes=Objects]
[-list_index_boxes=Objects] [-list_structure_index_boxes=Objects]
[-list_all_index_boxes]
[-clear_indexes] [-clear_structure_indexes=Objects] [-clear_all_indexes]
[-clear_queue_processed] [-clear_all_queue]
[-check_indexes=Objects]
[-check_structure_indexes=Objects]
[-list_suggested_updates[=filename] | -force_queue_suggested_updates |
-follow_only_check_failures] [-find_cycles]
[-count_occurrences=Objects] [-count_substructure=Objects]
[-task=task-list] [-verbose] [-print_names] [-h]
```

ARGUMENTS

-u

Specifies the user ID.

This is generally **infodba** or another user with administration privileges. If this argument is used without a value, the operating system user name is used.

Note

If Security Services single sign-on (SSO) is enabled for your server, the **-u** and **-p** arguments are authenticated externally through SSO rather than being authenticated against the Teamcenter database. If you do not supply these arguments, the utility attempts to join an existing SSO session. If no session is found, you are prompted to enter a user ID and password.

-p

Specifies the password.

If used without a value, the system assumes a null value. This argument is mutually exclusive with the **-pf** argument.

If this argument is not used, the system assumes the *user-ID* value to be the password.

-pf

Specifies the password file. If used without a value, the system assumes a null value. If this argument is not used, the system assumes the *user-ID* value to be the password.

For more information about managing password files, see [Manage password files](#).

This argument is mutually exclusive with the **-p** argument.

-g

Specifies the group associated with the user.

If used without a value, the user's default group is assumed.

-list_queue

Lists all unprocessed entries in the queue.

-list_all_queue

Lists all entries in the queue, including processed entries.

-show_queue_oldest_date

Shows the creation date of the oldest unprocessed entry in the queue.

-process_queue

Processes the unprocessed entries in the queue.

-process_queue_repeatedly

Processes the unprocessed entries in the queue repeatedly. Optionally, you can wait *N* seconds between process runs (default delay is 5 seconds). You can also specify a maximum of *M* times for the process to repeat (default is forever).

-delay

Specifies a delay of *N* seconds between process runs.

-repeat

Specifies a maximum of *M* times for the process to repeat (default is forever).

-clear_queue

Clears the unprocessed entries from the queue.

-force_queue_update

Adds an entry to the queue for the specified objects.

-force_queue_substructure_update

Adds a entry to the queue for the leaf items of the assembly beneath the specified objects. This action updates the entire assembly.

-force_queue_all_leaf_item_updates

Adds an entry to the queue for all leaf items. This action updates all assemblies.

-force_queue_all_possible_updates

Adds an entry to the queue for the primary of each **TC_bounding_box** relation.

-force_queue_all_necessary_updates

Adds an entry to the queue for the primary of each **TC_bounding_box** relation that lacks an index.

-force_queue_all_inconsistent_updates

Adds an entry to the queue for the primary of each **TC_bounding_box** relation with an apparently inconsistent index. You can optionally specify the percentage inconsistency to ignore, overriding the default value of 10%.

-tolerance

Specifies the percentage of inconsistency to ignore.

-ask_global_search_box_delta

Calculates the current global search box delta.

-list_volumes

For each specified object, lists the total volume occupied by all its contributing bounding-boxes and the total volume of all its index-boxes.

-list_index_boxes

Lists the index boxes for the specified objects.

-list_structure_index_boxes

Lists the index boxes for all possible configured structures for the given objects.

-list_all_index_boxes

Lists the index boxes for all objects.

-clear_indexes

Removes the indexes from the specified objects.

-clear_structure_indexes

Removes the indexes for all possible configured structures from the specified objects.

-clear_all_indexes

Removes the indexes from all objects.

-clear_queue_processed

Clears the processed entries from the queue.

-clear_all_queue

Clears all entries from the queue, that is, both processed and unprocessed entries.

-check_indexes

Checks the indexes for the specified objects.

-check_structure_indexes

Checks the indexes for all possible configured structures for the specified objects.

- If you specify **-list_suggested_updates**, the utility lists the objects for should be updated to fix any incorrect indexes detected for the structure. The list is written to the specified file or **stdout** if no file is specified. The file is written in a format suitable for **qsearch_process_queue -force_queue_update -uid=@filename**.
- If you specify **-force_queue_suggested_updates**, the utility adds entries to the queue to fix any incorrect indexes detected for the structure.
- If you specify **-follow_only_check_failures**, the utility assumes that if the indexes of the specified object are correct, the indexes for the entire substructure are also correct.

-find_cycles

Finds all cyclical structures.

-count_occurrences

Counts all occurrences in all structures of all items. Optionally, you can just count all occurrences of specified objects.

-count_substructure

Counts the substructure of all root items. Optionally, you can just count the substructure of specified objects.

-task

Specifies a task list of multiple arguments. Omit the leading dashes and separate entries with commas. For example:

```
-task=list_queue,process_queue
```

-verbose

Runs the utility in verbose mode to display the maximum amount of information. Typically, nonverbose utility sessions only display error messages.

-print_names

Prints item IDs and names as well as UIDs.

-h

Displays help for this utility.

ENVIRONMENT

As specified in [Manually configuring your environment for Teamcenter utilities](#).

FILES

As specified in [Log files produced by Teamcenter](#).

RESTRICTIONS

Where appropriate, objects may be specified as a list of UIDs or a list of item ID patterns, as follows:

- A list of UIDs must be preceded by **-uid=** and entries separated with commas:

```
-uid=uid1,uid2,uid3,...
```

You can also supply the UIDs in a separate text file, with one UID per line. In this case, use the file name instead of the list of UIDs, preceding it with @:

```
-uid=@filename
```

- A list of item ID patterns must be preceded by **-item_id=** and entries separated with commas:

```
-item_id=item-id-pattern1,item-id-pattern2,item-id-pattern3,...
```

You can also supply the item ID patterns in a separate text file, with one item ID pattern per line. In this case, use the file name instead of the list of item ID patterns, preceding it with @:

```
-item_id=@filename
```

EXAMPLES

The following example lists all unprocessed entries in the queue:

```
qsearch_process_queue -u=infodba -p=infodba -g=dba -list_queue
```

The following example adds an entry to the queue for each of the objects listed in the **c:\temp\objfile.txt** file:

```
qsearch_process_queue -u=infodba -p=infodba -g=dba -force_queue_update  
-uid=@c:\temp\objfile.txt
```

The following example checks the indexes for all objects that have an item ID prefixed with **123**:

```
qsearch_process_queue -u=infodba -p=infodba -g=dba -check_indexes -item_id=123*
```

The following example lists the unprocessed entries in the queue and then processes them:

```
qsearch_process_queue -u=infodba -p=infodba -g=dba -task=list_queue,process_queue
```

RESOLVING CIRCULAR REFERENCES

Some assemblies may contain circular references (cycles). If cycles exist in an assembly, you may encounter poor performance when creating indexes and running searches. If so, use this utility with the **find_cycles** argument to identify all such cycles. You should then remove the cycles manually before executing other cacheless search utilities. You should also identify why these cycles are created and establish a process to prevent the creation of more cycles.

You can run the utility with the **find_cycles** argument against a single item or the entire database, as shown in the following examples.

Processing the entire database:

```
qsearch_process_queue -u=infodba -p=infodba -g=dba -find_cycles 2>
c:\temp\list_of_cycles.txt
```

Processing a single item:

```
qsearch_process_queue -u=infodba -p=infodba -g=dba -find_cycles
-item_id=AKT75443
```

Note The utility output is written to standard error and you should redirect it to a file. For example, specify `2< c:\temp\list_of_cycles.txt`, as shown in the first example.

The output format is similar to the following example:

```
Cycle begins =====
Y31VRf_fQyYmUC AKY43846-LV2.X 1C PROFILE SECTIONS
YW7VRf_fQyYmUC AKY43841-LV2.X 1C Y_SECTIONAL VIEW
i70lXf0fgzknZC ATR04589-T04A WS LWR AT CL
h38NawpPQyYmUC AKT75443-TOTAL VEHICLE
i01NNpCxQyYmUC AKU25433-VEHICLE INTEGRATION MODULE
ii2NNpCxQyYmUC AKU25441-COMPARTMENT VEHICLE
yP2Vwa5kQyYmUC AKY08647-LV2.X HYB CDH 1B GA
XFwVwmjAQyYmUC AKY10317-LV2.X CDH STRUCTURE REF ASM
LZ8Vw$FeQyYmUC AKY27566-LV2.X HYB CDH MIKE FRAEYMAN PART
iiMl7dfgzknZC ATR05553-IA-CDH-REAR COMPT-MM
VH1VB_7gQyYmUC AKY41691-CDH RR RAIL U/B SI
aC6VRf_fQyYmUC AKY43866-LV2.X HYB CDH 1C GA

Cycle ends =====
```

To report cycles found when you execute an attribute search in Structure Manager or DesignContext, set the **QS_START_CIRCULAR_STRUCTURE_CHECK_SIZE** and **QSEARCH_DEBUG_TEXT** user preferences to 1.

The output report format in this mode is as follows:

```
>>>> Found cyclic structure.UnconfiguredStep: STUDY MODULE ( 68342) - 0 -- 0 - 76915
(VPPS 20 STUDIES CHASSIS) ( zMMNRX46gzknZC) -- 77314 (E1JNRX46gzknZC) -- 0 --- 75941
(STUDY-GMT172/7 SPARE TIRE) ( SQ$Jb7D_QyYmUC)-- 76845 ( zc6J7309QyYmUC) -- 0 --- 73558
(TOTAL VEHICLE) (xTw01mOJgzknZC) -- 75523 ( jh3Jd9saQyYmUC) -- 0 --- 70375
(STUDY MODULE) ( RXw01mOJgzknZC) -- 71949 ( RuEFkrRSgzknZC) -- 0 --- 68342
(STUDY MODULE) ( TEHNRX46gzknZC) -- 69364 ( U1ONRX46gzknZC) -- 0 --- 67615
```

```
(VPPS 40 STUDIES INTERIOR) ( zkMNRX46gzknZC) -- 67831 (ExNNRX46gzknZC) -- 0 --- 65940
(STUDY GMT172/177 FD SPEAKER PKG) ( THyJbqzAQyYmUC) -- 66630 (DVzNwd24QyYmUC) -- 0 --- 63228
(IA-FRT S/D HARDWARE LH 1&2LA00) (jEDBpNkogzknZC) -- 64916 ( 1t4JbqzAQyYmUC) -- 0 --- 61430
(BOLT) ( wT1FNGA$QyYmUC) -- 62442 ( RdHFgSL$gzknZC) -- 0 ---
```

To clean up the cycles, search for the associated item in My Teamcenter. Do not perform a where used search or browse in Structure Manager because the applied revision rule prevents you from finding the items. When you find the item, send the associated BVR to Structure Manager and manually remove the cycle. In the previous example, look in the My Teamcenter **View** folder for every revision of item **AKY43866-LV2.X HYB CDH 1C GA** and remove the cycles in Structure Manager as you find them.

update_bomchanges

Updates BOM change records corresponding to all existing change objects. Only users with **DBA** permissions can run this utility.

The engineering change objects to be updated are selected based on the values provided by the **-c** argument.

A log file containing the list of engineering changes and the corresponding affected assemblies that have been successfully updated is created. The log file can be provided using the **-l** option. If no log file is provided the information is published to the standard output.

SYNTAX

```
[-u=user-id -p=password | -pf=password-file -g=group]
-c={0 | 1 | 2} -e=change-id -key=change-key -l=log-file [-h]
```

ARGUMENTS

-u

Specifies the user ID.

This is generally **infodba** or another user with administration privileges. If this argument is used without a value, the operating system user name is used.

Note If Security Services single sign-on (SSO) is enabled for your server, the **-u** and **-p** arguments are authenticated externally through SSO rather than being authenticated against the Teamcenter database. If you do not supply these arguments, the utility attempts to join an existing SSO session. If no session is found, you are prompted to enter a user ID and password.

-p

Specifies the password.

If used without a value, the system assumes a null value. This argument is mutually exclusive with the **-pf** argument.

If this argument is not used, the system assumes the *user-ID* value to be the password.

-pf

Specifies the password file. If used without a value, the system assumes a null value. If this argument is not used, the system assumes the *user-ID* value to be the password.

For more information about managing password files, see [Manage password files](#).

This argument is mutually exclusive with the **-p** argument.

-g

Specifies the group associated with the user.

If used without a value, the user's default group is assumed.

-c

Determines the category of engineering changes for which the affected assemblies must be updated.

0

Only updates released engineering change objects.

1

Only updates released and in process engineering change objects.

2

Updates all the engineering change objects in the database.

-e

Specifies the change ID, if any.

-key

Specifies the key of the change object whose revisions need to be updated. Either the **-c**, **-e**, or **-key** argument must be provided

The key represents the multifield key of the object. To find the key of an object, use the [get_key_string](#) utility.

For more information, see the [Business Modeler IDE Guide](#).

-l

Creates a log file containing the list of engineering changes and the corresponding affected assemblies that have been successfully updated. If no log file is provided, the information is published to the standard output.

-h

Displays help for this utility.

ENVIRONMENT

As specified in [Manually configuring your environment for Teamcenter utilities](#).

FILES

Log file contains the list of engineering changes and the affected assemblies that have been update successfully.

RESTRICTIONS

None.

EXAMPLES

- The following example updates the BOM changes associated with the affected assembly of all the released engineering changes in the database. It also generates a log file named **log.txt** at the location **/tmp/**. The log file contains the list of engineering changes and the affected assemblies that update successfully:

```
$TC_BIN/update_bomchanges -u=infodba -p=infodba -g=dba  
-c=0 -l=/tmp/log.txt
```

- The following example updates the BOM changes associated with the affected assembly of all the engineering changes in the database. The information pertaining to the list of engineering changes and the affected assemblies that update successfully is published to the standard output:

```
$TC_BIN/update_bomchanges -u=infodba -p=infodba -g=dba -c=2
```

update_loggeddatetogmt

DESCRIPTION

Run this utility after you migrate Teamcenter from a version prior to 8.3 to version 8.3 or later. This utility updates the values in the **LoggedDate** column from the local time zone to the GMT time zone.

This utility does a bulk update for Oracle and SQL Server databases. DB2 database is updated record by record.

Note Run this utility only once. If you run this utility multiple times, the time zone correction is applied multiple times.

This utility may take a long time to run depending on your data size.

SYNTAX

update_loggeddatetogmt [-u=*user-id* {-p=*password* | -pf=*password-file*} -g=*group*] [-h]

ARGUMENTS

-u

Specifies the user ID.

This is generally **infodba** or another user with administration privileges. If this argument is used without a value, the operating system user name is used.

Note If Security Services single sign-on (SSO) is enabled for your server, the **-u** and **-p** arguments are authenticated externally through SSO rather than being authenticated against the Teamcenter database. If you do not supply these arguments, the utility attempts to join an existing SSO session. If no session is found, you are prompted to enter a user ID and password.

-p

Specifies the password.

If used without a value, the system assumes a null value.

If this argument is not used, the system assumes the *user-ID* value to be the password.

This argument cannot be replaced with the **-pf** argument.

-pf

Specifies the password file. If used without a value, the system assumes a null value. If this argument is not used, the system assumes the *user-ID* value to be the password.

For more information about managing password files, see [Manage password files](#).

This argument is mutually exclusive with the **-p** argument.

-g

Specifies the group associated with the user.

If used without a value, the user's default group is assumed.

-h

Displays help for this utility.

update_objecttype utility

Run this utility after you migrate Teamcenter from a version prior to 8.3 to version 8.3 or later. This utility updates the **ObjectType** column with the correct Teamcenter object type. The default object type is **POM_object**.

Note This utility may take a long time to run depending on your data size.

Running this utility is optional.

SYNTAX

update_objecttype [-u=*user-id* {-p=*password* | -pf=*password-file*} -g=*group*]
[-h]

ARGUMENTS

-u

Specifies the user ID.

This is generally **infodba** or another user with administration privileges. If this argument is used without a value, the operating system user name is used.

Note If Security Services single sign-on (SSO) is enabled for your server, the **-u** and **-p** arguments are authenticated externally through SSO rather than being authenticated against the Teamcenter database. If you do not supply these arguments, the utility attempts to join an existing SSO session. If no session is found, you are prompted to enter a user ID and password.

-p

Specifies the password.

If used without a value, the system assumes a null value.

If this argument is not used, the system assumes the *user-ID* value to be the password.

This argument cannot be replaced with the **-pf** argument.

-pf

Specifies the password file. If used without a value, the system assumes a null value. If this argument is not used, the system assumes the *user-ID* value to be the password.

For more information about managing password files, see [Manage password files](#).

This argument is mutually exclusive with the **-p** argument.

-g

Specifies the group associated with the user.

If used without a value, the user's default group is assumed.

-h

Displays help for this utility.

update_project_bom

Allows you to update all items in a BOM structure within specified projects.

SYNTAX

update_project_bom [-u=*user-id* -p=*password* | -pf=*password-file* -g=*group*]
[-f={**add** | **remove**}] [-type={*item* | *rev*}] [-item=*item-id* | -key=*key-id*]
[-rev_id=*revision-id*]
[-rev_rule=*revision-rule*] [-unit_no=*unit-number*] [-date=*date*]
[-end_item=*end-item-id* | -end_key=*end-key-id*] [-var_rule=*variant-rule*]
[-depth=*depth-of-bom*]
[-level={ 1 | 2 }] [-projects=*project-lists*] [-h]

ARGUMENTS**-u**

Specifies the user ID.

This is generally **infodba** or another user with administration privileges. If this argument is used without a value, the operating system user name is used.

Note

If Security Services single sign-on (SSO) is enabled for your server, the **-u** and **-p** arguments are authenticated externally through SSO rather than being authenticated against the Teamcenter database. If you do not supply these arguments, the utility attempts to join an existing SSO session. If no session is found, you are prompted to enter a user ID and password.

-p

Specifies the password.

If used without a value, the system assumes a null value. This argument is mutually exclusive with the **-pf** argument.

If this argument is not used, the system assumes the *user-ID* value to be the password.

-pf

Specifies the password file. If used without a value, the system assumes a null value. If this argument is not used, the system assumes the *user-ID* value to be the password.

For more information about managing password files, see [Manage password files](#).

This argument is mutually exclusive with the **-p** argument.

-g

Specifies the group associated with the user.

If used without a value, the user's default group is assumed.

-f

Specifies one of the following types of operation for this utility:

Note

If this argument is omitted, the default action is to add items to projects.

add

Adds item objects to projects.

remove

Removes item objects from projects.

-type

Specifies one of the following types to be updated for each BOM line:

item

Updates each child item for the projects.

rev

Updates only child item revision objects to the projects.

Note

If this argument is omitted, the default type is **item**.

-item

Specifies the ID of the root item of the BOM to update. Use either the **-item** argument or the **-key** argument.

-key

Specifies the key of the root item of the BOM to update. Use either the **-item** argument or the **-key** argument.

To find the key of an object, use the [get_key_string](#) utility.

For more information, see the [Business Modeler IDE Guide](#).

-rev_id

Specifies the ID of the root item revision. If omitted, the default is the latest revision.

-rev_rule

Specifies the configuration rule to be applied to the item revision. If omitted, the default is the **Latest Working** revision rule.

-unit_no

Specifies the unit number associated with the revision rule.

-date

Specifies the effectivity date associated with the revision rule. The date should be specified in the following format:

yyyy MM dd hh mm ss

-end_item

Specifies the ID of the end item associated with the revision rule. Use with the **-item** argument.

-end_key

Specifies the key of the end item associated with the revision rule. Use with the **-key** argument.

To find the key of an object, use the [get_key_string](#) utility.

For more information, see the [Business Modeler IDE Guide](#).

-var_rule

Specifies the variant rule to be applied to the BOM structure. If omitted, no variant rule is applied.

-depth

Specifies to what depth the BOM is traversed. If omitted, the entire BOM structure is traversed.

-level

Indicates the level of propagation. Specify either **1** or **2**.

-projects

Lists the projects to which the BOM structure will be added or from which the BOM structure will be removed.

When the **-add** option is specified, all items in the BOM structure are added to these projects.

When the **-remove** option is specified, all items currently belonging to the projects are removed from the projects. You can specify more than one project. If there is more than one project in the list, each project name is separated by a comma (,).

-h

Displays help for this utility.

RESTRICTIONS

None.

EXAMPLES

- To display usage help for this utility, enter the following command on a single line:

```
update_project_bom -h
```

- To traverse a BOM structure with the top-level item **ABC001**, item revision **001**, and revision rule **Latest Working**, and to find all child items and add these items to the following three projects: **CusProj1**, **CusProj2**, and **CusProj3**, enter the following command on a single line:

```
update_project_bom -u=user -p=password -g=dba  
-f=add -item=ABC001 -rev_id=001 -rev_rule="Latest Working"  
-projects=CusProj1,CusProj2,CusProj3
```

- To traverse a BOM structure with the top-level item **ABC001**, item revision **001**, and revision rule **Latest Working**, and to find all the child revision items and add only these revision items to projects **CusProj1** and **CusProj2**, enter the following command on a single line:

```
update_project_bom -u=user -p=password -g=dba  
-f=add -type=rev -item=ABC001 -rev_id=001 -rev_rule="Latest Working"  
-projects=CusProj1,CusProj2
```

- To traverse the BOM structure starting at the top-level item **ABC001** with item revision **001** by applying the revision rule **Latest Released** and variant rule **AlphaRelease**, and find all the child revision items and add only these revision items to the projects **CusProj1** and **CusProj2**, enter the following command on a single line:

```
update_project_bom -u=user -p=password -g=dba  
-f=add -type=rev -item=ABC001 -rev_id=001 -rev_rule="Latest Released"  
-var_rule="AlphaRelease" -projects=CusProj1,CusProj2
```

- To traverse the BOM structure of the top-level item **ABC002**, item revision **001**, and revision rule **Latest Working** and find all the child items and remove these

items from projects **CusProj1** and **CusProj2**, enter the following command on a single line:

```
update_project_bom -u=user -p=password -g=dba
-f=remove -item=ABC002 -rev_id=001 -rev_rule="Latest Working"
-projects=CusProj1,CusProj2
```

- To traverse the BOM structure of the top-level item **ABC002**, item revision **001**, and revision rule **Latest working** and find all the child revision items and remove only these revision items from projects **CusProj1** and **CusProj2**, enter the following command on a single line:

```
update_project_bom -u=user -p=password -g=dba
-f=remove -type=rev -item=ABC002 -rev_id=001
-rev_rule="Latest Working" -projects=CusProj1,CusProj2
```

- To traverse the BOM structure in the top three levels with the top-level item **ABC002**, item revision **001**, and revision rule **Latest working**, and find all the child revision items in the top three levels and remove only these revision items from the projects **CusProj1** and **CusProj2**, enter the following command on a single line:

```
update_project_bom -u=user -p=password -g=dba
-f=remove -type=item -item=ABC002 -rev_id=001
-rev_rule="Latest Working" -projects=CusProj1,CusProj2 -depth=3
```

- To traverse the BOM structure with the top-level item **ABC001** and perform:
 - o Level 1 propagation: locate all the child items in the BOM based on item revision **001**, revision rule **Latest working**, and include these items into the **CusProj1**, **CusProj2**, and **CusProj3** projects.
 - o Level 2 propagation: no level 2 propagation.

```
update_project_bom -u=user -p=password -g=dba
-item=ABC001 -rev_id=001 -rev_rule="Latest Working"
-level=1 -projects=CusProj1,CusProj2,CusProj3
```

- To traverse the BOM structure with the top-level item **ABC001** and perform:
 - o Level 1 propagation: locate all the child items in the BOM based on item revision **001**, revision rule **Latest working**, and include these items into the **CusProj1**, **CusProj2**, and **CusProj3** projects.
 - o Level 2 propagation: collect all dataset type objects attached to the BOM line during the BOM traversal. Recursively find all objects that relate to the dataset type objects through the relation specified in the [TC_project_propagate_from_dataset](#) preference and propagate all of these level 2 objects into the **CusProj1**, **CusProj2**, and **CusProj3** projects.

```
update_project_bom -u=user -p=password -g=dba
-item=ABC001 -rev_id=001 -rev_rule="Latest Working" -level=2
-projects=CusProj1,CusProj2,CusProj3
```

- To traverse the BOM structure with the top-level item **ABC001** and perform:
 - o Level 1 propagation: traverse the BOM based on item revision **001**, revision rule **Latest working**, and variant rule **AlphaRelease**, find all the child revision items and include only these revision items into the **CusProj1**, **CusProj2**, and **CusProj3** projects.

- o Level 2 propagation: collect all dataset type objects attached to the revision items in the BOM structure during the level 1 propagation. Recursively locate all objects that relate to the dataset type objects through the relation specified in the **TC_project_propagate_from_dataset** preference and propagate all of these level 2 dependencies into the **CusProj1**, **CusProj2**, and **CusProj3** projects.

```
update_project_bom -u=user -p=password -g=dba  
-item=ABC001 -rev_id=001 -rev_rule="Latest Working"  
-var_rule="AlphaRelease" -level=2  
-projects=CusProj1,CusProj2,CusProj3
```

upgrade_rev_rules

Upgrades revision rules so that they can safely use the modified revision rule implementation. This is run automatically as part of the upgrade script, but may need to be run again if any locked revision rules are encountered (or if any other error occurs).

SYNTAX

upgrade_rev_rules [-u=*user-id* -p=*password* | -pf=*password-file*
-g=*group*][-v][-h]

ARGUMENTS

-u

Specifies the user ID.

This is generally **infodba** or another user with administration privileges. If this argument is used without a value, the operating system user name is used.

Note

If Security Services single sign-on (SSO) is enabled for your server, the **-u** and **-p** arguments are authenticated externally through SSO rather than being authenticated against the Teamcenter database. If you do not supply these arguments, the utility attempts to join an existing SSO session. If no session is found, you are prompted to enter a user ID and password.

-p

Specifies the password.

If used without a value, the system assumes a null value. This argument is mutually exclusive with the **-pf** argument.

If this argument is not used, the system assumes the *user-ID* value to be the password.

-pf

Specifies the password file. If used without a value, the system assumes a null value. If this argument is not used, the system assumes the *user-ID* value to be the password.

For more information about managing password files, see [Manage password files](#).

This argument is mutually exclusive with the **-p** argument.

-g

Specifies the group associated with the user.

If used without a value, the user's default group is assumed.

-v

Verbose mode. Shows additional messages.

-h

Displays help for this utility.

ENVIRONMENT

As specified in [Manually configuring your environment for Teamcenter utilities](#).

FILES

As specified in [Log files produced by Teamcenter](#).

RESTRICTIONS

None.

EXAMPLES

None.

upgrade_variants

Use the **upgrade_variants** utility to upgrade an existing Teamcenter system to the new variant model. It allows you to:

- Migrate the variants for one program (product item) at a time or migrate the entire database.
- Migrate only selected variant conditions and named variant expressions.
- Migrate the design structure or the architecture structure.
- Perform a dry run to validate data integrity before migrating production data.
- Generate a report on all named variant expressions that were migrated.

SYNTAX

upgrade_variants [-u=*user-id* -p=*password* | -pf=*password-file* -g=*group*]
 -product_code=*product code* -ves_list_file=*variant expression uid list file*
 -top_arch_id=*top level architecture item* -full_database=*yes* | *no*
 -lou_holder_id=*item id of lou holder* -file_path=*path name*
 -dry_run=*yes* | *no* [-long] -report_delete_failures=*yes* | *no*
 [-force] -delete_unreferenced_ves_only=*yes* | *no* -delete_tree=*yes* | *no*
 -bulk_delete=*yes* | *no* [-parallel] [-h]

ARGUMENTS

Note Entries in parentheses are accepted abbreviations for arguments.

-u

Specifies the user ID.

This is generally **infodba** or another user with administration privileges. If this argument is used without a value, the operating system user name is used.

Note If Security Services single sign-on (SSO) is enabled for your server, the **-u** and **-p** arguments are authenticated externally through SSO rather than being authenticated against the database. If you do not supply these arguments, the utility attempts to join an existing SSO session. If no session is found, you are prompted to enter a user ID and password.

-p

Specifies the password.

If used without a value, the system assumes a null value. This argument is mutually exclusive with the **-pf** argument.

If this argument is not used, the system assumes the *user-ID* value to be the password.

-pf

Specifies the password file. If used without a value, the system assumes a null value. If this argument is not used, the system assumes the *user-ID* value to be the password.

For more information about managing password files, see [Manage password files](#).

This argument is mutually exclusive with the **-p** argument.

-g

Specifies the group associated with the user.

If used without a value, the user's default group is assumed.

-product_code

Specifies the product code from the product definition form.

-ves_list_file

Specifies a text file containing the UIDs of the variant expressions to migrate.

-top_arch_id

Specifies the item ID of the top level assembly breakdown structure. The named variant expressions for the architecture breakdown are migrated.

-full_database

Specify **yes** to migrate all the variant conditions and NVEs in the entire database or **no** to migrate selected data. The default is **no**.

-lou_holder_id

Specifies the item ID of the LOU holder item. The variant conditions for the LOUs under this LOU holder are migrated.

-file_path

Specifies the path where the migration report should be stored. If no path is specified, the report is stored in the path identified in the **TC_TEMP_DIR** environment variable.

-dry_run

If specified as **yes**, the utility performs a dry run. A dry run does not migrate the variant expressions but generate a migration report. You must specifically set this argument to **no** if you do *not* want to perform a dry run.

-long

Optionally used when you perform a dry run. If specified, the utility outputs UIDs in addition to the variant expression text and named variant expression names.

-report_delete_failures

If specified as **yes**, reports failures encountered while deleting the variant expressions that are not successfully migrated. The default value is **no**.

-force

Optionally forces conversion of all variant expressions to the new variant model.

-delete_unreferenced_ves_only

Optionally deletes unreferenced variant expressions from the database. The default value is **no**.

-delete_tree

Optionally skips deletion of payload trees. The default value is **no**.

-bulk_delete

Optionally performs a bulk deletion of all unreferenced variant expressions. The default value is **no**.

-parallel

Optionally specifies a parallel variant model upgrade session when a session is already in progress.

-h

Displays help for this utility.

ENVIRONMENT

This utility should be run in a shell where the Teamcenter environment is set up.

FILES

A report file is generated in the specified location.

RESTRICTIONS

None.

EXAMPLES

1. To migrate only the assembly structure:

```
upgrade_variants -u=infodba -p=infodba -g=dba -product_code=PLM00001  
-file_path=c:\report.txt -dry_run=no
```

2. To migrate the assembly structure and architecture breakdown (NVEs and LOUs):

```
upgrade_variants -u=infodba -p=infodba -g=dba -product_code=PLM00001  
-top_arch_id=PLM0002 -lou_holder_id=PLM00003 -file_path=c:\report.txt -dry_run=no
```

3. To migrate only LOUs in the assembly breakdown:

```
upgrade_variants -u=infodba -p=infodba -g=dba -lou_holder_id =GMO00003  
-file_path=c:\report.txt -dry_run=no
```

4. To create a variant expressions list file:

```
upgrade_variants -u=infodba -p=infodba -g=dba -ves_list_file=c:\vesList.txt  
-file_path=c:\report.txt -dry_run=no
```

5. To perform a dry run of a full database migration:

```
upgrade_variants -u=infodba -p=infodba -g=dba -full_database=yes  
-file_path=c:\report.txt -dry_run=yes
```

Effectivity mode

effupgrade

Converts effectivity data created in iMAN versions prior to 7.0 into the effectivity model used in iMAN version 7.0 and later, Teamcenter Engineering and Teamcenter. The new effectivity model allows end item qualification and discontinuous ranges. The upgrade process goes through each unconverted release status and creates a 7.0 effectivity qualified against a null end item from the start date, end date or unit values on the release status.

SYNTAX

effupgrade [-u=*user-id* -p=*password* | -pf=*password-file* -g=*group*]
[-s] | [-i=*interval*] [-v [-e]] [-h]

ARGUMENTS**-u**

Specifies the user ID.

This is generally **infodba** or another user with administration privileges. If this argument is used without a value, the operating system user name is used.

Note

If Security Services single sign-on (SSO) is enabled for your server, the **-u** and **-p** arguments are authenticated externally through SSO rather than being authenticated against the Teamcenter database. If you do not supply these arguments, the utility attempts to join an existing SSO session. If no session is found, you are prompted to enter a user ID and password.

-p

Specifies the password.

If used without a value, the system assumes a null value. This argument is mutually exclusive with the **-pf** argument.

If this argument is not used, the system assumes the *user-ID* value to be the password.

-pf

Specifies the password file. If used without a value, the system assumes a null value. If this argument is not used, the system assumes the *user-ID* value to be the password.

For more information about managing password files, see [Manage password files](#).

This argument is mutually exclusive with the **-p** argument.

-g

Specifies the group associated with the user.

If used without a value, the user's default group is assumed.

-s

Specifies a single run. Upgrade runs once and ignores locked statuses.

-i

Specifies multiple runs and an interval between reruns in minutes. The default interval is 60 minutes.

-v

Verbose mode. Shows additional messages.

-e

Displays effectivities of release statuses to be converted. Must be used with the **-v** argument.

-h

Displays help for this utility.

ENVIRONMENT

As specified in [Manually configuring your environment for Teamcenter utilities](#).

FILES

As specified in [Log files produced by Teamcenter](#).

RESTRICTIONS

None.

EXAMPLES

- For cases where an upgrade is not going to be disruptive (less than 1000 effectivities), make sure all users are logged off, then upgrade with the **-s** argument. For example:

```
effupgrade -u=infodba -p=password -g=dba -s -v
```
- For extensive upgrades that are likely to take a long time (many thousands of effectivities), especially if statuses are likely to be locked (some users always logged on), run repetitively until complete.

```
effupgrade -u=infodba -p=password -g=dba
```

Product structure clearance analysis

batchmode_clearance_analysis

Performs clearance analysis on a product structure and stores all issue and results data in the clearance database. This utility can be run as a CRON job and can be run at different levels using the run level switch.

SYNTAX

TC_ROOT/clearanceDB/scripts/ [-u=*user-id* -p=*password* |
-pf=*password-file* -g=*group*] -bookmark [-h]

ARGUMENTS**-u**

Specifies the user ID.

This is generally **infodba** or another user with administration privileges. If this argument is used without a value, the operating system user name is used.

Note

If Security Services single sign-on (SSO) is enabled for your server, the **-u** and **-p** arguments are authenticated externally through SSO rather than being authenticated against the Teamcenter database. If you do not supply these arguments, the utility attempts to join an existing SSO session. If no session is found, you are prompted to enter a user ID and password.

-p

Specifies the password.

If used without a value, the system assumes a null value. This argument is mutually exclusive with the **-pf** argument.

If this argument is not used, the system assumes the *user-ID* value to be the password.

-pf

Specifies the password file. If used without a value, the system assumes a null value. If this argument is not used, the system assumes the *user-ID* value to be the password.

For more information about managing password files, see [Manage password files](#).

This argument is mutually exclusive with the **-p** argument.

-g

Specifies the group associated with the user.

If used without a value, the user's default group is assumed.

-bookmark

Generates a bookmark file of the product.

-h

Displays help for this utility.

ENVIRONMENT

As specified in [Manually configuring your environment for Teamcenter utilities](#) and in the setup document provided in the **TC_ROOT/clearanceDB/scripts** directory.

FILES

None.

EXAMPLES

To perform clearance analysis on a product structure and store the results data in the clearance database, enter the following command on a single line:

```
batchmode_clearance_analysis.pl
```

Appearance Configuration

apn_medic

Checks for redundant appearance path nodes (APNs).

SYNTAX

```
apn_medic [-u=user-id {-p=password | -pf=password-file} -g=group]  
-all {-item=item-id | [-key=[keyAttr1=keyVal1] [,keyAttr2=keyVal2]...  
[,keyAttrN=keyValN]}}  
-apn=APN-uid -repair -force=option -output=output-file  
[-h]
```

ARGUMENTS**-u**

Specifies the user ID.

This is generally **infodba** or another user with administration privileges. If this argument is used without a value, the operating system user name is used.

Note If Security Services single sign-on (SSO) is enabled for your server, the user and password arguments are authenticated externally through SSO rather than being authenticated against the Teamcenter database. If you do not supply these arguments, the utility attempts to join an existing SSO session. If no session is found, you are prompted to enter a user ID and password.

-p

Specifies the user's password.

If used without a value, the system assumes a null value. If this argument is not used, the system assumes the *user-ID* value to be the password.

This argument is mutually exclusive with the **-pf** argument.

-pf

Specifies the password file. If used without a value, the system assumes a null value. If this argument is not used, the system assumes the *user-ID* value to be the password.

For more information about managing password files, see [Manage password files](#).

This argument is mutually exclusive with the **-p** argument.

-g

Specifies the group associated with the user.

If used without a value, the user's default group is assumed.

-all

Traverses the APN structure to find all redundant APNs. If not specified, the utility only checks if there are any redundant root APNs.

-item

Starts the utility using the item as the root of the structure. If not specified, the utility searches the entire database.

-key

Specifies the key of the item to track. Use the following format:


```
[keyAttr1=keyVal1] [,keyAttr2=keyVal2]...[,keyAttrN=keyValN]
```

To find the key of an object, use the **get_key_string** utility.

For more information, see the [Business Modeler IDE Guide](#).

-apn

Specifies the UID of an APN. Once used, the **-item** argument is ignored. If the **-all** argument is also specified, the utility gets the root and purges the whole APN structure where the specific APN resides. If the **-all** argument is not specified, only the APN is purged.

Note This argument repairs the structure regardless of whether the **-repair** argument is specified.

-repair

Picks one APN among the duplicate APNs, copies GRM relations to the selected APN from the non-surviving nodes, and purges the children and the redundant nodes.

-force

Forces deletion of duplicate APNs. This argument works in conjunction with the **-repair** argument.

- Deletes duplicate APNs whether they have **absOccData** objects or not.
- Deletes all replica duplicate APNs in a multi-site environment.
- Deletes all duplicate APNs including master APNs.

-output

Specifies the output file containing the information for analyze. If not specified, the default **apn_medic_output.txt** file is used. The following information is placed in the output file: the UID of the redundant APN, the associated item, the parent item, the occurrence thread, the appearance path root, the number of children, the number of relations of the node and the associated absolute occurrence, number of app refs, and the type of the absolute occurrence data.

-h

Displays help for this utility.

ENVIRONMENT

As specified in [Manually configuring your environment for Teamcenter utilities](#).

FILES

As specified in [Log files produced by Teamcenter](#).

RESTRICTIONS

None.

appr_update_manager

Launches the Appearance Update Manager. Sets up the Teamcenter runtime environment and launches the Update Manager Supervisor (**appr_update_supervisor**) utility. Also automatically relaunches the supervisor when required.

SYNTAX

appr_update_manager

ARGUMENTS

None.

ENVIRONMENT

As specified in *[Manually configuring your environment for Teamcenter utilities](#)*.

RESTRICTIONS

None.

EXAMPLES

None.

appr_update_supervisor

Runs as a background process, invoking Teamcenter processes to perform appearance updates, as required.

Note Siemens PLM Software recommends that you use the **appr_update_manager** utility to run this program rather than running it directly.

Because the supervisor is not a Teamcenter process, it does not require a user name or password. However, the processes it spawns do require user name and password; therefore, you should run the program as a privileged user and allow the update processes to log on automatically. If autologin is not supported at your site, you can use the alternative log on mechanism documented in the **\$TC_ROOT/data/appr_update_env.default** file.

SYNTAX

appr_update_supervisor

ARGUMENTS

None.

ENVIRONMENT

As specified in *Manually configuring your environment for Teamcenter utilities*.

FILES

- **\$TC_ROOT/bin/.appr_update_env**
Configuration file for the Appearance Update Manager.
- **\$TC_ROOT/bin/.appr_update_env.default**
Example configuration file.
- **\$TC_ROOT/bin/appr_update_supervisor*.log**
Supervisor log file.

RESTRICTIONS

None.

EXAMPLES

None.

appr_update_console

Enables users to query the status of the Update Manager and allows administrators to control the Update Manager. Starting the console with no query or control arguments on the command line starts a simple menu system.

SYNTAX

```
appr_update_console [-u=user-id -p=password | -pf=password-file -g=group]  
[-host=host]  
[-port=port]  
[-menu]  
[-query]  
[-query=update-UID]  
[-dump_primary]  
[-dump_secondary]  
[-show_blocked]  
[-hide_blocked]  
[-log_status]  
[-log_file]  
[-log_level=n]  
[-clear_log]  
[-shutdown]  
[-shutdown_now]  
[-restart]  
[-restart_now]  
[-prod_queue]  
[-h]
```

ARGUMENTS**-u**

Specifies the user ID.

This is generally **infodba** or another user with administration privileges. If this argument is used without a value, the operating system user name is used.

Note

If Security Services single sign-on (SSO) is enabled for your server, the **-u** and **-p** arguments are authenticated externally through SSO rather than being authenticated against the Teamcenter database. If you do not supply these arguments, the utility attempts to join an existing SSO session. If no session is found, you are prompted to enter a user ID and password.

-p

Specifies the password.

If used without a value, the system assumes a null value. This argument is mutually exclusive with the **-pf** argument.

If this argument is not used, the system assumes the *user-ID* value to be the password.

-pf

Specifies the password file. If used without a value, the system assumes a null value. If this argument is not used, the system assumes the *user-ID* value to be the password.

For more information about managing password files, see [Manage password files](#).

This argument is mutually exclusive with the **-p** argument.

-g

Specifies the group associated with the user.

If used without a value, the user's default group is assumed.

-host

Connects to the supervisor on the specified host. Overrides the setting in the **.appr_update_config** file.

-port

Connects to the supervisor through the specified communications port. Overrides the setting in the **.appr_update_config** file.

-menu

Forces the console into interactive mode.

-query

Queries the current Update Manager status.

-query=update UID

Queries the status of a specific update package.

-dump_primary

Prints the primary queue.

-dump_secondary

Prints the secondary queue.

-show_blocked

Shows blocked packages in subsequent queue dumps.

-hide_blocked

Hides blocked packages in subsequent queue dumps.

-log_status

Requests that the Update Manager write a complete status report to the log file.

-log_file

Queries the name of the Update Manager log file.

-log_level

Sets the supervisor logging level. This command is available only to system administrators.

-clear_log

Requests that the Update Manager clear the log file. This command is available only to system administrators.

-shutdown

Requests that the Update Manager shut down as soon as the current task is finished.

-shutdown_now

Requests that the Update Manager shut down immediately. This command is available only to system administrators.

-restart

Requests that the Update Manager restart immediately. This command is available only to system administrators.

-restart_now

Restarts update manager immediately. This command is available only to system administrators.

-prod_queue

Sends the supervisor a dummy update request.

-h

Displays help for this utility.

ENVIRONMENT

As specified in [Manually configuring your environment for Teamcenter utilities](#).

FILES

As specified in [Log files produced by Teamcenter](#) along with the following files:

- **\$TC_ROOT/bin/.appr_update_env**
Configuration file for the Appearance Update Manager.
- **\$TC_ROOT/ bin/.appr_update_env.default**
Sample configuration file.

RESTRICTIONS

None.

EXAMPLES

1. To start the Update Manager console menu system, enter the following command at the prompt:

```
appr_update_console
```

2. To start the Update Manager console system in administration mode, enter the following command at the prompt:

```
appr_update_console -u=infodba -p=password -g=dba
```

3. To query the status and the log file name of the Update Manager on the **my_server.mycompany.com** host, enter the following command at the prompt:

```
appr_update_console -host=my_server.my_company.com -query -log_file
```

appearance_updater

Processes appearance updates as part of the Appearance Update Manager. This program should be invoked by the **appr_update_supervisor** program.

SYNTAX

```
appearance_updater [-u=user-id -p=password | -pf=password-file -g=group]
[-quiet]
[-nolog]
[-supervisor=host,port1,port2]
[-task=cmd [,cmd...]]
[-show_uids]
[-show_blocked]
[-since=date -item_id=item-id] ]
[-at=date]
[-h]
```

ARGUMENTS

-u

Specifies the user ID.

This is generally **infodba** or another user with administration privileges. If this argument is used without a value, the operating system user name is used.

Note

If Security Services single sign-on (SSO) is enabled for your server, the **-u** and **-p** arguments are authenticated externally through SSO rather than being authenticated against the Teamcenter database. If you do not supply these arguments, the utility attempts to join an existing SSO session. If no session is found, you are prompted to enter a user ID and password.

-p

Specifies the password.

If used without a value, the system assumes a null value. This argument is mutually exclusive with the **-pf** argument.

If this argument is not used, the system assumes the *user-ID* value to be the password.

-pf

Specifies the password file. If used without a value, the system assumes a null value. If this argument is not used, the system assumes the *user-ID* value to be the password.

For more information about managing password files, see [Manage password files](#).

This argument is mutually exclusive with the **-p** argument.

-g

Specifies the group associated with the user.

If used without a value, the user's default group is assumed.

-login

Forces the program to search for the **.appr_update_info** file in the **\$TC_DATA**, **\$TC_ROOT/data**, or **\$HOME** directory. The **.appr_update_info** file must contain

the **-u**, **-p**, and **-g** flags (on separate lines) that are used to log on. This is used primarily by the Update Manager Supervisor.

-supervisor

Identifies the Update Manger Supervisor that supplies tasks for the update process. This is used primarily by the Update Manager Supervisor.

-task

Manually identifies a task, or list of tasks, to be performed by the update process. The task list is a comma-separated list (no spaces) containing one or more of the following tasks:

query_update	Queries the POM UID of the current appearance update.
query_primary_size	Queries the size of the primary appearance update queue.
query_sets	Queries the number of appearance sets, number of processed sets, and number of sets yet to be processed (excluding those currently being processed) for the update.
query_packages	Queries the number of update packages in the current update process.
query_unprocessed_packages	Queries the number of update packages in the current update that have not yet been processed.
process_primary	Requests the processing of the next update in the primary queue. This fails if there is already an update in process.
process_secondary	Requests the processing of a single appearance set for the current update. This fails if there are no appearance sets that require processing.
process_all_secondary	Requests that the program loops until all appearance sets are processed for the current update.
finish_primary	Completes the final processing of an update once all secondary sets have been processed.
dump_primary	Dumps information about the primary queue to standard output (stdout).
dump_secondary	Dumps information about the secondary queue to standard output (stdout).
-quiet	Suppresses the output of diagnostics to standard output (stdout).
-nolog	Suppresses diagnostics to the log file.
-show_uids	Includes tag/UID information.

-show_blocked

Includes information about blocked, as well as unblocked packages.

-since= *yyyy mm dd hh mm ss*

Includes information about all packages, processed or unprocessed, since the specified date.

-item_id

Shows only those packages relevant to the release of any revision of the specified item. This argument is only supported for use when the **-since** argument is used.

-at= *yyyy mm dd hh mm ss*

Shows the package that was running on the specified date. This is assumed to be the earliest package with order-by and run dates that straddle the specified date.

-h

Displays help for this utility.

ENVIRONMENT

As specified in [Manually configuring your environment for Teamcenter utilities](#).

FILES

As specified in [Log files produced by Teamcenter](#).

RESTRICTIONS

None.

EXAMPLES

- Enter the following command at the prompt to query the size of an appearance update queue:

```
appearance_updater -task=query_primary_size
```
- Enter the following command at the prompt to query the UID and state of the current update:

```
appearance_updater -task=query_update,query_set
```
- Enter the following command at the prompt to process the next complete update in the queue:

```
appearance_update -task=process_primary,process_all,  
process_al_secondary,finish_primary
```

appr_working_scheduler

Provides an update scheduler for *working appearances* (working appearances is a collective name for all appearance sets that have a **working** entry in their context revision rule).

This utility places the **ApprUpdWorkingPkg** appearance working package on the primary update queue at every user-specified time interval. The ensuing execution of the working appearance package by the appearance supervisor ensures that the appearance working cache is updated. If an existing working update package is already in the updater queue in an unprocessed or processing state, a new package is not created. The scheduler waits until the end of the next elapsed interval before it attempts to put another package on the updater queue.

SYNTAX

appr_working_scheduler [-u=*user-id* -p=*password* | -pf=*password-file* -g=*group*]
-minutes=*minutes-time-interval* -hours=*hour-time-interval* [-h]

ARGUMENTS

-u

Specifies the user ID.

This is generally **infodba** or another user with administration privileges. If this argument is used without a value, the operating system user name is used.

Note

If Security Services single sign-on (SSO) is enabled for your server, the **-u** and **-p** arguments are authenticated externally through SSO rather than being authenticated against the Teamcenter database. If you do not supply these arguments, the utility attempts to join an existing SSO session. If no session is found, you are prompted to enter a user ID and password.

-p

Specifies the password.

If used without a value, the system assumes a null value. This argument is mutually exclusive with the **-pf** argument.

If this argument is not used, the system assumes the *user-ID* value to be the password.

-pf

Specifies the password file. If used without a value, the system assumes a null value. If this argument is not used, the system assumes the *user-ID* value to be the password.

For more information about managing password files, see [Manage password files](#).

This argument is mutually exclusive with the **-p** argument.

-g

Specifies the group associated with the user.

If used without a value, the user's default group is assumed.

-minutes

Specifies time interval, in minutes, at which the scheduler creates a working appearance update package. This value is added to the time interval specified by the **-hours** argument.

-hours

Specifies time interval, in hours, at which the scheduler creates a working appearance update package. This value is added to the time interval specified by the **-minutes** argument.

-h

Displays help for this utility.

ENVIRONMENT

As specified in [Manually configuring your environment for Teamcenter utilities](#) and the [Appearance Configuration Guide](#).

FILES

As specified in [Log files produced by Teamcenter](#).

RESTRICTIONS

- Appearances environment must be setup prior to using this utility.
For more information, see the [Appearance Configuration Guide](#).
- The appearance supervisor must be up and running, or in exceptional circumstances for troubleshooting, explicitly disabled by the administrator by setting **supervisor.enabled** to **0** in the **.appr_update_env** file.

EXAMPLES

- The following example creates an update package every 5 minutes:
`appr_working_scheduler -u=infodba -p=infodba -g=dba -minutes=5`
- The following example creates an update package every 2 hours 30 minutes:
`appr_working_scheduler -u=infodba -p=infodba -g=dba -hours=2 -minutes=30`
- The following example immediately creates one package in the updater queue. This task is achieved when neither the **hours** or the **minutes** option is specified.

```
appr_working_scheduler -u=infodba -p=infodba -g=dba
```

- Executing the examples above generates the following messages on the appearance update supervisor console window:

```
Logging in...
Logged in, waiting for instructions
Performing general query
no update in progress
primary size = 1
No primary in process
  Sets: 0, Done: 0, Unallocated: 0
Selecting next primary update
Checking for update in progress
no update in progress
Looking for next update
Attempting to lock primary
Locked successfully
Selected primary update hGAVPMUCAAgcRA
ApprUpdWorkingPkg: queued on 08-Mar-2007 16:06:43, (no run date),
unprocessed (0), unblocked
```

```
(no Release Status)
(no cloned-for AppearanceRoot)
0 secondary package(s)
Processing primary update
Attempting to process primary
Processed primary to secondary
Listing secondaries
Secondaries count = 0
Performing general queue query
primary size = 0
Listing secondaries
Secondaries count = 0
Sets: 0, Done: 0, Unallocated: 0
Current primary update appears to be complete
Primary marked as complete
```

create_appearances

Creates appearances representing an initial product structure in a particular context. It also creates an appearance root, if required.

SYNTAX

```
create_appearances [-u=user-id -p=password | -pf=password-file -g=group]
{[-item_id=item-id | [-key=[keyAttr1=keyVal1] [,keyAttr2=keyVal2]...
[,keyAttrN=keyValN]]} -config_rule=config-rule -view_type=view-type
[-in_date=in-date] [-out_date=out-date]
[-in_unit_no=in-unit-no] [-out_unit_no=out-unit-no] [-h]
```

ARGUMENTS

-u

Specifies the user ID.

This is generally **infodba** or another user with administration privileges. If this argument is used without a value, the operating system user name is used.

Note

If Security Services single sign-on (SSO) is enabled for your server, the **-u** and **-p** arguments are authenticated externally through SSO rather than being authenticated against the Teamcenter database. If you do not supply these arguments, the utility attempts to join an existing SSO session. If no session is found, you are prompted to enter a user ID and password.

-p

Specifies the password.

If used without a value, the system assumes a null value. This argument is mutually exclusive with the **-pf** argument.

If this argument is not used, the system assumes the *user-ID* value to be the password.

-pf

Specifies the password file. If used without a value, the system assumes a null value. If this argument is not used, the system assumes the *user-ID* value to be the password.

For more information about managing password files, see [Manage password files](#).

This argument is mutually exclusive with the **-p** argument.

-g

Specifies the group associated with the user.

If used without a value, the user's default group is assumed.

-item_id

Specifies the item to track.

-key

Specifies the key of the item to track. Use the following format:

```
[keyAttr1=keyVal1] [,keyAttr2=keyVal2]...[,keyAttrN=keyValN]
```

To find the key of an object, use the [get_key_string](#) utility.

For more information, see the [Business Modeler IDE Guide](#).

-config_rule

Specifies the revision rule to use.

-view_type

Specifies the view to use.

-h

Displays help for this utility.

ENVIRONMENT

As specified in [Manually configuring your environment for Teamcenter utilities](#).

FILES

None.

RESTRICTIONS

None.

EXAMPLES

None.

fix_appearances

Rolls back and deletes appearances to the last point at which the appearance root was marked as consistent, that is, the **corruption_status** attribute was **0**. It also requeues the relevant update packages. When the packages are reprocessed, Teamcenter creates a new set of appearances that more accurately represent the actual structure. This is effectively a partial rollback and does not force recreation of the complete appearance set.

Note An error displays if the input values result in more than one item being found in the database.

SYNTAX

```
fix_appearances [-u=user-id -p=password | -pf=password-file -g=group]
[-item_id=item-id]
[-key=[keyAttr1=keyVal1] [,keyAttr2=keyVal2]...[,keyAttrN=keyValN]]
[-config_rule=config-rule] [-view_type=view-type]
[-ok_date=ok-date] -force [-v] [-h]
```

ARGUMENTS

-u

Specifies the user ID.

This is generally **infodba** or another user with administration privileges. If this argument is used without a value, the operating system user name is used.

Note If Security Services single sign-on (SSO) is enabled for your server, the **-u** and **-p** arguments are authenticated externally through SSO rather than being authenticated against the Teamcenter database. If you do not supply these arguments, the utility attempts to join an existing SSO session. If no session is found, you are prompted to enter a user ID and password.

-p

Specifies the password.

If used without a value, the system assumes a null value. This argument is mutually exclusive with the **-pf** argument.

If this argument is not used, the system assumes the *user-ID* value to be the password.

-pf

Specifies the password file. If used without a value, the system assumes a null value. If this argument is not used, the system assumes the *user-ID* value to be the password.

For more information about managing password files, see [Manage password files](#).

This argument is mutually exclusive with the **-p** argument.

-g

Specifies the group associated with the user.

If used without a value, the user's default group is assumed.

-item_id

Specifies an item ID for filtering appearance roots.

-key

Specifies the key for filtering appearance roots. Use the following format:

```
[keyAttr1=keyVal1] [,keyAttr2=keyVal2]...[,keyAttrN=keyValN]
```

To find the key of an object, use the [get_key_string](#) utility.

For more information, see the [Business Modeler IDE Guide](#).

-config_rule

Specifies a configuration rule for filtering appearance roots.

-view_type

Specifies a view type for filtering appearance roots.

Note

You use the **item_id**, **config_rule**, and **view_type** arguments as filters to determine which appearance roots to fix.

-ok_date

Assert the date when there was no corruption thereby overriding the appearance root's stored **ok_date**. The date format is *yyyy MM dd hh mm ss*. This argument is optional.

By default, the **fix_appearances** utility uses the date of the **ok_date** attribute of the appearance root, that is, the date at which the last successful automatic validation occurred. If the validation never occurred, the value of **ok_date** is the same as the appearance root's creation date.

However, you can specify a date for roll back with the **ok_date** argument. This date can be earlier than the last **ok-date** on the appearance root. If you run the [find_appearances](#) utility and set **-verbose** and **-item_id=ID** arguments, the utility dumps the last **ok-date** of the appearance root. You can then use this date and run the **fix_appearances** utility with the **ok-date** value immediately before it.

-force

Force a fix even if appearance root determines it is not corrupt. Use this argument if there are differences between the automatic and the manual checking processes. An error message informs the user whether to use the **-force** argument.

-v

Specifies verbose mode.

-h

Displays help for this utility.

ENVIRONMENT

As specified in [Manually configuring your environment for Teamcenter utilities](#).

FILES

As specified in [Log files produced by Teamcenter](#).

RESTRICTIONS

None.

EXAMPLES

The following example shows an invocation of the **fix_appearances** utility:


```

$TC_BIN/fix_appearances -force -ok_date=2002 -item_id=0150_app_01
-view=view -config_rule=0150_app_context -u=infodba -p=infodba"
Found 1 AppearanceRoot
AppearanceRoot[0] = 00003baf = hdHVNDf_AAgcRA (Item ID: 0150_app_01 (View: view)
Revision Rule: 0150_app_context (without spatial data, not for pre-release,
active, never fixed, never checked, last updated 21-Feb-2007 09:26:31,
with 6 appearances)), ok at 2007-02-21 09:24:00
At 21-Feb-2007 09:27:25, re-queued cloned primary package:
(00003f5e = h5LVNHQhAAgcRA) ApprUpdChangedRevsPkg: queued on
21-Feb-2007 06:35:58, (no run date), unprocessed (0),
Release Status: (00003cd7 = x1BV96Y4AAgcRA) Production on 21-Feb-2007
06:35:58 by possum (possum) 0 Effectivities
Cloned-for AppearanceRoot: (00003baf = hdHVNDf_AAgcRA)
Item ID: 0150_app_01 (View: view)
Revision Rule: 0150_app_context (without spatial data, not for pre-release,
active, never fixed, never checked, never updated, with 0 appearances)
0 secondary package(s)
0 filtered-by AppearanceRoot(s)
1 changed ItemRevision(s):
(00003cd9 = h9JV96Y4AAgcRA) 0170_01/B-Car owned by possum (possum)
At 21-Feb-2007 09:27:25, re-queued cloned primary package:
(00003f61 = h5PVNHQhAAgcRA) ApprUpdChangedRevsPkg: queued on
21-Feb-2007 06:36:29, (no run date), unprocessed (0),
Release Status: (00003cda = ByKV96Y4AAgcRA) Production on 21-Feb-2007
06:36:29 by possum (possum) 0 Effectivities
Cloned-for AppearanceRoot: (00003baf = hdHVNDf_AAgcRA) Item ID: 0150_app_01
(View: view)
Revision Rule: 0150_app_context (without spatial data, not for pre-release,
active, never fixed, never checked, never updated, with 0 appearances)
0 secondary package(s)
0 filtered-by AppearanceRoot(s)
1 changed ItemRevision(s):
(00003cdb = BMWV96Y4AAgcRA) 0170_02/B-Axle owned by possum (possum)
At 21-Feb-2007 09:27:25, re-queued cloned primary package:
(00003f64 = h9DVNHQhAAgcRA) ApprUpdChangedRevsPkg: queued on
21-Feb-2007 06:37:31, (no run date), unprocessed (0),
Release Status: (00003cdd = xxGV96IqAAgcRA) Production on 21-Feb-2007
06:37:31 by possum (possum) 0 Effectivities
Cloned-for AppearanceRoot: (00003baf = hdHVNDf_AAgcRA) Item ID: 0150_app_01
(View: view)
Revision Rule: 0150_app_context (without spatial data, not for pre-release,
active, never fixed, never checked, never updated, with 0 appearances)
0 secondary package(s)
0 filtered-by AppearanceRoot(s)
3 changed ItemRevision(s):
(00003cde = hxMV96IqAAgcRA) 0180_01/B-Car owned by possum (possum)
(00003cdf = htBV96IqAAgcRA) 0180_02/A-Axle owned by possum (possum)
(00003ce0 = hxDV96IqAAgcRA) 0180_03/A-Wheel owned by possum (possum)
At 21-Feb-2007 09:27:25, re-queued cloned primary package:
(00003f67 = h9HVNQhAAgcRA) ApprUpdChangedRevsPkg: queued on
21-Feb-2007 06:39:47, (no run date), unprocessed (0),
Release Status: (00003ce3 = xxNV964zAAgcRA) Manufacture on 21-Feb-2007
06:39:47 by possum (possum) 0 Effectivities
Cloned-for AppearanceRoot: (00003baf = hdHVNDf_AAgcRA) Item ID: 0150_app_01
(View: view)
Revision Rule: 0150_app_context (without spatial data, not for pre-release,
active, never fixed, never checked, never updated, with 0 appearances)
0 secondary package(s)
0 filtered-by AppearanceRoot(s)
3 changed ItemRevision(s):
(00003ce4 = xFKV964zAAgcRA) 0200_03/A-0200 1-1 owned by possum (possum)
(00003ce5 = xBIV964zAAgcRA) 0200_02/A-0200 1 owned by possum (possum)
(00003ce6 = h5HV964zAAgcRA) 0200_01/B-0200 tracked item owned by possum (possum)

```

update_apprpathroot

Creates or updates the appearance path root, or removes the links between appearance path nodes and appearances.

An appearance path maps an occurrence to the corresponding BOM line. It is unique to the context of a specific BOM; different BOMs cannot contain the same appearance paths. An appearance path is also called an appearance path node (APN) or an occurrence path.

SYNTAX

update_apprpathroot [-u=*user-id* {-p=*password* | -pf=*password-file*} -g=*group*]
[-item=*item-id* | -key=*key-id*]
-task=*action*
[-h]

ARGUMENTS**-u**

Specifies the user ID.

This is generally **infodba** or another user with administration privileges. If this argument is used without a value, the operating system user name is used.

Note

If Security Services single sign-on (SSO) is enabled for your server, the user and password arguments are authenticated externally through SSO rather than being authenticated against the Teamcenter database. If you do not supply these arguments, the utility attempts to join an existing SSO session. If no session is found, you are prompted to enter a user ID and password.

-p

Specifies the user's password.

If used without a value, the system assumes a null value. If this argument is not used, the system assumes the *user-ID* value to be the password.

This argument is mutually exclusive with the **-pf** argument.

-pf

Specifies the password file. If used without a value, the system assumes a null value. If this argument is not used, the system assumes the *user-ID* value to be the password.

For more information about managing password files, see [Manage password files](#).

This argument is mutually exclusive with the **-p** argument.

-g

Specifies the group associated with the user.

If used without a value, the user's default group is assumed.

-item

Specifies the item to be used as the path root.

-key

Specifies the key of the item to be used as the path root. You can use the **-key** argument instead of the **-item** argument.

Use the following format:

```
[keyAttr1=keyVal1] [,keyAttr2=keyVal2]...[,keyAttrN=keyValN]
```

To find the key of an object, use the [get_key_string](#) utility.

For more information, see the [Business Modeler IDE Guide](#).

-task

Specifies the task to run:

update

Creates or updates the appearance path root.

remove

Removes the links between appearance path nodes and appearances.

-h

Displays help for this utility.

ENVIRONMENT

As specified in [Manually configuring your environment for Teamcenter utilities](#).

FILES

As specified in [Log files produced by Teamcenter](#).

RESTRICTIONS

None.

Chapter

4 Teamcenter Rapid Start utilities

smartuibldr_configure

Imports, exports, and updates the Smart Codes configuration text file to and from the Teamcenter Rapid Start database.

SYNTAX

```
smartuibldr_configure -imp=filename [-u=user-id {-p=password |  
-pf=password-file} -g=group] [-j] [-h]  
smartuibldr_configure -exp=filename [-u=user-id {-p=password |  
-pf=password-file} -g=group] [-j] [-h]
```

Note You can only export the file after it has already been imported.

ARGUMENTS**-imp**

Specifies the name of the text file to import.

-exp

Specifies the name of the text file to export.

-u

Specifies the user ID.

This is generally **infodba** or another user with administration privileges. If this argument is used without a value, the operating system user name is used.

Note If Security Services single sign-on (SSO) is enabled for your server, the **-u** and **-p** arguments are authenticated externally through SSO rather than being authenticated against the Teamcenter database. If you do not supply these arguments, the utility attempts to join an existing SSO session. If no session is found, you are prompted to enter a user ID and password.

-p

Specifies the password.

If used without a value, the system assumes a null value.

If this argument is not used, the system assumes the *user-id* value to be the password.

This argument is mutually exclusive with the **-pf** argument.

-pf

Specifies the password file. If used without a value, the system assumes a null value. If this argument is not used, the system assumes the *user-id* value to be the password.

For more information about managing password files, see [Manage password files](#).

This argument is mutually exclusive with the **-p** argument.

-g

Specifies the group associated with the user.

If used without a value, the user's default group is assumed.

-h

Displays help for this utility.

-J

Specifies whether the journaling option is enabled or disabled. To enable journaling, set the value to **On**. To disable journaling, set the value to **Off**.

ENVIRONMENT

This utility must be run in the Teamcenter Rapid Start shell environment.

FILES

As specified in *[Log files produced by Teamcenter](#)*.

RESTRICTIONS

This utility can be run only by users with Teamcenter Rapid Start administrator privileges.

EXAMPLES

None.

Chapter

5 *Collaborative Product
Development utilities*

4gd_populate_cd utility

Generates a collaborative design from an existing product or assembly structure. The collaborative design has the following characteristics:

- A name derived from the name of the source product or assembly.
- A set of physical partitions and optional corresponding functional partitions.
- One or more reuse design elements with names derived from the names of the items from which the reuse design elements are realized.

Transformation data is read from each level of the original product or assembly. The utility concatenates this data and uses it for design element to collaborative design transformations. Consequently, the collaborative design appears the same as the original assembly. If the original assembly includes absolute occurrences, it uses the absolute occurrence transformations. If the source item revision has bounding boxes attached, the same bounding boxes are attached to the new design element.

You can realize an entire assembly as a single reuse design element or, by default, realize each leaf of the assembly as an individual reuse design element. With the exception of the top-level node, a partition is created for all intermediate nodes in the assembly tree. Optionally, you can also create a partition for the top-level node.

By default, the utility configures the assembly with the **Latest Working** revision rule and the latest revision of the source assembly item. Optionally, you can specify a particular revision rule and a item revision.

The name assigned to the collaborative design is the name of the assembly with **_CD** appended to it. The name of a reuse design element is the name of the source item with **_DE_**. An index is appended to the name when leaves are realized as reuse elements. The index differentiates between leaves when the same component appears as a leaf more than once in the source assembly.

The utility places the new collaborative design in the **Newstuff** folder of the specified user.

SYNTAX

```
4gd_populate_cd [-u=user-id {-p=password | -pf=password-file}
-g=group] -i=item_id [-rev=revision] [-rule=config_rule] [-norelease]
[-root_partition=yes | no] [-top] -partition_names=assy | auto | name
| desc | uid [-add_functional_partitions] [-partition_types=types]
[-effectivity_qualification=unit_numbers] [-h]
```

ARGUMENTS

-u

Specifies the user ID.

This is generally **infodba** or another user with administration privileges. If this argument is used without a value, the operating system user name is used.

Note If Security Services single sign-on (SSO) is enabled for your server, the user and password arguments are authenticated externally through SSO rather than being authenticated against the Teamcenter database. If you do not supply these arguments, the utility attempts to join an existing SSO session. If no session is found, you are prompted to enter a user ID and password.

-p

Specifies the password.

If used without a value, the system assumes a null value. This argument is mutually exclusive with the **-pf** argument.

If this argument is not used, the system assumes the *user-id* value to be the password.

-pf

Specifies the password file. If used without a value, the system assumes a null value. If this argument is not used, the system assumes the *user-id* value to be the password.

For more information about managing password files, see [Manage password files](#).

This argument is mutually exclusive with the **-p** argument.

-g

Specifies the group associated with the user.

If used without a value, the user's default group is assumed.

-i

Specifies the item ID of the product or assembly to realize.

-rev

Specifies the revision of the specified item.

-rule

Specifies the name of the revision rule to configure the source assembly. If the name of the rule contains spaces, you must enclose it in quotation mark, that is, "*rev-rule-name*".

-norelease

If specified, assumes a release status is assigned to all source items and does not assign a release status to realized objects.

-root_partition

If specified, a partition is created for the top node of the assembly.

Note This is a **yes** | **no** switch.

-top

If specified, the entire assembly is realized as a single reuse design element, rather than as individual leaves. Also, no partitions are created.

-partition_names

Specifies how partitions are named, as follows:

assy

If specified, the partition is given the same name as the corresponding BOM line in the source assembly.

auto

If specified, the partitions are automatically named **Partition1**, **Partition2**, and so on, in the order they are encountered.

name

If specified, the partition is named according to the source item revision and the revision ID.

desc

If specified, the partition is named according to the description of the source item revision.

uid

If specified, the partitions are named using the assembly option with the session UID appended.

If no argument is specified or an invalid argument is specified, the utility uses **assy**.

-add_functional_partitions

If specified, functional partitions are also created. Their contents are the same as the physical partitions.

-partition_types

If specified, partitions are created only for items of the specified type.

-effectivity_qualification

If specified, the new design elements and features are assigned unit effectivity from the specified unit or units.

-h

Displays help for this utility.

ENVIRONMENT

As specified in [Manually configuring your environment for Teamcenter utilities](#).

FILES

As specified in [Log files produced by Teamcenter](#).

Results are written to the **bom_output.txt** file in the **c:\temp** directory on Windows systems or **/tmp** directory on UNIX systems. It lists the shape and reuse elements that are created, together with the total number of objects created. It also contains the time of the last run and total memory consumed.

RESTRICTIONS

None.

EXAMPLES

```
4gd_populate_cd -u=adminjones -p=passjones -g=dba -i=engine -rev=B -rule="
latest released" -norelease -root_partition=yes -partition_names=auto
-add_functional_partitions -effectivity_qualification=unit1, unit2
```

This example creates a collaborative design from revision B of an item assembly called **engine**. Before conversion, the latest released revision rule is applied to the source assembly, but no release status is assigned during conversion. A root partition is created for the top node of the assembly, partition names are created automatically, and both physical and functional partitions are created. New design elements and features receive the unit effectivity of units 1 and 2.

manage_effectivity_options

Allows you to manage 4GD effectivity options from a command line.

SYNTAX

```
manage_effectivity_options [-u=user-id {-p=password |  
-pf=password-file} -g=group]  
-contextItemId=ItemID | -contextItemKey=Key -contextRevision=revisionID  
-effOpt=option-name-string  
[-effValues=valid-values]  
-owningItemId=ItemID | -owningItemKey=Key  
[-h]
```

ARGUMENTS

-u

Specifies the user ID.

This is generally **infodba** or another user with administration privileges. If this argument is used without a value, the operating system user name is used.

Note

If Security Services single sign-on (SSO) is enabled for your server, the user and password arguments are authenticated externally through SSO rather than being authenticated against the Teamcenter database. If you do not supply these arguments, the utility attempts to join an existing SSO session. If no session is found, you are prompted to enter a user ID and password.

-p

Specifies the password.

If used without a value, the system assumes a null value. This argument is mutually exclusive with the **-pf** argument.

If this argument is not used, the system assumes the *user-ID* value to be the password.

-pf

Specifies the password file. If used without a value, the system assumes a null value. If this argument is not used, the system assumes the *user-ID* value to be the password.

For more information about managing password files, see [Manage password files](#).

This argument is mutually exclusive with the **-p** argument.

-g

Specifies the group associated with the user.

If used without a value, the user's default group is assumed.

-contextItemId

Specifies the item ID to which to attach a new option. This is typically the item ID of the product for which the effectivity option should be added or modified. The effectivity option may be an effectivity stream group such as Engineering Intent or Model Year.

-contextItemKey

Specifies keys and values to pass if the item ID and revision ID are insufficient to uniquely identify the context item revision. You can specify **contextItemId** or **contextItemKey**, not both.

-contextRevision

Specifies the item revision for which a new effectivity option should be attached or modified. The item revision represents the product namespace in which the option will be available.

-effOpt

Identifies the effectivity option to create or modify. This option usually represents an effectivity stream grouping such as Engineering Intent or Model Year.

-effValues

Specifies a list of valid values for the effectivity option. This is an absolute list and *all* valid values must be passed. Teamcenter treats the absence of valid values when you update existing effectivity options as a request to remove those values from the context revision. Valid values must be given in the context of the context item revision. You can attach the modified item revision to multiple models for which effectivity streams with these valid values are needed.

-owningItemId

Specifies the item that defines the effectivity option to modify. If specified, the effectivity option defined on the owning item is reused on the product item. It is restricted to the set of specified values if **-effValues** is populated with a list of valid values.

-owningItemKey

Specifies keys and values to pass if the owning item ID is insufficient to uniquely identify the owning item. You can specify **owningItemId** or **owningItemKey**, not both.

-h

Displays help for this utility.

ENVIRONMENT

As specified in [Manually configuring your environment for Teamcenter utilities](#).

FILES

As specified in [Log files produced by Teamcenter](#).

RESTRICTIONS

None.

ptn0_persist_dynamicMembers

Traverses the complete partition breakdown of a collaborative design, executes every partition recipe, and creates static membership objects in the database. Traversal of a large product may take a significant time and Siemens PLM Software recommends you schedule the utility to execute periodically as a cron job. Execute it at a frequency that ensures the saved state of dynamic members is sufficiently up-to-date to produce reliable results from the **wherePartitioned** service (this service uses static membership objects to identify owning partitions for design elements). When you run this utility, it deletes membership objects it previously created and creates new membership objects, according to the latest recipes on the partitions.

SYNTAX

```
ptn0_persist_dynamicMembers [-u=user-id {-p=password | -pf=password-file}  
-g=group] -m=model_id -s=partition scheme type name [-h]
```

ARGUMENTS

-u

Specifies the user ID.

This is generally **infodba** or another user with administration privileges. If this argument is used without a value, the operating system user name is used.

Note

If Security Services single sign-on (SSO) is enabled for your server, the user and password arguments are authenticated externally through SSO rather than being authenticated against the Teamcenter database. If you do not supply these arguments, the utility attempts to join an existing SSO session. If no session is found, you are prompted to enter a user ID and password.

-p

Specifies the password.

If used without a value, the system assumes a null value. This argument is mutually exclusive with the **-pf** argument.

If this argument is not used, the system assumes the *user-id* value to be the password.

-pf

Specifies the password file. If used without a value, the system assumes a null value. If this argument is not used, the system assumes the *user-id* value to be the password.

For more information about managing password files, see [Manage password files](#).

This argument is mutually exclusive with the **-p** argument.

-g

Specifies the group associated with the user.

If used without a value, the user's default group is assumed.

-m

Specifies the identifier of the collaborative design (model) to traverse.

-s

Specifies the type name of the partition scheme, for example, **Ptn0SchemeFunctional**, **Ptn0SchemePhysical**, or **Ptn0SchemeSpatial**.

-h

Displays help for this utility.

RESTRICTIONS

This utility must be run by a user with DBA permissions, for example, **infodba**.

EXAMPLES

```
ptn0_persist_dynamicMembers -u=adminjones -p=passjones -g=dba -m=new_ship_design  
-s=Ptn0SchemeFunctional
```


Ptn0_set_is_partition_owned_true

Sets the **ptn0is_partition_owned** property on partition memberships, if you are upgrading to Teamcenter 10.1 or later from an earlier version. This property must be set if you want to export or import partition members using TC XML.

SYNTAX

Ptn0_set_is_partition_owned_true [-u=*user-id* {-p=*password* | -pf=*password-file*} -g=*group*] [-h]

ARGUMENTS

-u

Specifies the user ID.

This is generally **infodba** or another user with administration privileges. If this argument is used without a value, the operating system user name is used.

Note

If Security Services single sign-on (SSO) is enabled for your server, the user and password arguments are authenticated externally through SSO rather than being authenticated against the Teamcenter database. If you do not supply these arguments, the utility attempts to join an existing SSO session. If no session is found, you are prompted to enter a user ID and password.

-p

Specifies the password.

If used without a value, the system assumes a null value. This argument is mutually exclusive with the **-pf** argument.

If this argument is not used, the system assumes the *user-ID* value to be the password.

-pf

Specifies the password file. If used without a value, the system assumes a null value. If this argument is not used, the system assumes the *user-ID* value to be the password.

For more information about managing password files, see [Manage password files](#).

This argument is mutually exclusive with the **-p** argument.

-g

Specifies the group associated with the user.

If used without a value, the user's default group is assumed.

-h

Displays help for this utility.

ENVIRONMENT

As specified in [Manually configuring your environment for Teamcenter utilities](#).

FILES

As specified in [Log files produced by Teamcenter](#).

RESTRICTIONS

This utility must be run by a user with DBA permissions, for example, **infodba**.

purge_historical_revisions

Searches the database for historical revisions of any revisable class that are no longer needed. It purges all historical revisions it finds, except for the following:

- All private and latest public revisions.
- Revisions with one or more labels.
- Any revision with a precise reference from another object.
- Any revision with populated attributes that identify it as purge protected or having an ITAR license.
- All revisions that were updated more recently than the interval specified in the **POM_PURGE_AGE_LIMIT** environment variable.

Note Set this environment variable to a negative value to disable purging, for example, **-1**.

The utility exits when no candidates for purging remain or if the specified timeout period expires.

You can run the utility periodically using the Dispatcher, for example, overnight or other times when system activity is low. If you do this, ensure you set the value of the **timeout** argument to less than the repeat interval of the periodic runs.

Revisions are purged in batches to reduce impact to users of the system.

SYNTAX

purge_historical_revisions [-u=*user-id* {-p=*password* | -pf=*password-file*} -g=*group*] -timeout=*timeout_seconds* [-h]

ARGUMENTS

-u

Specifies the user ID.

This is generally **infodba** or another user with administration privileges. If this argument is used without a value, the operating system user name is used.

Note If Security Services single sign-on (SSO) is enabled for your server, the user and password arguments are authenticated externally through SSO rather than being authenticated against the Teamcenter database. If you do not supply these arguments, the utility attempts to join an existing SSO session. If no session is found, you are prompted to enter a user ID and password.

-p

Specifies the password.

If used without a value, the system assumes a null value. This argument is mutually exclusive with the **-pf** argument.

If this argument is not used, the system assumes the *user-ID* value to be the password.

-pf

Specifies the password file. If used without a value, the system assumes a null value. If this argument is not used, the system assumes the *user-ID* value to be the password.

For more information about managing password files, see [Manage password files](#).

This argument is mutually exclusive with the **-p** argument.

-g

Specifies the group associated with the user.

If used without a value, the user's default group is assumed.

-timeout

Specifies the maximum period that the utility runs, in seconds. When the timeout period expires, the utility exits.

-h

Displays help for this utility.

ENVIRONMENT

As specified in [Manually configuring your environment for Teamcenter utilities](#).

FILES

As specified in [Log files produced by Teamcenter](#).

RESTRICTIONS

This utility must be run by a user with DBA permissions, for example, **infodba**. This restriction applies whether you run it manually or periodically using Dispatcher.

EXAMPLES

```
purge_historical_revisions -timeout=3600 -u=adminjones -p=passjones -g=dba
```

validate_revrule_effectivity

Validates the effectivity criteria on one or more specified revision rules, generates validation records, and associates them with the revision rules. Optionally, you can create a text file containing the names of revision rules to validate.

Validation may take a significant time, depending on the number of constraints and default rules defined in the system. If you have a large number of constraints and rules, consider running the utility during off-peak hours.

SYNTAX

```
validate_revrule_effectivity [-u=user-id {-p=password |  
-pf=password-file} -g=group]  
{-rule_name=rule-name1 [-rule_name=rule-name2] | -rev_rule_names_file=file-name}  
substitute_dep_vars=0 | 1+ [-apply_constraints=0 | 1+] [-apply_default=0 | 1+]  
{-model_id=model-id | [-product_name=product-name  
-product_namespace=product-namespace]} [-h]
```

ARGUMENTS**-u**

Specifies the user ID.

This is generally **infodba** or another user with administration privileges. If this argument is used without a value, the operating system user name is used.

Note

If Security Services single sign-on (SSO) is enabled for your server, the user and password arguments are authenticated externally through SSO rather than being authenticated against the Teamcenter database. If you do not supply these arguments, the utility attempts to join an existing SSO session. If no session is found, you are prompted to enter a user ID and password.

-p

Specifies the password.

If used without a value, the system assumes a null value. This argument is mutually exclusive with the **-pf** argument.

If this argument is not used, the system assumes the *user-ID* value to be the password.

-pf

Specifies the password file. If used without a value, the system assumes a null value. If this argument is not used, the system assumes the *user-ID* value to be the password.

For more information about managing password files, see [Manage password files](#).

This argument is mutually exclusive with the **-p** argument.

-g

Specifies the group associated with the user.

If used without a value, the user's default group is assumed.

-rule_name

Specifies the name of the revision rule to validate. If you want to validate more than one revision rule, use this argument once for each revision rule. This argument

is optional if you use the **-rev_rules_file_name** argument, but takes precedence if both arguments are entered.

-rev_rules_file_name

Specifies the name and full path of a text file containing the names of revision rules to validate. The file must contain the name of one revision rule on each line. This argument is optional if you use the **-rule_name** argument.

-substitute_dep_vars

Defines if and how the configurator substitutes dependent variables (for example, **PlantVacationShutdown**) in the effectivity criteria on the revision rules.

Note Teamcenter configurators do not currently support dependent variable substitutions and this argument has no effect.

0

Dependent variables are not substituted.

1

Dependent variables are substituted and all modifiers are off. Forward and reverse substitutions are applied as follows:

- Forward substitutions

Dependent variable substitution are defined by:

- o A dependent variable, for example, *D*.
- o A master variable, for example, *M*.
- o Formula describing how *D* depends on *M*.

When these definitions have been made, the system can accept effectivity conditions such as $M = D$, $M \neq D$, $M > D$, and $M \leq D$.

For example:

- o *Vacation* depends on *Date*.
- o *Date* determines the state of the variable *Vacation*.
- o $Vacation := Jul-1 \leq Date < Aug-1$

A forward substitution dynamically complements criteria containing statements about *Vacation* with statements about *Date*. For example, a criterion " $Date \geq Jul-1 \ \&\& \ Date < Aug-1$ " is added for the criterion " $Vacation=Date$ ". This enables users to author effectivity criteria in May in terms of the dependent variable *Vacation* whose state is set dynamically at the time the configuration criteria are applied. This time may be later than when you author the effectivity criteria. If the definition of *Vacation* changes in June, the effectivity criteria that were authored in May still evaluate correctly.

Forward substitutions are defined from a dependent to a master. Users who provide solve criteria in terms of the dependent correctly evaluate effectivity statements that were made in terms of the master. In the previous example,

forward substitution ensures that date conditions are substituted for all date-dependent conditions.

- Reverse substitutions

Date-dependent conditions such as **PlantVacationShutdown** are derived from date conditions. In the following example of a dependent variable substitution:

- o *Vacation* depends on *Date*
- o *Date* determines the state of the variable *Vacation*
- o *Vacation* := Jul-1 <= *Date* < Aug-1

A reverse substitution dynamically complements criteria containing statements about *Date* with statements about *Vacation*. For example, you can add a *Vacation Date* criterion to criteria like *Date* >= Sep-1". (You can control the detailed behavior of this substitution with modifiers). This enables users to author effectivity conditions in May that respond to a *Vacation* variable whose state is dynamically set at the time a configuration filter is applied. This may be set later than you author the effectivity condition. If the definition of *Vacation* changes in June, effectivity conditions that were authored in May still evaluate correctly.

Reverse substitutions are defined from a master to a dependent. Users who provide solve criteria in terms of the master correctly evaluate effectivity statements that were made in terms of the dependent. In the previous example, reverse substitution ensures that date-dependent criteria such as *Vacation* are derived from the date ranges specified in the solve criteria.

3

Forward substitution is suppressed, if modifier 2 is on.

5

Reverse substitution is suppressed, if modifier 4 is on.

9

Reverse substitution uses a set-based substitution mode. For example consider the following definition of a dependent variable substitution.

- *Vacation* depends on *Date*.
- *Date* determines the state of the variable *Vacation*
- *Vacation* := Jul-1 <= *Date* < Aug-1

A set-based reverse substitution complements solve criteria of Jul-1 <= *Date* <= Dec-31 with *Vacation* <= *Date*. Equality is given because the substituent (Jul-1 <= *Date* < Aug-1) is a subset of the criteria, while precedency is given because the criteria also contain dates that compare greater than the substituent.

If modifier 8 is off, the system uses the default reverse substitution mode, which is vertex based. This mode would complement solve criteria Jul-1 <= *Date* <= Dec-31 with *Vacation* < *Date* | *Vacation* = *Date* | *Vacation* > *Date*. It does this because the criteria contain dates that compare to at least one vertex of the substituent (Jul-1 <= *Date* < Aug-1) with each comparison operator ("<", "=", ">").

17

Reverse substitution uses universal quantification. If modifier 16 is off, the system uses the default of existential quantification. While existential quantification searches for at least one match, universal quantification attempts to prove the absence of a violation. Both modes should produce the same results, though the required effort could be substantially different.

Note

The following values are not permitted and may give incorrect results: any even number, 7, 13, 21, and 29.

-apply_constraints

Specifies whether the system applies constraints.

0

The system does not apply constraints.

1

The system applies constraints.

-apply_defaults

Specifies whether the system applies defaults.

0

The system does not apply defaults.

1

The system applies defaults.

-model_id

This is an optional argument and is specified only if you connect to an external configurator. Specifies the identifier of the 4GD model object in the context of which the revision rules should be validated. CPD model objects define the appropriate external configurator using the **EffectivityInModel** object.

-product_name

Specifies the name of the product, for example, the item ID. Teamcenter uses this value in conjunction with the **-product_namespace** argument to resolve any ambiguities in effectivity option value names. If you do not specify a value, Teamcenter deduces the product name from the **EffectivityInModel** object associated with the model.

-product_namespace

Specifies the namespace of the product in which the product name has unique semantics, for example, the item revision ID, model year, and product type. Teamcenter uses this value in conjunction with the **-product_name** argument to resolve any ambiguities in effectivity option value names. If you do not specify a value, Teamcenter deduces the product namespace from the **EffectivityInModel** object associated with the model.

-h

Displays help for this utility.

ENVIRONMENT

As specified in [Manually configuring your environment for Teamcenter utilities](#).

FILES

As specified in [Log files produced by Teamcenter](#).

RESTRICTIONS

Consider the following two products that are configured with effectivity:

Product	Default	Rule checks
P1	WHILE Unit > 5 DEFAULT Date >= 2012-01-01	WHILE Unit <= 5 RAISE ERROR “message” IF Date < 2012-01-01
P2		RAISE ERROR “message” Unit > 5

If you validate a specified revision rule with an effectivity criteria of **Unit=7**, you obtain the following results:

Revision rule effectivity criteria	Product	Result	Violations	Applied default
Unit=7	P1	Unit=7, Date=2012-01-01..	None	Date >= 2012-01-01
	P2	Unit=7	Unit > 5: “message”	None

These validation results are persisted in records that are attached to the revision rule. If a client queries the revision rule effectivity criteria in the context of product P1 or P2, the server response includes the corresponding validation records. This allows sites to run this utility to perform potentially CPU intensive criteria validation once for each revision rule during off-peak hours, while permitting the system to deliver an acceptable validation service to hundreds of users during working hours.

There are two limitations to this practice:

- In this version of Teamcenter, you cannot create, view, or change defaults and rule check constraints in the user interface. You can create them only with ITK APIs.
- In this version of Teamcenter, you cannot review validation results in the user interface when asking the server for revision rule effectivity criteria. You can view results only with the **validate_revrule_effectivity** utility.

Also, when you validate the effectivity criteria of a revision rule, the criteria are sent to a configurator service for validation. These criteria may be sent to external or remote configurator services in a future version of Teamcenter (this capability is not supported in the current version). In this case, the construction of the validation records is subject to any limitations of the external or remote configurator service.

EXAMPLES

- This example validates a revision rule called **My Latest Released** in the context of a model whose ID is **ship123**. It does not substitute dependent variables, and does not apply constraints or default rules. It creates a validation record and associates it with the revision rule.


```
validate_revrule_effectivity -u=infodba -p=password -g=dba  
-rule_name=My Latest Released -model_id=ship123
```

- This example validates revision rules called **My Latest Working** and **My Latest Released** which are listed in the file **C:\xyz\revrule.txt** in the context of a model whose ID is **Malibu_V6_2010**. It enables forward and reverse substitution of dependent variables (modifiers are off) and applies defaults and constraints. It creates validation records for each revision rule and associates them with the relevant revision rules.

```
validate_revrule_effectivity -u=infodba -p=password -g=dba  
-rev_rules_file_name=C:\xyz\revrule.txt -substitute_dep_vars=1  
-apply_constraints=1 -apply_defaults=1 -model_id=Malibu_V6_2010
```

Chapter

6 *Workflow utilities*

clear_process_stage_list

Clears the **process_stage_list** field of the workspace object.

Note Running this utility changes the date and time stamp of the objects that it is run against.

SYNTAX

clear_process_stage_list [-u=*user-id* {-p=*password* |
-pf=*password-file*} -g=dba] -folder [-h]

ARGUMENTS**-u**

Specifies the user ID.

If this argument is used without a value, the operating system user name is used.

-p

Specifies the user's password.

If used without a value, the system assumes a null value. If this argument is not used, the system assumes the *user-ID* value to be the password.

This argument is mutually exclusive with the **-pf** argument.

-pf

Specifies the password file. If used without a value, the system assumes a null value. If this argument is not used, the system assumes the *user-ID* value to be the password.

For more information about managing password files, see [Manage password files](#).

This argument is mutually exclusive with the **-p** argument.

-g

Specifies the group associated with the user. The group value must be **dba** to run this utility.

-folder

Specifies the folder name where the target workspace objects should be placed whose process stage lists are to be cleared. The folder must be a single folder directly inside the executing user's **Home** folder.

-h

Displays help for this utility.

ENVIRONMENT

As specified in [Manually configuring your environment for Teamcenter utilities](#).

FILES

As specified in [Log files produced by Teamcenter](#).

RESTRICTIONS

You must be logged on as a member of the **dba** group.

EXAMPLES

To clear the **process_stage_list** fields of all the objects in the **my_folder** object, enter the following command on a single line:

```
$TC_ROOT/bin/clear_process_stage_list -u=user-id -p=password -g=dba  
-folder=my_folder
```

global_transfer

Transfers all tasks of one user ID or resource pool to another user ID or resource pool. This utility provides the capability to transfer tasks, as follows:

- Users can transfer their own tasks to another user.
- Users can transfer the tasks of other users.
- Users with group administrator privileges can transfer tasks assigned to members of their group.
- System administrators can transfer tasks belonging to any user to a different user.

When transferring tasks, such as do tasks or **select-signoff-team** tasks, the responsible party for each task is transferred to the new resource pool or user.

When transferring **perform-signoff** tasks, the tasks of the current resource pool or current user are delegated to a new resource pool or user if the new resource pool or user meets the same requirements as if the task were delegated using the delegate feature in the Teamcenter interface. If the signoff task is associated with a signoff profile, the delegation is constrained to another user or resource pool of the group/role specified by the signoff profile and the list of users to select from is filtered. If the signoff task is not associated with a signoff profile, delegation to any user or resource pool is possible, and the list of users to select from is not filtered.

SYNTAX

global_transfer [-u=*user-id* {-p=*password* | -pf=*password-file*}
-g=*group-name*] [-f=*user-ID*] [-t=*user-ID*] [-h]

ARGUMENTS**-u**

Specifies the user ID.

This is generally **infodba** or another user with administration privileges. If this argument is used without a value, the operating system user name is used.

Note

If Security Services single sign-on (SSO) is enabled for your server, the **-u** and **-p** arguments are authenticated externally through SSO rather than being authenticated against the Teamcenter database. If you do not supply these arguments, the utility attempts to join an existing SSO session. If no session is found, you are prompted to enter a user ID and password.

-p

Specifies the user's password.

If used without a value, the system assumes a null value. If this argument is not used, the system assumes the *user-ID* value to be the password.

This argument is mutually exclusive with the **-pf** argument.

-pf

Specifies the password file. If used without a value, the system assumes a null value. If this argument is not used, the system assumes the *user-ID* value to be the password.

For more information about managing password files, see [Manage password files](#).

This argument is mutually exclusive with the **-p** argument.

-g

Specifies the group associated with the user.

If used without a value, the user's default group is assumed.

-f

Specifies the ID of the user whose inbox tasks are being transferred or the group/role resource pool inbox tasks of a specified group/role. **Group/Role** transfers resource pool inbox tasks for a specified group/role. **Group/*** transfers resource pool inbox tasks of a specified group and any role. ***/Role** transfers resource pool inbox tasks of a specified role and any group.

-t

Specifies the ID of the user to whom the tasks are being transferred, or the group/role transfers resource pool inbox tasks of a specified group/role. **Group/*** transfers resource pool inbox tasks of a specified group and any role. ***/Role** transfers resource pool inbox tasks of a specified role and any group.

-h

Displays help for this utility.

ENVIRONMENT

As specified in [Manually configuring your environment for Teamcenter utilities](#).

FILES

The **gtransfer_XXXXX.log** file provides a listing of selected tasks, whether they have been transferred (**Y/N**), to whom they were transferred, and if there were any errors in the transfer.

RESTRICTIONS

None.

EXAMPLES

To transfer all tasks from user Mike to user Kevin, enter the following command on a single line:

```
global_transfer -f=mike -t=kevin
```

install_handlers

Defines action handlers. It can also configure new action handlers and modify the definition of existing handlers.

SYNTAX

```
install_handlers [-u=user-id {-p=password | -pf=password-file} -g=group-name]
-f= {install | create | modify | delete | listall} -id=handler-ID
[-funcname=function-name] -functype=1 | 2
-execmode=1 | 2 -desc=handler-description
[-retrycount=retry-count] [-retryinterval=retry-interval-in-minutes]
[-exectime=time-of-the-day-in-24-hour-format] -override=true | false [-h]
```

ARGUMENTS

-u

Specifies the user ID.

This is generally **infodba** or another user with administration privileges. If this argument is used without a value, the operating system user name is used.

Note

If Security Services single sign-on (SSO) is enabled for your server, the **-u** and **-p** arguments are authenticated externally through SSO rather than being authenticated against the Teamcenter database. If you do not supply these arguments, the utility attempts to join an existing SSO session. If no session is found, you are prompted to enter a user ID and password.

-p

Specifies the user's password.

If used without a value, the system assumes a null value. If this argument is not used, the system assumes the *user-ID* value to be the password.

This argument is mutually exclusive with the **-pf** argument.

-pf

Specifies the password file. If used without a value, the system assumes a null value. If this argument is not used, the system assumes the *user-ID* value to be the password.

For more information about managing password files, see [Manage password files](#).

This argument is mutually exclusive with the **-p** argument.

-g

Specifies the group associated with the user.

If used without a value, the user's default group is assumed.

-f

Specifies the mode in which the utility must execute. The mode must be one of the following:

- **install**
- **create**
- **modify**

- **delete**
- **listall**

-id=

Specifies the handler ID.

-funcname

Specifies the name of the library function. Use this argument when the **-functype** argument is set to **1**.

-functype

Specifies whether the handler is a library function or a stand-alone executable. The value for this argument must be one of the following:

1

Library function

2

Stand-alone executable

-execmode

Specifies the handler's execution mode. The value for this argument must be one of the following:

1

Executes the handler in the calling process.

2

Executes the handler as a separate process.

-desc

Specifies the handler description.

-override

Specifies if the handler execution time can be overridden. The value for this argument must be one of the following:

true

Allows override.

false

Disables override.

-h

Displays help for this utility.

ENVIRONMENT

As specified in [Manually configuring your environment for Teamcenter utilities](#).

FILES

As specified in [Log files produced by Teamcenter](#).

RESTRICTIONS

None.

EXAMPLES

- To install the default action handlers, enter the following command on a single line:

```
install_handlers -f=install
```

- To create an action handler with the specified attribute values to set the execution time for the handler to 6.00 p.m., enter the following command on a single line:

```
install_handlers -f=create -id=MyActionHandler -funcname=Myfunc  
-functype=1 -execmode=1 exectime=1800
```

- To set the retry count value to **5** for the **MyActionHandler** action handler, enter the following command on a single line:

```
install_handlers -f=modify -id=MyActionHandler -retryCount=5
```

- To delete the **MyActionHandler** action handler, enter the following command on a single line:

```
install_handlers -f=delete -ID=MyActionHandler
```

- To list all the action handlers defined in the database, enter the following command on a single line:

```
install_handlers -f=listall
```

migrate_ecm_ids

Allows users to create Change Viewer objects with a new naming rule that changes the name length in a database originally created in Teamcenter engineering process management. This utility must be run in the Teamcenter environment after upgrading from a Teamcenter engineering process management.

SYNTAX

migrate_ecm_ids [-u=*user-id* {-p=*password* | -pf=*password-file*} -g=*group-name*]

ARGUMENTS

-u

Specifies the user ID.

This is generally **infodba** or another user with administration privileges. If this argument is used without a value, the operating system user name is used.

Note

If Security Services single sign-on (SSO) is enabled for your server, the **-u** and **-p** arguments are authenticated externally through SSO rather than being authenticated against the Teamcenter database. If you do not supply these arguments, the utility attempts to join an existing SSO session. If no session is found, you are prompted to enter a user ID and password.

-p

Specifies the user's password.

If used without a value, the system assumes a null value. If this argument is not used, the system assumes the *user-ID* value to be the password.

This argument is mutually exclusive with the **-pf** argument.

-pf

Specifies the password file. If used without a value, the system assumes a null value. If this argument is not used, the system assumes the *user-ID* value to be the password.

For more information about managing password files, see [Manage password files](#).

This argument is mutually exclusive with the **-p** argument.

-g

Specifies the group associated with the user.

If used without a value, the user's default group is assumed.

-h

Displays help for this utility.

ENVIRONMENT

As specified in [Manually configuring your environment for Teamcenter utilities](#).

FILES

As specified in [Log files produced by Teamcenter](#).

RESTRICTIONS

None.

EXAMPLES

None.

migrate_wf_handlers

Transforms the name and/or arguments of workflow handlers from one format to another. Task templates may have one or more workflow handlers associated with them. This utility transforms the handlers associated with templates that are active, as well as template versions that are obsolete but that are still referenced by uncompleted workflow processes or jobs.

This utility uses an XML mapping file to migrate handlers and their arguments. The mapping file top-level nodes define an action (rule or transform) to perform, such as **Replace**, **Remove**, or **Update**, on the individual handler specified within them.

SYNTAX

```
migrate_wf_handlers [-u=user-id {-p=password | -pf=password-file}
-g=group-name] [-report=report-file-name [-dryrun] [-listonly]
[-templates=template-to-migrate, ...] [-templates_file=path-and-name-of-csv-file]
[-mapping_file=path-and-name-of-xml-mapping-file] [-v] [-h]
```

ARGUMENTS

Entries in parentheses are accepted abbreviations for arguments.

-u

Specifies the user ID.

This is generally **infodba** or another user with administration privileges. If this argument is used without a value, the operating system user name is used.

Note

If Security Services single sign-on (SSO) is enabled for your server, the **-u** and **-p** arguments are authenticated externally through SSO rather than being authenticated against the Teamcenter database. If you do not supply these arguments, the utility attempts to join an existing SSO session. If no session is found, you are prompted to enter a user ID and password.

-p

Specifies the user's password.

If used without a value or if neither the **-pf** nor the **-p** argument is used, the system displays an error and asks for a username and password interactively.

This argument is mutually exclusive with the **-pf** argument.

-pf

Specifies the password file.

If used without a value or if neither the **-pf** nor the **-p** argument is used, the system displays an error and asks for a username and password interactively.

For more information about managing password files, see [Manage password files](#).

This argument is mutually exclusive with the **-p** argument.

-g

Specifies the group associated with the user.

If used without a value, the user's default group is assumed.

-report

Specifies the output file. If this argument is not specified, the output is written to the standard output.

-dryrun

Runs the command without making any changes. Actions that would have been performed are displayed instead.

-listonly

Lists the handlers and arguments it would have migrated, but makes no changes.

-templates

Limits the migration of handlers to ones owned by the specified templates. If you use more than one template name, separate them with a comma. If a template name has spaces in it, quotes are required around the name. Mutually exclusive with the **-templates_file** argument.

-templates_file

Limits the migration of handlers to ones owned by the templates specified in the named file. The file contains a comma-delimited list of one or more template names. Mutually exclusive with the **-templates** argument.

-mapping_file

Specifies the path and file name of the XML mapping file containing transforms or rules with the old and new names for handlers and arguments. The mapping file provided by Siemens PLM Software to convert handler names and arguments from versions prior to Teamcenter 10.1 is **TC_DATA\wf_handler_migration.xml**.

-v

Displays verbose output and its use is recommended if you are running this utility manually. If this argument is not specified, nothing is displayed when the utility is run.

-h

Displays help for this utility.

ENVIRONMENT

As specified in [Manually configuring your environment for Teamcenter utilities](#).

FILES

As specified in [Log files produced by Teamcenter](#).

RESTRICTIONS

- After replacing one handler with two or more new handlers using the **Replace** rule in the map file, the new handlers are not available for further processing during the same run of the utility in subsequent mapping file elements. However, the handlers would be available for processing when you run the utility again.
- The **Replace** rule cannot replace an action handler with a rule handler and vice versa.

EXAMPLES

- The following example does not migrate handlers, it only performs a dry run in verbose mode, with the output information sent to the standard output. The

TC_DATA variable is specified for Windows systems. Because **-u** and **-p** are not specified, the user's operating system credentials are used.

```
migrate_wf_handlers -v -dryrun -mapping_file=
%TC_DATA%\wf_handler_migration.xml
```

- The following example migrates the handlers using the **tcadmin** user with verbose output to the standard output. The *TC_DATA* variable is specified for UNIX/Linux systems.

```
migrate_wf_handlers -v -u=tcadmin -p=tcadmin -mapping_file=
$TC_DATA/wf_handler_migration.xml
```

- The following example migrates the handlers using the **tcadmin** user and a password file with verbose output to the specified report file (**rpt_file**):

```
migrate_wf_handlers -v -u=tcadmin -pf=pswd_file -report=rpt_file
-mapping_file=./map.xml
```

- The following example migrates in silent mode only the handlers used by the specified workflow template:

```
migrate_wf_handlers -templates="Authorization WF Template"
-mapping_file=D:\Temp\MyMigrationMappingFile.xml
```

purge_processes

Purges completed processes based on the last-modified date of the process. Objects such as e-mail messages that reference the process are not deleted. Use the **-f** argument to delete both in-process and completed processes and sever the references between objects and the processes.

SYNTAX

purge_processes [-u=*user-id* {-p=*password* | -pf=*password-file*}
-g=*group-name*] -d=*MM-DD-YYYY* [-force] [-r] [-h]

ARGUMENTS

Entries in parentheses are accepted abbreviations for arguments.

-u

Specifies the user ID.

This is generally **infodba** or another user with administration privileges. If this argument is used without a value, the operating system user name is used.

Note

If Security Services single sign-on (SSO) is enabled for your server, the **-u** and **-p** arguments are authenticated externally through SSO rather than being authenticated against the Teamcenter database. If you do not supply these arguments, the utility attempts to join an existing SSO session. If no session is found, you are prompted to enter a user ID and password.

-p

Specifies the user's password.

If used without a value, the system assumes a null value. If this argument is not used, the system assumes the *user-ID* value to be the password.

This argument is mutually exclusive with the **-pf** argument.

-pf

Specifies the password file. If used without a value, the system assumes a null value. If this argument is not used, the system assumes the *user-ID* value to be the password.

For more information about managing password files, see [Manage password files](#).

This argument is mutually exclusive with the **-p** argument.

-g

Specifies the group associated with the user.

If used without a value, the user's default group is assumed.

-d

Specifies the cut-off date for processes to be purged. This is the last-modified date of the process. All processes with a last-modified date equal to or before the specified date are purged.

-f

Deletes in-process and completed processes specified by the **-d** argument from the system and severs references between objects and the processes being

deleted. If this argument is not specified, only completed processes that have no referenced objects are purged.

-r

Generates a report of the number and names of processes to be purged without purging the processes.

-h

Displays help for this utility.

ENVIRONMENT

As specified in [Manually configuring your environment for Teamcenter utilities](#).

FILES

As specified in [Log files produced by Teamcenter](#).

RESTRICTIONS

None.

EXAMPLES

To remove all processes that have been modified on or before 15th April 1998, enter the following command on a single line:

```
purge_processes -u=user-id -p=password -g=dba -d=04-15-1998
```

release_man

Releases objects in batch mode without creating workflow processes or audit files.

SYNTAX

```
release_man [-u=user-ID {-p=password | -pf=password-file} -g=dba]
[-spec] [-unrelease] -retain_release_date [-status=status-type]
-folder=folder-name [-dataset=dataset] [item=item-ID | -key=[keyAttr1=keyVal1]
[,keyAttr2=keyVal2]...[,keyAttrN=keyValN]] [-rev=revision-ID] [-relation=relation]
[-datasetName=dataset-name] [-datasetType=dataset-type] [-force] [-h]
```

ARGUMENTS

-u

Specifies the user ID.

If this argument is used without a value, the operating system user name is used.

-p

Specifies the user's password.

If used without a value, the system assumes a null value. If this argument is not used, the system assumes the *user-ID* value to be the password.

This argument is mutually exclusive with the **-pf** argument.

-pf

Specifies the password file. If used without a value, the system assumes a null value. If this argument is not used, the system assumes the *user-ID* value to be the password.

For more information about managing password files, see [Manage password files](#).

This argument is mutually exclusive with the **-p** argument.

-g

Specifies the group. Must be the **dba** group.

See restriction #1.

-spec

Indicates that specifications and BOM view revisions of an item revision in the release folder are released along with the item revision.

-unrelease

Removes the specified status type. See restrictions #2 and #5.

-retain_release_date

Specifies that if the object to be released is already released, the original release date is retained. See restriction #5.

-status

Specifies the status type to be applied to all objects.

See restriction #2.

-folder

Specifies the name of the release folder. Place the objects whose status you want changed into the release folder. See restriction #3.

Status is changed only on the objects one level down within the folder. For example, if you place an item in the folder that contains multiple item revisions, and these item revisions contain multiple datasets, only the status of the item is changed. The status of the item revisions and their datasets is not changed. To change the status of the item revisions, each item revision must be individually placed in the folder at the same level as the item.

Use the **-spec** argument to change the status of specifications and BOM view revisions of item revisions.

-dataset

Specifies that a dataset is released.

-item

Specifies the item ID of the dataset to be released. Use with the **-dataset** argument.

-key

Specifies the key of the dataset to be released. Use with the **-dataset** argument.

Use the following format:

```
[keyAttr1=keyVal1] [,keyAttr2=keyVal2]...[,keyAttrN=keyValN]
```

To find the key of an object, use the **get_key_string** utility.

For more information, see the *Business Modeler IDE Guide*.

-rev

Specifies the revision ID of the dataset to be released. Use with the **-dataset** argument.

-relation

Specifies the relation of the dataset to be released. Use with the **-dataset** argument.

-datasetName

Specifies the name of the dataset to be released. Use with the **-dataset** argument.

-datasetType

Specifies the type of dataset to be released. Use with the **-dataset** argument.

-force

Forces the specified release status to be unreleased, even if there is no release type associated with the status. Must be used with the **-unrelease** argument.

-h

Displays help for this utility.

ENVIRONMENT

As specified in *Manually configuring your environment for Teamcenter utilities*.

FILES

As specified in *Log files produced by Teamcenter*.

RESTRICTIONS

1. The user must be a member of the **dba** group.
2. The status type must be a valid status type defined for your site.
3. The release folder must be a single folder directly inside the executing user's workspace **Home** folder.

4. The **release_man** utility does not release invalid objects or objects locked by other processes.
5. The **-retain_release_date** and **-unrelease** arguments cannot to be used together.

EXAMPLES

- To apply the **Released** status type to all objects in the **my_folder** folder (including ItemRevision specifications and BOM view revisions), enter the following command on a single line:

```
$TC_ROOT/bin/release_man -u=user-id -p=password -g=dba -spec  
-status=Released -folder=my_folder
```

- To remove the **Released** status type from all objects in the **my_folder** folder (including ItemRevision specifications and BOM view revisions), enter the following command on a single line:

```
TC_ROOT/bin/release_man -u=user-id -p=password -g=dba -spec  
-unrelease -status=Released -folder=my_folder
```

released_parts_collector

Socket-based server application that receives requests from the **RDV-add-released-parts-queue** workflow handler using the Teamcenter server. Upon receiving the request, it opens the file that contains the list of parts to be processed, locks the file, appends the newly released IA information to the file, and unlocks the file. This application can be used in two modes. When using the **-S** option, it works as a server, and when using the **-C** option, it works as a client application.

SYNTAX

Server mode:

released_parts_collector_server S *host-name:port-number master-file*

Client mode:

released_parts_collector_server C *host-name:port-number*
[**-u**=*user-id* {**-p**=*password* | **-pf**=*password-file*} **-g**=*dba*]
Command 1: **Cf** *host-name:port-number client-IA-file*

Command 2: **C** *host-name:port-number stop*

Command 3: **C** *host-name:port-number empty-master-list*
[**-h**]

ARGUMENTS

-u

Specifies the user ID.

If this argument is used without a value, the operating system user name is used.

-p

Specifies the user's password.

If used without a value, the system assumes a null value. If this argument is not used, the system assumes the *user-ID* value to be the password.

This argument is mutually exclusive with the **-pf** argument.

-pf

Specifies the password file. If used without a value, the system assumes a null value. If this argument is not used, the system assumes the *user-ID* value to be the password.

For more information about managing password files, see [Manage password files](#).

This argument is mutually exclusive with the **-p** argument.

-g

Specifies the group. Must be the **dba** group.

See restriction #1.

-S

Specifies that the application functions as a server.

host-name:port-number

Indicates the port number at which this server should listen for incoming requests.

master-file

Specifies the absolute path to the master XML file to be created/updated with the released parts information. The released parts information is received from either the **RDV-add-released-parts-queue** workflow handler or the **released_parts_collector** command line utility in client mode.

-C

Specifies that the application functions as a client.

host-name:port-number

Specifies the name and port number of the server. This is required when running the utility in client mode.

client-IA-file

Defines the absolute path of an XML file containing the list of released parts. This option must be specified only when running the utility in client mode. When this option is used, the client application sends the list of parts in the specified file to the server, which appends the new parts to the master list.

stop

Sends a message to stop the server.

empty_master_list

Sends a message to the server to empty the master released parts list.

-h

Displays help for this utility.

FILES

None.

RESTRICTIONS

None.

EXAMPLES

- To start the server on the **trysun12** machine at port **5567** and create the file **master_list.xml** file, enter the following command on a single line:

```
released_parts_collector -S trysun12:5567 /tmp/master_list.xml
```

- To send the contents of the file **/tmp/my_released_list.xml** to the server running on **trysun12** at port **5567** and add the new list of parts to the master list of released parts, enter the following command on a single line:

```
released_parts_collector -Cf trysun12:5567 /tmp/my_released_list.xml
```

- To stop the server running on **trysun12** at port **5567**, enter the following command on a single line:

```
released_parts_collector -C trysun12:5567 stop
```

- To empty the released parts list XML file on the server running on **trysun12** at port **5567**, enter the following command on a single line:

```
released_parts_collector -C trysun12:5567 clear_master_file
```

tc_workflow_postprocess

Executes a specific action on a specific task in the workflow process from which the related background process was initiated.

For example, you can use this utility to evoke the **complete** action on the **Review** task in the workflow process from which a background tessellation process was initiated. This utility then prompts the workflow task to either execute an action (defined with the **-action** argument) or submit a decision (defined with the **-signoff** argument). This can be useful when the conditions to execute the action are not met until the background process completes. In these cases, the user has typically moved on to other tasks or ended the session.

Use the **-member_group** and **-member_role** arguments to define the group/role used for the background process. This is useful at sites where users have multiple groups/roles and the user has changed to a group/role that is different from his default login group/role while initiating the background process. These arguments allow the **tc_workflow_postprocess** utility to assume the same group/role the user was using at the time the workflow process was initiated. It is expected that the same group/role is required to execute any action in that workflow process on behalf of the user.

SYNTAX

```
tc_workflow_postprocess [-u=user-id {-p=password | -pf=password-file}
-g=group-name] -status_xfer_type=transfer-type
[itemid=item-id | -key=[keyAttr1=keyVal1][,keyAttr2=keyVal2]...[,keyAttrN=keyValN]
-revid=rev-id -dsname=dataset-name -dataset_tag=dataset-tag -task_tag=tag
[-member_group=group] [-member_role=role] [-action=action-name]
[-trigger_comment=comment] [-signoff=decision]
[-signoff_comment=comment] [-h]
```

ARGUMENTS

-u

Specifies the user ID.

This is generally a **infodba** or another user with administration privileges. If this argument is used without a value, the operating system user name is used.

-p

Specifies the user's password.

If used without a value, the system assumes a null value. If this argument is not used, the system assumes the *user-ID* value to be the password.

This argument is mutually exclusive with the **-pf** argument.

-pf

Specifies the password file. If used without a value, the system assumes a null value. If this argument is not used, the system assumes the *user-ID* value to be the password.

For more information about managing password files, see [Manage password files](#).

This argument is mutually exclusive with the **-p** argument.

-g

Specifies the group associated with the user.

If used without a value, the user's default group is assumed.

If the utility is called using the [EPM-invoke-system-action](#) or [EPM-invoke-system-rule](#) handlers, the utility inherits the session authentication and the **-u/-p** switches are not needed, irrespective of the autologin setting.

-status_xfer_type

Indicates the type of status transfer to perform. An argument value of **cae_mesh** transfers the release status to an associated **CAEMesh** dataset. Any other value, or not supplying the argument, transfers the release status to an associated **DirectModel** dataset. This argument and value are valid only if the **-dataset_tag** argument is specified; otherwise, this argument is ignored.

-itemid

Identifies the item under which to locate the **CAEMesh** dataset. The utility transfers the release status to the **CAEMesh** dataset at the location indicated by this argument, the **-revid**, and the **-dsname** arguments. This argument and value must be supplied only if the argument/value **-status_xfer_type=cae_mesh** is specified; otherwise, it is ignored.

-key

Specifies the key of the item under which to locate the **CAEMesh** dataset. The utility transfers the release status to the **CAEMesh** dataset at the location indicated by this argument, the **-revid**, and the **-dsname** arguments. This argument and value must be supplied only if the argument/value **-status_xfer_type=cae_mesh** is specified; otherwise, it is ignored.

Use the following format:

```
[keyAttr1=keyVal1] [,keyAttr2=keyVal2]...[,keyAttrN=keyValN]
```

To find the key of an object, use the [get_key_string](#) utility.

For more information, see the [Business Modeler IDE Guide](#).

-revid

Identifies the item revision under which to locate the **CAEMesh** dataset. The utility transfers the release status to the **CAEMesh** dataset at the location indicated by this argument, the **-itemid**, and the **-dsname** arguments. This argument and value must be supplied only if the argument/value **-status_xfer_type=cae_mesh** is specified; otherwise, it is ignored.

-dsname

Identifies the name of the **CAEMesh** dataset to which to transfer the release status. The utility transfers the release status to the **CAEMesh** dataset at the location indicated by this argument, the **-itemid** and the **-revid** arguments. This argument and value must be supplied only if the argument/value **-status_xfer_type=cae_mesh** is specified; otherwise, it is ignored.

-dataset_tag

Specifies the tag of the dataset to which the release status is transferred. The release status of the primary object (target object of the workflow process) is applied to the specified dataset.

-task_tag

Provides a text representation of the tag of the workflow task. The value can be extracted from the XML file provided by either the **EPM-invoke-system-action** or **EPM-invoke-system-rule** handler.

-member_group

Assumes the defined group name before executing an action on the workflow process.

Use when users have multiple groups/roles and the user is expected to change to a group different from their default login group while initiating the background process.

This argument allows the utility to assume the same group the user was using at the time the workflow process was initiated. It is expected that the same group is required to execute any action in that workflow process on behalf of the user.

-member_role

Assumes the defined role name before executing an action on the workflow process.

Use when users have multiple groups/roles and the user is expected to change to a role different from their default login role while initiating the background process.

This argument allows the utility to assume the same role the user was using at the time the workflow process was initiated. It is expected that the same role is required to execute any action in that workflow process on behalf of the user.

-action

Defines which action to trigger in the workflow task specified with the **-task_tag=tag** argument.

Valid actions are: **assign**, **start**, **complete**, **skip**, **suspend**, **resume**, **undo**, **abort**, and **demote**. These action values are not case sensitive.

-trigger_comment

Provides the comment when triggering the action specified in the **-action=action-name** argument.

-signoff

Specifies the utility will perform a signoff with the specified decision.

Valid signoff values are: **approve**, **reject**, **nodecision**. These signoff values are not case sensitive.

-signoff_comment

Provides the comment for the signoff specified with the **-signoff=decision** argument.

-h

Displays help for this utility.

ENVIRONMENT

As specified in *[Manually configuring your environment for Teamcenter utilities](#)*.

FILES

As specified in *[Log files produced by Teamcenter](#)*.

RESTRICTIONS

None.

EXAMPLES

- The following example checks the group/role defined by the **-member_group** and **-member_role** arguments (**Body/Designer**) against the group/role the user is currently logged on with. If they do not match, the group/role of the background processes group/role is changed to **Body/Designer**.

The **complete** action is then invoked for the task specified by the **-task_tag** argument and the comment **Tessellation completed successfully** is displayed.

```
$TC_ROOT/bin/tc_workflow_postprocess
-member_group=Body -member_role=Designer
-task_tag=AZwszeaegnHvqDAAAAAAAAAAAAAA -action=Complete
-trigger_comment="Tessellation completed successfully"
```

- The following example checks the group/role defined by the **-member_group** and **-member_role** arguments (**Body/Designer**) against the group/role the user is currently logged on with. If they do not match, the group/role of the background processes group/role is changed to **Body/Designer**.

The signoff decision **No Decision** is then made for the task specified by the **-task_tag** argument.

```
$TC_ROOT/bin/tc_workflow_postprocess
-member_group=Body -member_role=Designer
-task_tag=AZwszeaegnHvqDAAAAAAAAAAAAAA -signoff=NoDecision
-signoff_comment="Tessellation failed - disk full"
```

- The following example illustrates the use of the **-dataset_tag** argument to apply the release status of a rendering parent object to the related child object.

```
$TC_ROOT/bin/tc_workflow_postprocess
-dataset_tag=AXwszeagnHvqDAAAAAAAAAAAAAA
```

- The following example transfers the release status from a **UGMASTER** dataset to its corresponding **CAEMesh** dataset:

```
$TC_ROOT/bin/tc_workflow_postprocess
-dataset_tag=QZPBK4_6x4$kbDAAAAAAAAAAAAAA
-status_xfer_type=cae_mesh -itemid=000266
-revid=A -dsname=000266/A
```

verify_tasks

Finds all corrupted Change Viewer tasks, jobs, and other associated internal task model objects in order to delete them from the database. If a corrupted object, such as a job, is referenced in a folder, the reference is removed and the job is deleted.

SYNTAX

verify_tasks [-u=*user-id* {-p=*password* | -pf=*password-file*} -g=*group-name*] [-m={**list** | **delete**}] [-h]

ARGUMENTS**-u**

Specifies the user ID.

This is generally **infodba** or another user with administration privileges. If this argument is used without a value, the operating system user name is used.

Note

If Security Services single sign-on (SSO) is enabled for your server, the **-u** and **-p** arguments are authenticated externally through SSO rather than being authenticated against the Teamcenter database. If you do not supply these arguments, the utility attempts to join an existing SSO session. If no session is found, you are prompted to enter a user ID and password.

-p

Specifies the user's password.

If used without a value, the system assumes a null value. If this argument is not used, the system assumes the *user-ID* value to be the password.

This argument is mutually exclusive with the **-pf** argument.

-pf

Specifies the password file. If used without a value, the system assumes a null value. If this argument is not used, the system assumes the *user-ID* value to be the password.

For more information about managing password files, see [Manage password files](#).

This argument is mutually exclusive with the **-p** argument.

-g

Specifies the group associated with the user.

If used without a value, the user's default group is assumed.

-m

Sets mode to one of the following:

list

Lists corrupted jobs and tasks without deleting them.

delete

Lists and deletes corrupted jobs and tasks.

-h

Displays help for this utility.

ENVIRONMENT

As specified in *Manually configuring your environment for Teamcenter utilities*.

FILES

As specified in *Log files produced by Teamcenter*.

RESTRICTIONS

None.

EXAMPLES

None.

Chapter

7 *Data sharing utilities*

attribute_export

Queries for the objects that satisfy the conditions specified by condition property name/property value pairs, sets new values for the properties to be updated on the found objects, and exports the data to a TC XML file. You then must use the [texml_import](#) utility to import the TC XML file containing the updated values back into the database.

Caution Using the bulk load feature (**-bulk_load** argument) of the [texml_import](#) utility to import a TC XML file requires the **SITCONS_AUTH_KEY** environment variable be set to a valid license key value. Due to the potential for data integrity issue if this feature is used improperly, you must meet specific criteria before you can obtain the required key from GTAC.

For more information about loading bulk data, see the [Data Exchange Guide](#).

For more information about loading bulk data, see the Data Exchange Guide.

You can use these two utilities in conjunction to update the properties of numerous objects at once. The update process allows you to update one or more attributes of an object. When a specified object contains multiple attributes, only the specified attributes are exported and updated.

For complex update operations, you can create an input XML file containing instructions on which objects to update and the new values to apply. The file is constructed as a series of **UpdateSet** entries; each entry must contain **type**, **where**, and **update** components.

Before performing lengthy update operations, you can use the utility to determine the number of objects affected by a specified update operation.

For more information about the bulk update process, including instructions on creating the input file, examples of different update operations, performance statistics, and methods for managing the operation's duration, see the [System Administration Guide](#).

SYNTAX

```
attribute_export
-u=user-ID
{-p=password | -pf=password-file}
[-g=group]
{-inputfile=path-to-input-XML-file | -type=type-name}
-cond_prop=property-name
-cond_value=value-name
-update_prop=property-name
-update_value=property-value}
[-cond_operator=operator]
[performSchemaValidation]
[-queryonly]
[-log= log-file-path]
[-outdir=output-XML-file-directory]
[-batchsize=number-of-objects-to-update-per-batch]
[-islandsize=number-of-objects-to-update-per-island]
```

[-untransformed]
[-uidfile]
[-h]

ARGUMENTS

-u

Specifies the user ID. The user must have administrative privileges.

If this argument is used without a value, the operating system user name is used.

-p

Specifies the user's password.

If used without a value, the system assumes a null value. If this argument is not used, the system assumes the *user-ID* value to be the password.

This argument is mutually exclusive with the **-pf** argument.

-pf

Specifies the password file. If used without a value, the system assumes a null value. If this argument is not used, the system assumes the *user-ID* value to be the password.

For more information about managing password files, see [Manage password files](#).

This argument is mutually exclusive with the **-p** argument.

-g

Specifies the group associated with the user. The group value must be **dba** to run this utility.

-inputfile

Specifies the full path to the input XML file containing instructions on which objects to update and the new values to apply. The file is constructed as a series of **UpdateSet** entries; each entry must contain **type**, **where**, and **update** components.

For more information about creating the input file, see the [System Administration Guide](#).

Use the input file for complex updates. (For simpler instances, you can use the following arguments.)

You must use either this argument, or the **-type**, **-cond_prop**, **-cond_value**, **-update_prop** and **-update_value** arguments.

-type

Specifies the type of objects to be updated. For example, **item** or **dataset** or **StructureContext**, and so on.

This argument accepts a single value, which must be a valid Teamcenter object. Use multiple instances of this argument to specify multiple object types.

You must either use this argument in conjunction with the **-cond_prop**, **-cond_value**, **-update_prop** and **-update_value** arguments, or use the **-inputfile** argument.

-cond_prop

Specifies the name of the property to be queried for and updated during export.

This argument accepts multiple values in a comma-separated list. Each value must be a valid property on a Teamcenter object. For example:

-cond_prop=object_name, last_mod_date

You can use multiple instances of this argument. Each instance of this argument must be paired with the **-cond_value** argument.

You must either use this argument in conjunction with the **-type**, **-cond_value**, **-update_prop** and **-update_value** arguments, or use the **-inputfile** argument.

-cond_value

Specifies the new value for the property specified by the **-cond_prop** argument.

This argument accepts multiple values in a comma-separated list. Each value must be a valid property on a Teamcenter object. For example:

-cond_value=TextData_es_ES, "01-DEC-2011 00:00"

You can use multiple instances of this argument. Each instance of this argument must be paired with the **-cond_value** argument.

You must either use this argument in conjunction with the **-type**, **-cond_prop**, **-update_prop** and **-update_value** arguments, or use the **-inputfile** argument.

-update_prop

Specifies the internal name of the property to be updated (as opposed to the display name). For example **object_desc**, **char VLA**, and so on.

This argument accepts multiple values in a comma-separated list. Each value must be a valid property on a Teamcenter object. For example:

-update_prop=object_name, object_desc

You can use multiple instances of this argument. Each instance of this argument should be paired with the **-update_value** argument.

You must either use this argument in conjunction with the **-type**, **-cond_prop**, **-cond_value**, and **-update_value** arguments, or use the **-inputfile** argument.

-update_value

Specifies the new value for the property specified by the **-update_prop** argument.

This argument accepts multiple values in a comma-separate list. For example:

-update_value=folder, "Home folder"

You can use multiple instances of this argument. Each instance of this argument must be paired with the **-update_prod** argument.

You must either use this argument in conjunction with the **-type**, **-cond_prop**, **-cond_value**, and **-update_prop** arguments, or use the **-inputfile** argument.

-cond_operator

Specifies the operator to be used with the condition arguments. Valid operations are: **equal**, **not equal**, **greater than**, **greater than and equal**, **less than**, **less than and equal**.

When setting the value for this argument in the command window, the format for these operations are: **EQ**, **NE**, **GT**, **GE**, **LT**, and **LE**.

Note When setting the value for this argument in the input file, you can use the above format, or the following characters: =, !=, >, >=, <, <=.

This argument is optional and is only valid when used with, and placed after, the **-cond_prop** and **-cond_value** arguments. For example:

```
-cond_prop=object_name -cond_value="My obj #1"  
-cond_prop="last_mod_date" -cond_value="01-DEC-2011 00:00"  
-cond_operator="LE" -cond_prop="last_mod_date"  
-cond_value="01-DEC-2010 00:00" -cond_operator="GE"
```

-performSchemaValidation

Enables schema validation of the XML input file.

-queryonly

Outputs the number of target objects affected by the specified update parameters to a log file. If the number of objects is less than 100, the object UIDs are included in the log file.

When you specify this switch, the utility does not perform the update, it merely reports the number of affected objects.

Use this switch in conjunction with either the input file or the condition and update arguments to determine how many objects are affected by specified update operation. You can use the resulting information to determine batch size, and to estimate the duration of the update operation.

For more information about update duration, see the [System Administration Guide](#).

-log

Specifies the path to the log file. By default, this is the current directory.

-outdir

Specifies the directory to which the TC XML file is generated. The file name is automatically generated, using the following naming convention:

BulkAttrOut_1.xml, BulkAttrOut_2.xml, BulkAttrOut_3.xml

This argument is optional. If not specified, the current directory is used.

-batchsize

Specifies the number of objects to update per TC XML file.

This argument is optional. The default value is **800**.

-islandsize

Specifies the number of objects to update per island. An island ties logically related objects together.

For example, a simple island is:

[Item, ItemRev, MasterForm]

The data in low level TC XML is grouped into islands by closure rules, in which the root objects are assigned island IDs and the traversed objects are assigned new island IDs when an **INTER_ISLAND** clause is evaluated.

The following example illustrates an assembly in which every child component is placed in a separate island. The **I** at the end is the predicate for the **INTER_ISLAND** clause.

```
[TYPE.PSOccurrence:CLASS.WorkspaceObject:ATTRIBUTE.child_item:
PROCESS+TRAVERSE:$opt_entire_bom=="&quot;true&quot; &amp;&amp;
$opt_hl_tie=="&quot;false&quot;;I]
```

An example of how island IDs are written in low level TC XML is:

```
<ItemRevision elemId="id7" island_id="4" item_revision_id="A"
items_tag="Q9L5sgKI4ghuD" last_mod_date="2011-11-15T07:57:46Z" object_desc=
"new desc" object_name="my data" object_type="ItemRevision" owning_group="#id4"
owning_user="#id2" parent_uid="Q9L5sgKI4ghuD" puid="gBB5sgKI4ghuD" />

<Item elemId="id7" island_id="4" puid="asd876jerTf"
last_mod_date="2011-11-15T16:03:30Z" object_desc="new description" item_id="003293"/>

<Item elemId="id8" island_id="4" puid="xgr23hgfg43"
last_mod_date="2011-11-15T16:03:30Z" author="John Yates" item_id="003423"/>
```

This argument is optional. The default value is **100**.

-untransformed

Exports the original values of the specified attributes.

This argument is optional.

-uidfile

Specifies the full path to the input XML file containing a list of object UIDs.

This argument is optional. If specified, it suppresses the function of the **-cond_prop** argument.

-h

Displays help for this utility.

ENVIRONMENT

As specified in *Configuring utilities* in the *Utilities Reference*.

FILES

As specified in *Log files* in the *Utilities Reference*.

RESTRICTIONS

The IDSM server must be running when you use this utility.

You must use either the **-inputfile** argument, or the **-type**, **-update_prop**, **-update_value**, **-cond_prop** and **-cond_value** arguments.

Note The condition arguments are the equivalent of the where clause in an SQL phrase. Running this utility without specifying the **-cond_prop** and **-cond_value** arguments it technically possible, but impractical.

EXAMPLES

- To update the attributes of certain properties as specified in the **propFile.xml** file, enter the following command on a single line.

```
attribute_export -u=infodba -p=infodba -g=dba-inputfile=
d:\propFile.xml -log=logFile.txt
```

- To update **prop3** to **value3** and **prop4** to **value4** for **type1** objects, when **prop1** equals **value1** and **prop2** is greater than **value2**, enter the following command on a single line.

```
attribute_export -u=infodba -pf=d:\password.txt -g=dba
-type=type1 -cond_prop=prop1 -cond_value="value1" -cond_prop=prop2
-cond_value="value2" -cond_operator="GT" -update_prop=prop3 -update_value=
"value3" -update_prop=prop4 -update_value="value4"
```

- To update **prop3** to **value3** and **prop4** to **value4** for **type1** objects, when **prop1** equals **value1** and **prop2** is greater than **value2** in batches of 400, enter the following command on a single line.

```
attribute_export -u=infodba -pf=d:\password.txt
-g=dba -type=type1 -cond_prop=prop1 -cond_value="value1" -cond_prop=prop2
-cond_value="value2" -cond_operator="GT" -update_prop=prop3 -update_value=
"value3" -update_prop=prop4 -update_value="value4" -batchsize=400
```

- To update an **object description** to **red Desc** and the **object name** to **This is a new object name** for all items where the **object_name** property is **my object** and the **str_attr** property is **red,white,yellow**, enter the following command on a single line.

```
attribute_export -u=infodba -p=infodba -g=dba
-type=Item -update_prop=object_desc -update_value="red Desc"
-update_prop=object_name -update_value="This is a new obj name"
-cond_prop=object_name -cond_value="my obj" -cond_value="str_attr"
-cond_value="red,white,yellow"
```

- To update an **object description** to **blue Desc** and the **int_VLA** to **10,3,0,99** for all datasets where the **object_name** property is **TextData_es_ES** and the **last_mod_date** property is **01-DEC-2011 00:00** or greater, enter the following command on a single line.

```
attribute_export -u=infodba -p=infodba -g=dba
-type=Dataset -update_prop=object_desc -update_value="blue Desc"
-update_prop="int_VLA" -update_value="10,3,0,99" -cond_prop=object_name
-cond_value="TextData_es_ES" -cond_prop="last_mod_date" -cond_value=
"01-DEC-2011 00:00" -cond_operator="GE"
```

- To update the **char_VLA** to **w,y,d,k,a** for all items where the **owning_user** property is **AsL5bfuW4ghudD**, enter the following command on a single line.

```
attribute_export -u=infodba -p=infodba -g=dba -type=Item -update_prop=
"char_VLA" -update_value="w,y,d,k,a" -cond_prop=owning_user
-cond_value="AsL5bfuW4ghudD"
```

batch_export_translate_import

Exports, translates, and/or imports the named reference of a given type attached to the specified dataset type that is attached to a specified item revision with the given relation type. For example, this utility could be used to export **.wire** files (named references) from the **ALIAS_PROJECT** dataset attached to a specified **CORP_CriteriaRevision** item revision with an **IMAN_specification** relation to the directory specified by the **-output_path** directory.

This utility works in one of the following modes:

- **Export mode (-e)**
Exports datasets from Teamcenter.
- **Import mode (-i)**
Imports datasets to Teamcenter.
- **Export, translate, import mode (-eti)**
Exports, translates, and imports the dataset back in to Teamcenter.

SYNTAX

```
batch_export_translate_import [-u=user-id {-p=password | -pf=password-file}  
-g=group] -infile=input-file -output_path=output-path  
-translator=translator-executable -e -i -eit -nolog [-h]
```

ARGUMENTS**-u**

Specifies the user ID.

This is generally **infodba** or another user with administration privileges. If this argument is used without a value, the operating system user name is used.

Note

If Security Services single sign-on (SSO) is enabled for your server, the **-u** and **-p** arguments are authenticated externally through SSO rather than being authenticated against the Teamcenter database. If you do not supply these arguments, the utility attempts to join an existing SSO session. If no session is found, you are prompted to enter a user ID and password.

-p

Specifies the password.

If used without a value, the system assumes a null value. This argument is mutually exclusive with the **-pf** argument.

If this argument is not used, the system assumes the *user-ID* value to be the password.

-pf

Specifies the password file. If used without a value, the system assumes a null value. If this argument is not used, the system assumes the *user-ID* value to be the password.

For more information about managing password files, see [Manage password files](#).

This argument is mutually exclusive with the **-p** argument.

-g

Specifies the group associated with the user.

If used without a value, the user's default group is assumed.

-infile

Specifies the input file for exporting, translating, and importing a list of datasets.

-i

Specifies that the utility performs only the batch import. The input file format is as follows:

```
item-id|rev-id|dataset-type|relation|reference-type|dataset-uid|
new-dataset-name|path-of-file-to-be-imported
```

-e

Specifies that the utility performs only batch export. The file format for exporting a given list of datasets is as follows:

```
item-id|rev-id|dataset-type|relation|reference-type|dataset-uid
```

-eti

Specifies that the utility export, translate, and import a list of datasets.

-translator

Specifies the translator executable or batch file used to translate the exported files.

-nolog

Specifies that a log file will not be generated when the utility is run.

-h

Displays help for this utility.

ENVIRONMENT

As specified in [Manually configuring your environment for Teamcenter utilities](#). In addition, the log file is created in the local directory specified by either the **TMP** or **TEMP** environment variable.

FILES

As specified in [Log files produced by Teamcenter](#).

RESTRICTIONS

None.

EXAMPLES

- To export the list of given datasets to the **C:\temp** directory, enter the following command on a single line:

```
batch_export_translate_import -u=user-name -p=password -g=group-name -e
-infile=input-file-with-dataset-details -output_path=c:\temp
```

- To import the list of given datasets, enter the following command on a single line:

```
batch_export_translate_import -u=user-name -p=password -g=group-name -i
-infile=input-file-with-dataset-details -output_path=c:\temp
```

- To translate the list of given datasets, enter the following command on a single line:

```
batch_export_translate_import -u=user -p=password -g=group-name
-eti -infile=input-file-with-dataset-details
```

```
-translator=input-translator-file -output_path=c:\temp
```

cleanup_ic_objects

Reverts changes made to an assembly as part of an incremental change (IC) object. This utility analyzes the incremental change object and cleans up the database to nullify the impact of the IC changes. For example, if a PLM XML file that is updating an existing structure in the database is imported using an IC object, this utility can be used to revert the effect the import. This can be very helpful in cases where the PLM XML import failed and the data is not in a consistent state.

Note This utility does not restore the database to its original state before the import.

This utility can revert the following changes:

- Adding, removing, or modifying a BOM line.
 - o Add an existing item as a BOM line.
 - o Add an existing item as a BOM line and occurrence group child.
 - o Add an existing item with GDE children as a BOM line and occurrence group child.
 - o Add an existing connection as a BOM line and occurrence group child and connect it to GDE BOM lines.
 - o Modify a BOM line property.
 - o Remove an item BOM line.
 - o Remove an item BOM line with GDE children.
 - o Remove a connection BOM line.
- Adding, removing, or modifying a GDE line.
 - o Add a GDE line.
 - o Modify a GDE line property.
 - o Remove a GDE line.
- Adding, removing, or modifying attachments (forms/datasets).
 - o Attach a form/dataset to an item BOM line.
 - o Attach a form/dataset to an item BOM line in absolute occurrence context.
 - o Attach a form to a GDE line in absolute occurrence context.
 - o Attach a form to a connection BOM line in absolute occurrence context.
 - o Modify a BOM line form/dataset.
 - o Modify a BOM line form/dataset in absolute occurrence context.

- o Remove a BOM line form/dataset.
- o Remove a BOM line form/dataset in absolute occurrence context.

A given IC object may contain many ICEs (incremental change elements). Each ICE specifies the change type that was made (add/remove) and the affected object of the change.

For add scenarios, the change type is **IC_add**.

- When adding a BOM line/GDE line, the affected object is **PSOccurrence/GDEOccurrence**. This utility deletes the ICE and **PSOccurrence/GDEOccurrence**.
- When attaching a dataset/form to a BOM line, the affected object is **ImanRelation**. This utility deletes the ICE and **ImanRelation**. The secondary objects of the relation are not deleted.

For modify scenarios, the change type is **IC_add**.

- When modifying a BOM line property, the affected object is **AbsOccData**. This utility deletes the **AbsOcc** data.
- When modifying a BOM line attachment, the affected object is a copy of the attachment (dataset/form) with the modifications. This utility deletes the modified copy.

For remove scenarios, the change type is **IC_remove**.

- The affected objects point to the objects that are removed from the structure. This utility deletes the ICE. It automatically restores the removed objects.

SYNTAX

```
cleanup_ic_objects [-u=user-id {-p=password | -pf=password-file} -g=group]
{ [-ic_id=IC-object-item-ID | -key=IC-object-search-key]
[-ic_rev_id=IC-object-rev-ID] } [-dryrun] [-verbose] [-start_date=start-date]
[-end_date=end-date]
```

ARGUMENTS

-u

Specifies the user ID.

This is generally **infodba** or another user with administration privileges. If this argument is used without a value, the operating system user name is used.

Note

If Security Services single sign-on (SSO) is enabled for your server, the **-u** and **-p** arguments are authenticated externally through SSO rather than being authenticated against the Teamcenter database. If you do not supply these arguments, the utility attempts to join an existing SSO session. If no session is found, you are prompted to enter a user ID and password.

-p

Specifies the password.

If used without a value, the system assumes a null value. This argument is mutually exclusive with the **-pf** argument.

If this argument is not used, the system assumes the *user-ID* value to be the password.

-pf

Specifies the password file. If used without a value, the system assumes a null value. If this argument is not used, the system assumes the *user-ID* value to be the password.

For more information about managing password files, see [Manage password files](#).

This argument is mutually exclusive with the **-p** argument.

-g

Specifies the group associated with the user.

If used without a value, the user's default group is assumed.

-ic_id

Specifies the item ID of the incremental change object whose associated data will be deleted.

-key

Specifies the search key of the incremental change object. Use the following format:

```
[keyAttr1=keyVal1] [,keyAttr2=keyVal2]...[,keyAttrN=keyValN]
```

To find the key of an object, use the [get_key_string](#) utility.

For more information, see the [Business Modeler IDE Guide](#).

-ic_rev_id

Specifies the revision ID of the incremental change object whose associated data will be deleted.

If this argument is not specified, the latest revision will be used.

-dryrun

Deletes nothing. Objects are searched and a report is created.

-verbose

Prints extra information about the objects being deleted.

You can combine the **-dryrun** and **-verbose** options as follows:

- No option: Prints errors only.
- **-verbose** option only: Prints maximum information, such as what is being searched, how many objects were found, type of objects being deleted, and success/failure message after deletion.
- **-dryrun** option only: Prints the number of objects found and the type of objects that will be deleted in a normal run.
- Both options: Prints all of the information from the **-verbose** option except the success/failure message.

-start_date

Changes made after the specified date and time are reverted. Use the format **DD-MMM-YYYY HH:MM:SS**. For example, **01-Jan-1970 00:00:00** is January 1, 1970.

If no time is specified, 00:00:00 is used.

-end_date

Changes made after the specified date and time are reverted. Use the format **DD-MMM-YYYY HH:MM:SS**. For example, **01-Jan-1970 00:00:00** is January 1, 1970.

If no time is specified, 00:00:00 is used.

-h

Displays help for this utility.

ENVIRONMENT

As specified in [*Manually configuring your environment for Teamcenter utilities*](#).

FILES

As specified in [*Log files produced by Teamcenter*](#).

RESTRICTIONS

None.

convert_replica_files_to_stubs

Converts existing replica Teamcenter file (**ImanFile** objects) to **pom_stub** objects at the specified remote site, all remote sites, or specified objects at all remote sites; it purges the corresponding operating system volume files to conserve space. It can also populate the file server cache (FSC) with replica files.

SYNTAX

```
convert_replica_files_to_stubs [-u=admin-id {-p=password |  
-pf=password-file} -g=dba]  
{[-by_site_name=remote-site-id | ALL] | [-by_plmxml=file-name]} [-verbose]  
[-populate_cache] [-query] [-batch_size=size-value]  
[-h]
```

ARGUMENTS

-u

Specifies the user ID. The user must have administrative privileges.

If this argument is used without a value, the operating system user name is used.

-p

Specifies the user's password.

If used without a value, the system assumes a null value. If this argument is not used, the system assumes the *user-ID* value to be the password.

This argument is mutually exclusive with the **-pf** argument.

-pf

Specifies the password file. If used without a value, the system assumes a null value. If this argument is not used, the system assumes the *user-ID* value to be the password.

For more information about managing password files, see [Manage password files](#).

This argument is mutually exclusive with the **-p** argument.

-g

Specifies the group associated with the user. The group value must be **dba** to run this utility.

-by_site_name

Specifies the name of the remote site where replica files are replaced with stubs. If **ALL** is specified, replica files at all remote sites are replaced. This argument cannot be combined with the **-by_plmxml** argument.

-by_plmxml

Specifies the name of a PLM XML file that contains replica objects to be replaced by stubs. Only the objects listed in the file are replaced.

You must generate the PLM XML file specified in this argument using the **ConfiguredDataExportDefault** or **justDatasetsOut** transfer modes. Other transfer modes are not supported.

This argument cannot be combined with the **-by_sitename** argument.

-verbose

Includes additional information about the process in the utility's output.

-populate_cache

Populates the FSC with replica files at the sites where replica objects are replaced with stub objects.

-query

Returns the number of objects that will be converted to stubs.

-batch_size

Specifies the number of objects in each batch that is processed. If this argument is omitted, all objects are processed in a single batch.

-h

Displays help for this utility.

ENVIRONMENT

As specified in *Configuring utilities* in the *Utilities Reference*.

FILES

As specified in *Log files* in the *Utilities Reference*.

RESTRICTIONS

The IDSM server must be running when you use this utility.

EXAMPLES

- Replace all replica objects at all sites with stub objects:

```
convert_replica_files_to_stubs -u=infodba -p=password -g=dba  
-by_site_name=ALL
```

- Replace replica objects at the cologne site and populate the site's FSC:

```
convert_replica_files_to_stubs -u=infodba -pf=pwfile -g=dba  
-by_site_name=cologne -populate
```

- Determine the number of replica objects that will be replaced at a site:

```
convert_replica_files_to_stubs -u=infodba -p=password  
-by_site_name=cologne -query
```

- Replace all replica objects at all sites with stub objects processed in batches of 200 objects:

```
convert_replica_files_to_stubs -u=infodba -pf=pwfile -g=dba  
-by_site_name=ALL -batch_size=200
```

- Replace replica objects in the **stub_replicas.xml** file at all sites with stub objects processed in batches of 200 objects:

```
convert_replica_files_to_stubs -u=infodba -pf=pwfile -g=dba  
-by_plmxml=stub_replicas.xml -batch_size=200
```

database_verify

Compares database schema, Teamcenter types, tools, release statuses, and units of measure between two specified Multi-Site Collaboration sites and generates a report of any database discrepancies.

You can use this utility to query types and classes from a specified remote site and create a local dataset named **TCTYPES_SITE***siteid* containing those mappings. There is one dataset for each remote site defined with the **-site** argument. You can also create and update the local dataset of remote class and types mappings for a specified remote site. Run this utility whenever there are changes for the POM transmit file of a specified site.

SYNTAX

```
database_verify [-u=user-id {-p=password | -pf=password-file} -g=group]
-from=site-name1 -to=site-name2
[-schema] [-type] [-tool] [-status] [-uom] [-all] [-output=file-name]
-site=site-name -force -offline [-filename=file-name] [-v] [-h]
```

ARGUMENTS

-u

Specifies the user ID.

This is generally **infodba** or another user with administration privileges. If this argument is used without a value, the operating system user name is used.

Note

If Security Services single sign-on (SSO) is enabled for your server, the **-u** and **-p** arguments are authenticated externally through SSO rather than being authenticated against the Teamcenter database. If you do not supply these arguments, the utility attempts to join an existing SSO session. If no session is found, you are prompted to enter a user ID and password.

-p

Specifies the password.

If used without a value, the system assumes a null value. This argument is mutually exclusive with the **-pf** argument.

If this argument is not used, the system assumes the *user-ID* value to be the password.

-pf

Specifies the password file. If used without a value, the system assumes a null value. If this argument is not used, the system assumes the *user-ID* value to be the password.

For more information about managing password files, see [Manage password files](#).

This argument is mutually exclusive with the **-p** argument.

-g

Specifies the group associated with the user.

If used without a value, the user's default group is assumed.

-from

Specifies a Teamcenter database site to be verified.

-to

Specifies a Teamcenter database site to be verified.

-output

Specifies the output format. The report is output to a file if a file name is specified. If not, the report is displayed in a shell.

-schema

Compares schema between the two sites.

-type

Compares types between the two sites.

-tool

Compares tools between the two sites.

-status

Compares release status types between the two sites.

-uom

Compares units of measure between the two sites.

-notetype

Compares note types.

-all

Compares classes, types, tools, status types and units of measures between the two sites. This is the default if no argument is supplied.

-site

Specifies the site name where types and classes would be persisted locally. If the value of this argument is set to **ALL**, the utility generates these datasets for all sites in the database.

-force

Generates the type-class mappings file even when the POM transmit files for the remote and local sites have not changed.

-offline

Specifies the site identified with the **-site** argument is offline. If you specify this argument, you must also specify the **-filename** argument.

-filename

Specifies the file name generated by this utility from the **-site** argument. This argument is required if the **-offline** argument is specified.

-v

Runs utility in verbose mode. Displays maximum amount of information. Typically, nonverbose utility sessions only display error messages.

-h

Displays help for this utility.

ENVIRONMENT

As specified in the [Manually configuring your environment for Teamcenter utilities](#).

FILES

As specified in the *Log files produced by Teamcenter*.

RESTRICTIONS

- To use this utility, you must be a user with system administration privileges or be granted authorization by a user with system administration privileges.
- The **-from** and **-to** arguments must be specified.

EXAMPLES

None.

data_share

Used for various Multi-Site Collaboration operations, such as publishing and unpublishing objects collectively and sending objects to remote sites. It can be used as a deployment tool during the initial Multi-Site Collaboration implementation phase or as a day-to-day tool for performing functions that previously were available only through the user interface. This utility is especially helpful in setting up and maintaining a hub configuration.

The behavior of this utility is controlled by the [TC_force_remote_sites_exclude_files](#) preference. If this preference is set to **true**, the replica files are stored in the remote site FMS server cache (FSC); otherwise, the replica files are stored in the remote system volume.

The utility also supports TC XML transfers for of 4th Generation Design (4GD) data. The 4GD relation data mapping is controlled by the [TC_cms_relation_optset_map](#) preference. You use this preference when you want to control the relations that are included or excluded when replicating a 4GD object.

For more information about controlling relations using this preference, see the [Preferences and Environment Variables Reference](#).

This utility supports part family templates and part family members.

Use this utility to:

- Mass publish objects to one or more Object Directory Services (ODS) sites.
- Mass unpublish objects from one or more ODS sites.
- Publish or unpublish an entire assembly.
- List ODS sites currently defined in the local database and authorized for publication.
- Send objects to other sites.
- Delete obsolete publication records at the ODS.
- Check current status of authorized publication sites.
- List ODS sites to which an object is published.
- Import an item from a remote site.
- Export 4GD data in low-level TC XML format.

Data can be input to this utility in the following forms:

- Input file
- Folder name
- Object ID template

When sending objects to a specific user and/or group at a remote site using the **-owning_user** and **-owning_group** arguments, the following rules apply:

- If both the specified user and group exist at the importing site, the imported objects are owned by the user and group regardless of whether or not the user is a member of the group.
- If only the user is specified or if the group is specified but does not exist at the importing site, the user's default group at the importing site is the owning group of the imported objects.
- If only the group is specified or if the user is specified but does not exist at the importing site, the user context of the remote IDSM process is the owning user of the imported objects.

SYNTAX

```
data_share [-u=user-id {-p=password | -pf=password-file} -g=group]
-f=function [-site=remote-site-name1 -site=remote-site-name2... ]
[-owning_user=remote-user] [-owning_group=remote-group]
{-item_id=item-id | template} | -folder=folder-name |
-name=workspace-object | -filename=input-file |
[-key=keyAttr1=keyVal1,keyAttr2=keyVal2...,keyAttrN=keyValN |
-itemKeyFile=file-name] | [-itemRevisionKeysFile=file-name]]
[-class=wso-class-name | -classoffile=class-name]
[-include=relation-type1 -include=relation-type2...]
[-exclude=relation-type1 -exclude=relation-type2...]
[-revision-selector | -rev=rev-id ]
[-include_bom] [-transfer] [-attach] [-exclude_files] [-latest_ds_version]
[-exclude_folder_contents] [-include_bc] [-include_supercedures]
[-include_pfmembers] [-include_pftemplates] [-pf_bom_treatment=option]
[-continue_on_error] [batch_size=number-objects-per-batch]
[-report=report-file-name] [-user=user-id] [-group=group-name]
[-error_file=error-file-name] [-exclude_variant_options]
[-batch_variant_options] [-batch_objects=class-for-deferred-objects]
[-batch_file=file-name-for-deferred-objects]
[-include_dist_comp] [-log]
[-checkpoint [-compress_ind_files=S | I | N] ] [-transaction_id=transaction-id]
[-status]
[-cleanup_transaction [-transaction_id=transaction-id |
-before_last_process_date=date] ]
[-restart] [-commit_ixr] [-list_transactions]
[-low_level [-optionset | -optionset=optionset-name] [-de_incl_rlz_bom]
[-workset_include_relz_de] [-4gd_id=object-id -class=4gd-class-name] ]
[-all_roles] [-all_subgroups] [-all_groupmembers] [-h]
```

ARGUMENTS

Note Entries in parentheses are accepted abbreviations for arguments.

-u

Specifies the user ID.

This is generally **infodba** or another user with administration privileges. If this argument is used without a value, the operating system user name is used.

Note If Security Services single sign-on (SSO) is enabled for your server, the **-u** and **-p** arguments are authenticated externally through SSO rather than being authenticated against the Teamcenter database. If you do not supply these arguments, the utility attempts to join an existing SSO session. If no session is found, you are prompted to enter a user ID and password.

-p

Specifies the password.

If used without a value, the system assumes a null value. This argument is mutually exclusive with the **-pf** argument.

If this argument is not used, the system assumes the *user-ID* value to be the password.

-pf

Specifies the password file. If used without a value, the system assumes a null value. If this argument is not used, the system assumes the *user-ID* value to be the password.

For more information about managing password files, see [Manage password files](#).

This argument is mutually exclusive with the **-p** argument.

-g

Specifies the group associated with the user.

If used without a value, the user's default group is assumed.

Caution

For HTTP enabled sites, remote site operations log on using the default group for the user supplied with the **-u** argument. Any value supplied with the **-g** argument is ignored.

-f

Specifies the function to be performed; you must specify one of the following:

send

Sends objects to the specified remote sites.

The objects to send are determined by the **-item_id**, **-folder**, **-filename**, **-4gd_id**, **-all_subgroups**, **-all_roles**, or **-all_groupmembers** arguments.

publish (pub)

Publishes objects to the given ODS sites. The objects to publish are determined by the **-item_id**, **-folder**, or **-filename** arguments. Cannot be used with the **-all_subgroups**, **-all_roles**, or **-all_groupmembers** arguments.

unpublish (unp)

Unpublishes objects from the given ODS sites. The objects to unpublish are determined by the **-item_id**, **-folder** or **-filename** arguments. Cannot be used with the **-all_subgroups**, **-all_roles**, or **-all_groupmembers** arguments.

delete_pubrec (dpr)

Deletes obsolete publication records for the specified object from the local database. This must be run at the ODS site containing the publication record to be deleted. Only privileged users may use this function. Requires the **-item_id**

argument with specific item ID; no wildcards or other arguments are supported with this function.

Note To be used only if the master object has been deleted but publication records still exist at the ODS site.

register (reg)

Registers item IDs to the central item ID registry.

unregister (unreg)

Unregisters item IDs from the central item ID registry. The register and unregister functions must be supplied with the **-item_id** or **-filename** argument.

delete_exprec (dxr)

Deletes export records for the specified sites for objects listed in the text file identified by the **-filename** and **-classoffile** arguments. It does not traverse item structure. Only privileged users may use this function.

Note To be used only as a last resort after attempting to delete export records using the **-verify** argument of the [data_sync](#) utility.

list_ods (lo)

Lists the authorized ODS sites, which consist of the default ODS site and the sites specified by the [ODS_publication_sites](#) preference.

check_ods (co)

Lists the availability of authorized ODS sites.

list_pub_info (lpi)

Lists publication information about objects. Must be run at the owning site.

find_duplicates (fd)

Compares all of the item IDs at the remote site specified by the **-site** argument. The item IDs searched for may be filtered with the **-item_id**, **-created_before**, and **-created_after** arguments. The output may be directed to a file using the **-report** argument. The output is formatted to **csv** style, using comma-separated values.

-created_before

Restricts searches for duplicate items to those created at the target site before the specified date.

-created_after

Restricts searches for duplicate items to those created at the target site after a specified date.

list_remote_co (lremco)

Lists master objects that are checked out by remote users based on the specified user ID, group name, and site name.

list_replica_co (lrepc)

Lists replica objects that are checked out from a remote site based on the specified user ID, group name, and site name.

cancel_remote_co (cremco)

Cancels all remote checkouts based on the specified user ID, group name, and site name.

cancel_replica_co (crepco)

Cancels replica checkouts based on the specified user ID, group name, and site name. Canceling a replica checkout also cancels the remote checkout at the owning site.

remote_import (ri)

Imports the item specified by the **-item_id** argument from the owning site or the site specified by the **-site** argument. If the item is a replica at the local site, it is imported from the owning site and any site specified in the command is ignored.

Note Wildcard characters cannot be used in the **-item_id** argument.

You can also use this argument to import a list of items from an input file designated by the **-filename** argument. The input file must contain UIDs for the items to be imported, and the **-classoffile** argument value must be set to **Tagstring** when using an input file. You can use the **sync_on_demand** utility to generate a file that contains UIDs of items of an assembly enclosed within square brackets ([]) or other designated separator. You can then write a script to collect the UIDs into the input file.

The **data_share** arguments related to variants and line of usage (LOU) cannot be used with the **remote_import** argument.

-site

Specifies the name of the site to which objects are published or from which they are unpublished. It can be given multiple times in a command line.

-owning_user (ou)

Specifies the user ID of the user at the remote sites to which the objects are sent. The specified user owns the objects being sent. See [Restrictions](#).

-owning_group (og)

Specifies the name of the group at the remote sites to which the objects are sent. The group owns the objects being sent. See [Restrictions](#).

-item_id (item)

Specifies the item ID or template of items to process. It is mutually exclusive with the **-folder**, **-filename**, **-keyFileName**, **-name** and **-key** arguments. It is required for the **-delete_pubrec** argument.

-folder (fl)

Specifies the name of a Teamcenter folder containing the list of objects to process. It is mutually exclusive with the **-name**, **-filename**, and **-item_id** arguments.

Note If the **-include_bom** argument is used with the **-folder** argument, only the **ItemRevision** objects (and objects related to them) in the folder are replicated. The folder itself is not replicated.

-name

Specifies the name of a single workspace object to be preprocessed. If not an item, use the **-class** argument to specify the class of the object. It is mutually exclusive with the **-folder**, **-filename**, and **-item_id** arguments.

For organization objects, this argument accepts the following attributes:

- **User** objects require a **user_ID** attribute.
- **Role** objects require a **role_name** attribute.
- **Group** objects require the groups full name that uniquely identifies the group.
- **Person** objects require a **user_name** attribute.

Note You can use a text file containing a list of all organization objects of the same type that you want to export using the **-classoffile** and **-filename** arguments.

-filename (fn)

Specifies the name of the input file containing the list of IDs or names of objects to process. File entries are treated as IDs for **Items** and **ItemRevisions** objects and as names for other classes of objects. It is mutually exclusive with the **-name**, **-folder**, **-item_id**, **-key**, **-itemKeysFile**, and **-itemRevisionKeysFile** arguments.

If the input file contains names, the **-classoffile** argument is required.

-key

Specifies the keys of the items to process, the template of the item keys, or the 4GD object key. It is mutually exclusive with the **-item_id** argument. Use the following format:

```
keyAttr1=keyVal1,keyAttr2=keyVal2...,keyAttrN=keyValN
```

To find the key of an object, use the [get_key_string](#) utility.

For more information, see the [Business Modeler IDE Guide](#).

-itemKeyFile

Specifies the name of the input file containing the list of item key strings of the items you want to update. The following listing shows sample content of a file for updating a list of items:

```
item_id=export_001
item_id=M2Item1_001,object_name=M2Item_name1,object_type=M2Item1
```

If the item key file has 4GD object key strings, the corresponding 4GD class must be supplied using the **-classoffile(cof)=** argument. The following listing shows sample content of a file for updating 4GD data:

```
4gd_id=DE_Export_001
4gd_id=DE00001_ID,object_name=DE00001_Name,object_desc=DE00001_Desc
```

The **4g_id** entry maps to the corresponding unique ID of the 4GD class, for example:

```
Class=Cpd0DesignElement, 4gd_id=cpd0_design_element_id
```

-itemRevisionKeysFile

Specifies the name of the file containing the keys of the item revisions to process. It is mutually exclusive with the **-item_id** argument.

-class (cl)

Specifies the Teamcenter class of the object specified by the **-name** or **-4gd_id** argument. This argument is valid only with the **-name** or, when the **-low_level** argument is specified, with the **-4gd_id** argument. The default class is **Item**.

For organization objects, this argument accepts **Role**, **User**, **Group**, and **Person** classes.

-classoffile (cof)

Specifies the class of the objects listed in the input text file given with the **-filename** argument. If not defined, the default class is **Item**. It is required if the input file has names instead of IDs.

For organization objects, this argument accepts **Role**, **User**, **Group**, and **Person** classes.

-include

Specifies a relation type to include. This can be specified multiple times in a command line. The database name (not the display name) of the relation type must be used.

-exclude

Specifies a relation type to be excluded from the operation. This can be specified multiple times in a command line. The database name (not the display name) of the relation type must be used.

Note

The list of relation types to be included is determined by either the **TC_relation_required_on_export** (export without transferring ownership) or **TC_relation_required_on_transfer** (export with transfer of ownership) preferences.

For more information about Multi-Site Collaboration preferences, see the *Preferences and Environment Variables Reference*.

The **IMAN_master_form** relation cannot be used as an argument for the **-exclude** argument. The following relations cannot be used as an argument for the **-exclude** argument unless they are not included as a value in the preference for the location:

- **IMAN_requirement**
- **IMAN_specification**

-revision-selector**Note**

If no revision selector is specified, the default selector is **all_revisions**.

Identifies the item revisions to send. It is also used as the revision rule for identifying components when processing assemblies. When used with the **-include_bom** argument while publishing or unpublishing, it determines which revisions' BVR to follow in traversing the assembly tree. The valid revision selectors are as follows:

all_revisions

Sends all revisions. it is not valid when publishing.

latest_revision

Processes only the latest revision regardless of release status. This is the default if no revision selector is given when publishing or unpublishing.

selected_revision

Process only the selected revision.

latest_working

Processes only the latest working revision.

latest_released

Processes only the latest released revision with any release status.

latest_working_or_any

Sends only the latest working revision. If none, the latest released revision is processed.

release_status

Processes only the latest released revision with the given release status.

all_released_revs

Sends all revisions with a release status; it is not valid when publishing.

-rev

Specifies the ID of a specific item revision to be sent to a remote site. It is valid only with the **-item_id** argument and with the **-send** function, and is mutually exclusive with revision selectors.

-include_bom (bom)

Includes assembly components when sending, publishing, or unpublishing. A revision selector is required when publishing or unpublishing an assembly; if no revision selector is given, the **latest_revision** selector is used as the default. When sending, the default selector is **all_revisions**.

If not specified, this argument defaults to **off**.

This argument does not traverse the component relationships of subassemblies. This allows you to send the subassemblies in separate transactions. Using multiple, simultaneous transactions to transfer very large assemblies and their subassemblies separately provides improved scalability and performance.

Note

This argument, although similar to the rich client **Include Entire BOM** remote export option, may not export the same set of objects. The **Include Entire BOM** option traverses all components, subassemblies, and subassembly component relationships that can result in unacceptable performance for very large assemblies.

-transfer (tf)

Transfers site ownership when sending objects. Site ownership is not transferred by default.

-attach (att)

Attaches an object to the appropriate parent item or revision at the receiving site when sending an attachment with transfer of site ownership. Use this for situations in which you attach a dataset to a replica, such as a JT file, and you want to send

the JT file to the owning site with transfer of site ownership and attached to the appropriate parent item or revision.

-exclude_files (exf)

Excludes dataset files.

-latest_ds_version (ldv)

Sends only the latest dataset version. Unless this argument is specified, all dataset versions are sent.

-exclude_folder_contents (efc)

Excludes the contents of folders being exported.

-include_bc (ibc)

Identifies the **BomChange** objects associated with the affected assemblies to send. If not specified, **BomChange** objects are not sent.

-include_supercedures (isc)

Identifies the supercedure objects associated with the **BomChange** objects to send. If not specified, supercedure objects are not sent.

-include_pfmembers

Identifies the related part family members to be exported when handling part family templates.

-include_pftemplates

Identifies the related part family template to be exported when handling part family members.

-pf_bom_treatment

Identifies the part family objects associated with the assemblies to be exported. The argument must be used in conjunction with the **-include_bom** argument. Valid arguments are:

-members

Includes part family member components present in the assembly.

-templates

Includes part family template rather than part family member components.

-all

Includes both the part family member components and templates.

-none

Includes neither the part family member components nor the templates.

-continue_on_error (con)

Specifies processing is continue if there is an error on an optional object such as a reference or manifestation. This argument is not valid when transferring site ownership.

Outputs the error in a report file and continues processing the other items. The **-report** argument must be specified to see the error.

-batch_size (bs)

Specifies the number of objects per batch; a new process is created per batch. The default batch size is 1000. It must be a positive integer. This is useful when processing thousands of objects, because it helps avoid memory and disk space shortage problems.

-report (rep)

Specifies to output a report to the specified file.

-user

Specifies the user ID.

-group

Specifies the group name.

-error_file (err)

Specifies the name of the output error report when sending assemblies.

-exclude_variant_options (evo)

Indicates all variant options are to be excluded during a send operation.

-batch_variant_options (bvo)

Indicates all variant options are sent separately in batch mode.

-batch_objects (bo)

Indicates all objects of the given classes are sent separately in batch mode. Separate each class name with a comma. The list cannot contain spaces. The following table is a list of supported classes for this argument:

Dataset	ImanRelation	PSOccurrence
Folder	MEAppearancePathNode	VariantExpression
Form	NamedVariantExpression	VariantExpressionBILock

-batch_file (bof)

Indicates all objects of the classes in the specified file are sent separately in batch mode. List each class name separately on a line in the file. For a list of supported classes for this argument, see the table for the description of the **batch_objects** argument.

-include_dist_comp

Includes distributed components during import. This argument is valid only in conjunction with the **-remote_import** argument.

-log

Specifies detailed log information is written to the log file.

-checkpoint (cp)

Initiates a checkpoint transaction; that is, a transaction that can be restarted at the point of failing.

This argument is valid only with **send** function. It is not valid with the **-transfer** argument.

If a noncheckpoint operation is initiated for multiple target sites and some target sites are not currently available based on a preliminary availability check,

Teamcenter sends a message to **stdout** to notify the user about unavailable sites, removes unavailable sites from the target site list, and then performs the operation for the available sites.

compress_ind_files (cif)

Specifies compression mode to use to compress files in the export directory during a checkpoint transaction. If not specified, it creates a single large ZIP file. This argument is valid only with the **-checkpoint** argument. Valid values are:

- **S**
Creates a single large ZIP file.
- **I**
Creates a ZIP file for each individual file, resulting in multiple ZIP files.
- **N**
No files are compressed.

-transaction_id (trid)

Specifies a 14-character transaction ID for a given checkpoint-related operation or for a fast sync transaction if used with the **-low_level** argument. A fast sync transaction provides improved performance when synchronizing data using the [data_sync](#) utility.

-status (stat)

Displays the status of a given transaction ID.

If the **-site** switch is given, only status of the given sites are displayed. If no **-site** switch is given, the status returned depends on whether the command is given at the site that initiated the checkpointed transaction or the site is the receiving end of the transaction. If the command is given at the initiating site and no **-site** switch is given, the status of the local site and all target sites is returned.

-cleanup_transaction (ct)

Removes transient data generated during a checkpoint transaction or fast sync transactions if use with the **-low_level** argument.

For checkpoint transactions, the transient data consists of the export data and supporting directories and files used to manage the transaction. You must execute this function at a node or host that has direct access to the transfer area where the export was performed (if at initiating site) or where the data was transmitted to by the owning site (if at receiving site). You must also have **delete** access to the operating system directory where the export data is placed within the transfer area. If these conditions are not met, an error message is displayed to **stdout** and the utility returns a nonzero value.

For fast sync transactions, you must specify either the **-transaction_id** or **-before_last_process_date** argument.

-before_last_process_date (blpd)

Specifies the date used to determine which fast sync transactions to clean up. The date must be supplied in the following format:

YYYY-MM-DD:HH:NN:SS

YYYY represents the four-digit year value. MM represents the two-digit month value. DD represents the two-digit day of the month. HH represents a two-digit hour value from 0 to 23. NN represents a two-digit minute value from 0 to 59 and SS represents a two-digit second value from 0 to 59. The DD:HH:NN:SS are optional. If they are not specified, the utility uses 12:00 AM of the specified date.

Valid only with the **-cleanup_transaction** argument.

-restart (rs)

Restarts a given transaction at the point of failure.

Valid only with the **-f=send** function.

-commit_ixr (cmi)

Updates the export records at the owning site once the data is known to have been successfully imported at a target site.

Under normal conditions, the update of the export records are performed automatically by each subprocess that succeeds in completing the send operation to its assigned site. Use this function only if either of the following conditions occur:

- The failure occurs at the importing site, and the user performs the restart using the **item_import** utility.
- The failure occurs after the data is successfully imported by a target site, but a failure occurs just before or during the updating of the export records.

You must use at least one **-site** argument to identify the site or sites for which export records are to be updated.

You must execute this function at a node or host that has direct access to the transfer area where the export was performed (if at initiating site) or where the data was transmitted to by the owning site (if at receiving site). You must also have *read* access to the operating system directory where the export data is placed within the transfer area. If these conditions are not met, an error message is displayed to **stdout** and the utility returns a nonzero value.

-list_transactions (lt)

Lists all uncleaned checkpoint transactions or fast sync transactions if used with the **-low_level** argument. An uncleaned transaction is one in which its transient data has not been deleted from the transfer area using the **cleanup_transaction** function.

- Active transactions can only be detected at the site that initiated it. The receiving end of a transaction is not able to determine if a transaction is active or not.
- The list of inactive transactions initiated by the local site and the list of transactions initiated by remote sites are based only on the contents of the transfer area of the node where this command is executed.
- The list of active transactions initiated by the local site is always complete because it is based on data stored in the local database.

You must execute this function at a node or host that has direct access to the transfer area where the export was performed (if at initiating site) or where the data was transmitted to by the owning site (if at receiving site). You must also have **read** access to the operating system directory where the export data is placed within

the transfer area. If these conditions are not met, an appropriate error message is displayed to **stdout** and the utility returns a nonzero value.

-all_subgroups

Exports all subgroups of the selected group. Parent groups are always exported. This argument is valid only when the **-class** or **-classoffile** argument is set to **Group**.

-all_roles

Exports all roles associated with the selected group. All roles for subgroups are included if the **-all_subgroups** argument is also specified. If not specified, only the default role is exported.

This argument is valid only when the **-class** or **-classoffile** argument is set to **Group**.

-all_groupmembers

When exporting a **User** object, exports all **GroupMember** objects related to the selected user. When exporting groups or groups and subgroups (**-all_subgroups** argument), exports all **GroupMember** objects related to any role that is exported.

You must specify one of the following arguments to use this argument:

- **-class=User**
- **-classoffile=User**
- **-class=Group**
- **-classoffile=Group**

If this argument is not specified when exporting **User** class objects, only the default group related to the user is exported.

If this argument is not specified when exporting **Group** class objects, no **GroupMember** objects are exported.

-low_level

Exports the data using the low-level TC XML functionality. This provides better performance for large data transfers and must be used when exporting 4GD data.

You can replicate 4GD objects to a remote site when you specify the **-low_level** argument with the **send** function. Specify the objects to replicate using the **-class** and **-4gd_id** arguments. You can specify the transfer option set used for the export in the **-optionset** argument. If you do not specify an option set, the utility uses **MultiSiteOptSet** as the default transfer option set value. For 4GD objects, you can also specify the **-de_incl_rlz_bom** and **-workset_incl_relz_de** arguments only when you specify the **-optionset** argument.

-4gd_id

Specifies a 4GD object identifier or 4GD object pattern. The **-class=class-name** argument must be specified with the **-4gd_id** argument.

The utility maps the **-4gd_id** argument to the corresponding unique ID of the 4GD class, for example:

```
Class=Cpd0DesignElement, 4gd_id=cpd0design_element_id
```

A 4GD partition object and 4GD subset definition objects do not have a unique 4GD class ID. Therefore, using **-4gd_id** for partition objects or subset definition objects or may result in transfer of multiple objects.

To export unique partition object use multifield key attributes supplied in the **-key** argument, see [Examples](#).

-optionset

Specifies the name of the transfer option set when sending objects using low-level TC XML functionality (**-low_level** argument). Values of the options listed in the option set govern the object export. The option set must exist at the exporting site. If you do not specify this argument, the utility uses the **MultiSiteOptSet** option set.

-de_incl_rlz_bom

Exports the source objects of a design element (Type:**Cpd0DesignElement**). You must specify the **-low_level** argument and specify a 4GD object using the **-4gd_id**, **-key**, or **-itemKeyFile** argument.

-workset_incl_relz_de

Exports the source objects of a design element (**Cpd0DesignElement**) in a workset (**Cpd0Workset**). You must specify the **-low_level** argument and specify a 4GD object using the **-4gd_id**, **-key**, or **-itemKeyFile** argument.

-h

Displays help for this utility.

ENVIRONMENT

As specified in the [Manually configuring your environment for Teamcenter utilities](#).

FILES

As specified in the [Log files produced by Teamcenter](#).

RESTRICTIONS

- When sending objects to a specific user and/or group at a remote site using the **-owning_user** and **-owning_group** arguments, the following rules apply:
 - To use this utility, you must be a user with system administration privileges or be granted authorization by a user with system administration privileges.
 - If both the specified user and group exist at the importing site, the imported objects are owned by the user and group regardless of whether or not the user is a member of the group.
 - If only the user is specified or if the group is specified but does not exist at the importing site, the user's default group at the importing site is the owning group of the imported objects.
 - If only the group is specified or if the user is specified but does not exist at the importing site, the user context of the remote IDSM process is the owning user of the imported objects.
- If variant options are excluded using the **-exclude_variant_options** argument, it is implied that they cannot be sent separately in batch mode. Therefore, the **-exclude_variant_options** argument cannot be used with either **-batch_variant_options**, **-batch_objects=variant-expression**, or with the **-batch_file** arguments when the given file includes the class name **variantexpression**.
- Any number of objects can be sent separately in batch mode. Class names of objects can be given in a comma-delimited list with the **-batch_objects** argument or listed in a file whose name is specified in the **-batch_file** argument.

4. The **-item_id**, **-name**, **-filename**, **-folder**, **-key**, **-itemKeyFile**, **-itemRevisionKeysFile**, and **-4gd_id** arguments are mutually exclusive

EXAMPLES

Note Required logon information is omitted from the following examples.

- To send a list of items specified in a text file to two sites:

```
data_share -f=send -filename=my_item_list.txt -site=Site1 -site=Site2
```

- To send a list of items specified in a text file and output a report; continue processing if a nonfatal error is found:

```
data_share -f=send -filename=my_list.txt -site=Site1 -report=rep.txt -coe
```

- To transfer ownership of a given item:

```
data_share -f=send -transfer -item_id=item123 -site=Site1
```

- To publish an assembly item and all its components using the latest revision rule to determine components:

```
data_share -f=publish -item_id=Engine100 -site=Ods1 -include_bom
```

- To publish an assembly item and all its components using the latest released revision rule to determine components:

```
data_share -f=publish -item_id=Item1 -site=Ods1 -include_bom  
-latest_released
```

- To unpublish an assembly item and all its components from multiple ODS sites using the default revision rule **latest revision**:

```
data_share -f=unpublish -item_id=Item1 -site=Ods1 -site=Ods2  
-include_bom
```

- To delete a publication record in the local database:

Note Use this only if the master object has been deleted, but the publication record still exists.

```
data_share -f=delete_pubrec -item_id=ObsoleteItem1
```

- To list the authorized ODS sites:

```
data_share -f=list_ods
```

- To check availability of the authorized ODS sites:

```
data_share -f=check_ods
```

- To get publication information about a list of objects in a folder:

```
data_share -f=list_pub_info -folder=myFolder
```

- To send an item to a specific remote user and group:

```
data_share -f=send -item_id=xyz -site=Site1 -owning_user=joe  
-owning_group=engg
```

- When publishing thousands of items and you get errors after publishing several hundreds or even thousands of items, reduce the batch size:

```
data_share -f=publish -item_id=A* -site=Site1 -batch_size=200
```

- To publish an engineering change object and all its associated change objects:

```
data_share -f=publish -item_id=CR0001 -site=Ods1 -include_bom  
0-include_bc -include_supercedures
```

- To register an item ID with the central item ID registry:

```
data_share -f=register -item_id=myItem
```

- To find duplicate item IDs at another site:

```
data_share -f=find_duplicates -item_id=00* -site=Site1
```

- To list all objects that are checked out by remote users:

```
data_share -f=list_remote_co
```

- To list all objects that are checked out by user **justin** at **Site2**:

```
data_share -f=list_remote_co -user=justin -site=Site2
```

- To cancel check out of all objects by user **joseph** at **Site2**:

```
data_share -f=cancel_remote_co -user=joseph -site=Site2
```

- To list all replica objects that are checked out by local group **engg** from remote site **Site1**:

```
data_share -f=list_replica_co -site=Site1
```

- To cancel all replica objects that are checked out by user **davis** from **Site1**:

```
data_share -f=cancel_replica_co -user=davis -site=Site1
```

- To cancel remote checkout on a given item:

```
data_share -f=cancel_remote_co -item_id=item123
```

- To cancel remote checkouts on the datasets listed in the **dataset.lst** file:

```
data_share -f=cancel_remote_co -filename=dataset.lst -class=Dataset
```

- To cancel replica checkouts on the datasets in uniquely named folder:

```
data_share -f=cancel_replica_co -folder=unique_folder_xyz
```

- To exclude all variant options during a send operation:

```
data_share -f=send -item_id=CR0002 -site=remotel  
-exclude_variant_options
```

- To batch send all variant options during a send operation:

```
data_share -f=send -item_id=CR0002 -site=remotel  
-batch_variant_options -batch_size=10000
```

- To batch send one or more classes of objects during a send operation:

```
data_share -f=send -item_id=CR0002 -site=remotel  
-batch_objects=class1,class2 -batch_size=10000
```

- To batch send one or more classes of objects given in a text file during a send operation:

```
data_share -f=send -item_id=CR0002 -site=remote1
           -batch_file=my_list.txt -batch_size=10000
```

- To import an item from a remote site (**Site2**):

```
data_share -f=remote_import -site=Site2 -item_id=xyz
```

- To import the objects of an assembly using a file (**UIDs_list.txt**) containing a list of UIDs:

```
data_share -f=ri -filename=UIDs_list.txt -classoffile=Tagstring
```

- To import an item from its owning site that is published to an ODS site (**Site3**):

```
data_share -f=ri -ods_site=Site3 -item_id=xyz
```

- To initiate a checkpoint transaction at three specified sites:

```
data_share -f=send -checkpoint -item_id=item123
           -site=Site2 -site=Site3 -site=Site4
```

- To export a role that does not have any group members from the **TopGrp1** group:

```
data_share -class=group -name=TopGrp1 -f=send
           -all_groupmembers -site=Site2
```

- To return status for a transaction ID of **AhEZaOnRAAMfD** and no **-site** argument is specified:

```
data_share -f=status -trid=AhEZaOnRAAMfD
```

The output is similar to the following:

```
Site1: sending export data to all target sites (06-Nov-2007.14:31:28)
Site2: transmitting data (06-Nov-2007.14:35:31)
Site3: importing data (06-Nov-2007.14:34:29)
Site4: error 100107 - site not currently available (06-Nov-2007.14:32:26)
Site5: transaction complete (06-Nov-2007.14:40:10)
```

The time stamp represents the last time the status was updated.

- To return status at a receiving site and no **-site** argument is specified:

```
Site3: importing batch 5 out of 50 (06-Nov-2007.14:34:29)
```

At a receiving site, only the status of the local site is obtained; the status of other sites involved in a transaction are not available.

- To restart a given transaction for a given site:

```
data_share -f=send -transaction_id=AhEZaOnRAAMfD
           -restart -site=Site3
```

- To update export records at site **Site4** using a transaction ID of **AhEZaOnRAAMfD**:

```
data_share -f=commit_ixr -trid=AhEZaOnRAAMfD -site=Site4
```

- To clean up records with a transaction ID of **AhEZaOnRAAMfD**:

```
data_share -f=cleanup_transaction -trid=AhEZaOnRAAMfD
```

- To list checkpoint transactions:

```
data_share -f=list_transactions
```


The output is similar to the following:

```
Transactions initiated by local site:
    AhEZaOnRAAMfD - active
    BxyzZaOnRAAXYZ - inactive
Transactions initiated by remote sites:
    ZaOnRAAAYXCDA - Site3
```

- To send the **casCD001** collaborative design object to the **site2** site:

```
data_share -f=send -4gd_id=casCD001 -class=Cpd0CollaborativeDesign -site=site2
```

- To use the **data_share** utility to transfer architecture breakdown structures between sites, you must execute the following four steps sequentially:

1. Push the design assembly.

```
data_share -f=send -item_id=<Design Assembly ITEM_ID> -site<REMOTE_SITE_ID>
-exclude=IMAN_reference -exclude=IMAN_based_on -exclude=IMAN_snapshot
-exclude=IMAN_3D_snap_shot -exclude=IMAN_external_object_link
-exclude=TC_Generic_Architecture -bo=VariantExpression, MEAppearancePathNode
-bs=1000 -ldv
```

2. Push the **LOUHOLDER** object and synchronize the BOM view revision.

```
data_share -f=send -item_id=<ITEM_ID> -site=<REMOTE_SITE> -include_bom
-exclude=IMAN_reference -exclude=IMAN_based_on -exclude=IMAN_snapshot
-bvo -bo=PSOccurrence -bs=20 -bvrsync -report=<IMAN_TMP_DIR>/rpt
```

3. Push the architecture breakdown structure excluding the NVEs.

```
data_share -f=send -item_id=<ITEM_ID> -rev=001 -site=<REMOTE_SITE>
-exclude=IMAN_3D_snap_shot -exclude=IMAN_reference -exclude=IMAN_external_object_link
-exclude=IMAN_based_on -evo -bo=MEAppearancePathNode -bs=2000 -ldv
```

4. Push the NVEs.

```
data_share -f=send -item_id=<ITEM_ID> -rev=001 -site=<REMOTE_SITE>
-exclude=IMAN_3D_snap_shot -exclude=IMAN_external_object_link -exclude=IMAN_based_on
-bo=MEAppearancePathNode -bs=2000 -ldv
```

- To use the **data_share** utility to transfer architecture breakdown structures between sites, you must execute the following four steps sequentially:

1. Push the design assembly.

```
data_share -f=send -item_id=<Design Assembly ITEM_ID> -site<REMOTE_SITE_ID>
-exclude=IMAN_reference -exclude=IMAN_based_on -exclude=IMAN_snapshot
-exclude=IMAN_3D_snap_shot -exclude=IMAN_external_object_link
-exclude=TC_Generic_Architecture -bo=VariantExpression, MEAppearancePathNode
-bs=1000 -ldv
```

2. Push the **LOUHOLDER** object and synchronize the BOM view revision.

```
data_share -f=send -item_id=<ITEM_ID> -site=<REMOTE_SITE> -include_bom
-exclude=IMAN_reference -exclude=IMAN_based_on -exclude=IMAN_snapshot
-bvo -bo=PSOccurrence -bs=20 -bvrsync -report=<IMAN_TMP_DIR>/rpt
```

3. Push the architecture breakdown structure excluding the NVEs.

```
data_share -f=send -item_id=<ITEM_ID> -rev=001 -site=<REMOTE_SITE>
-exclude=IMAN_3D_snap_shot -exclude=IMAN_reference -exclude=IMAN_external_object_link
-exclude=IMAN_based_on -evo -bo=MEAppearancePathNode -bs=2000 -ldv
```

4. Push the NVEs.

```
data_share -f=send -item_id=<ITEM_ID> -rev=001 -site=<REMOTE_SITE>
-exclude=IMAN_3D_snap_shot -exclude=IMAN_external_object_link -exclude=IMAN_based_on
-bo=MEAppearancePathNode -bs=2000 -ldv
```

- To send a list of 4GD objects specified in a text file and output a report:

```
data_share -f=send -filename=my_4GD_list.txt -cof=Cpd0DesignElement
-site=Site1 -low_level -report=rep.txt
```

- To send a list of 4GD objects specified in a text file and output a report:

```
data_share -f=send -filename=my_4GD_list.txt -cof=Cpd0DesignElement
-site=Site1 -low_level -report=rep.txt
```

- To send a list of 4GD objects specified in a text file and output a report:

```
data_share -f=send -filename=my_4GD_list.txt -cof=Cpd0DesignElement
-site=Site1 -low_level -report=rep.txt
```

- To send a specific 4GD partition object and output a report:

```
data_share -f=send class=Cpd0DesignElement
-key=ptn0partition_id=CD001_id,ptn0partition_scheme_type=SchemeFunctional_CD001,
partitionptn0source_object=partition_source_CD001,mdl0model_object=model_CD001,
mdl0revision_id=model__id=001/A site=Site1 -low_level -report=rep.txt
```

- To clean up fast sync transactions prior to specific last process date, list the available transactions to get the last process dates:

```
data_sync -low_level -lt
```

Clean up the transactions:

```
data_sync -low_level -ct -blpd=2012-12-18:20:10:00
```

- To use the **TC_cms_relation_optset_map** preference to include or exclude relations:

1. Add the relation and option set to the **TC_cms_relation_optset_map** preference, for example:

```
IMAN_rendering, opt_rel_rendering
```

2. In the PLM XML/TC XML Export Import Administration application, expand the **TransferOptionSet**, click **MultiSiteOptSet**, and add the **opt_rel_rendering** option with the default value set to **false**.
3. Expand **ClosureRule**, click **MultiSiteDefaultCR**, and add the following clause:

```
CLASS:WorkspaceObject:CLASS:Dataset:RelationP2S:IMAN_rendering:
SKIP:$opt_rel_rendering==false;
```

Note

This clause states, from any **WorkspaceObject**, find the dataset using **IMAN_rendering** relation, and when the relation is **opt_rel_rendering**, skip the dataset during export. It means the default is to always exclude the **IMAN_rendering** relation for an exported object.

4. To include the **IMAN_rendering** in the export using the **data_share** utility, type:

```
data_share -include=IMAN_rendering -low_level
```

data_sync

Synchronizes copies of objects at remote sites with the latest version of the object master. It also updates publication records when republishing objects. In **verify** mode, the utility checks the existence of exported objects at the remote sites; if a copy no longer exists at the remote site, the corresponding import export record is deleted from the owning site.

The behavior of this utility is controlled by the [TC_force_remote_sites_exclude_files](#) preference. If this preference is set to **true**, the replica files stored in the remote site file server cache (FSC), otherwise the replica files are store in the remote system volume.

The **data_sync** utility uses import export records (IXRs) and publication audit records (PARs), which are attached to the master copy of an object, to determine whether or not to synchronize a copy or the publication record in the ODS. These records contain information on when the object was last sent to a particular site or last published to an ODS. It then compares these dates with the object's last-modified date and decides whether or not to synchronize the object. Thus, only those objects that were modified since the last successful run of the utility are updated.

When updating multiple sites and not all sites operational, the **data_sync** utility updates the sites that are available but remembers, using the IXRs and the PARs, which ones were unavailable so they can be updated next time.

Once the utility determines which objects and sites to synchronize, it uses the basic Multi-Site Collaboration mechanisms (export, import, IDSM, and ODS) to accomplish its task. For this reason, Siemens PLM Software recommends that the **data_sync** utility be run in batch mode during off hours so that it does not compete for computing and network resources during business hours.

The utility also supports TC XML transfers for 4th Generation Design (4GD) data. The 4GD relation data mapping is controlled by the [TC_cms_relation_optset_map](#) preference. You use this preference when you want to control the relations that are included or excluded when a 4GD object is replicated.

For more information about controlling relations using this preference, see the [Preferences and Environment Variables Reference](#).

The **data_sync** utility supports part family templates and members. It also supports organization classes, specifically, **Role**, **User**, **Group**, and **Person** classes.

Siemens PLM Software recommends the following practices when using the **data_sync** utility. The term *one at a time* means one command line invocation. This implies that your script for running **data_sync** consists of several lines invoking the **data_sync** utility.

- Synchronize one site at a time and use the default revision selector of **-same_as_last_export**. This allows you to use the Smart Sync capability which synchronizes only the revisions and attachments that the remote user specified when replicating an item.
- Synchronize one class at a time starting with the largest unit, which is **Item**, and down to the smallest units such as **Dataset** and **Form**.
- Always use the **-since** switch with the **-class** switch. This results in improved memory efficiency because replicated objects that have not been modified for

some time are excluded from the initial search for objects to be synchronized. Ideally, the date given to the **-since** switch should be the exact date and time of the last successful run of **data_sync**. However, if you are not sure about the date and time, use a date and time that you know is prior to the last successful run.

- When dealing with thousands of objects, **data_sync** tends to slow down as it loads more and more objects in memory. It handles this problem by cascading its work over several sub-processes. When the original process reaches its batch size, it starts another process and then terminates itself; the sub-process continues where its parent process left off. When it reaches its own batch size, it creates another sub-process, and so on. The optimum batch size varies for each installation depending primarily on the memory (both main memory and virtual memory), so you must determine the optimum batch size for your installation.

One tool that can help you do this is the use of the **-log** switch that records all significant events in the **data_sync** log file, the file with the **.log** extension. By analyzing the log file, you can detect at what point **data_sync** begins to slow down so you can then adjust the batch size accordingly. Note that the use of the **-log** switch itself can affect the overall efficiency of **data_sync** so you should turn off the switch once you have determined your optimum batch size.

- The synchronization process can put a heavy load on the network and the systems so **data_sync** should be scheduled during non-busy hours such as nights and weekends. Typically, you should run the synchronization script run as a cron job to be started at night.
- It is not necessary to have a separate verify run before synchronization because **data_sync** always performs a verification before synchronization. View a separate verification run as a cleanup procedure and run it only when the network and the systems are not busy, such as on weekends.
- Do not use the **-disable_modified_only** switch unless there is a known problem with the default **modified-only** mode.
- If you typically share whole assemblies with a site, it is best to use the **-filename** switch to synchronize specific assemblies and use the **-include_bom** switch to synchronize any modified components. Note that you may have to use the **-force** switch in the event the item itself was not modified but you want to synchronize modified components.

SYNTAX

```
data_sync [-u=user-name {-p=password | -pf=password-file} -g=group]
{-class=class-name [-filename=file-name | -itemKeyFile=file-name] |
-item_id=template | -key=keyAttr1=keyVal1
[keyAttr2=keyVal2...,keyAttrN=keyValN]} [-OnlyVIS]
{-site=site-name -sync | -republish | -verify}
[-f=sync | republish | verify] [-status] [-commit_ixr]
[-cleanup_transaction [-transaction_id=transaction-id |
-before_last_process_date=date] |
list_transactions] [-pull] [-update]
[-replacement_site=site-replacing-extinct-site] [-stubs_only] [-sync_file_stubs]
[-force] [-report[=file-name]] [-exclude_files] [-modified_only]
[-exclude=relation-type1 -exclude=relation-type2...]
[-include=relation-type1 -include=relation-type2...]
[-include_bom] [-classoffile=class-name] [-revision-selector]
```

```

[-latest_ds_version] [-assert_extinct_ods] [-assert_extinct_site]
[-exclude_folder_contents] [-since=YYYY-MM-DD:HH:NN]
[-batch_size=number-of-objects-per-batch]
[-deferred_batch_size=batch-size-for-deferred-objects]
[-batch_objects=list-of-deferred-classes]
[-batch_file=file-name-listing-deferred-classes]
[-verbose] [-log]
[-checkpoint [-compress_ind_files={S | I | N} ] ]
[-transaction_id] [-restart]
[-low_level [-optionset=optionset-name] [-de_incl_rlz_bom]
[-workset_incl_rlz_de] [-4gd_id=objec-id -class=4gd-class-name] ]
[-h] [-bp]

```

ARGUMENTS

Note Entries in parentheses are accepted abbreviations for arguments.

-u

Specifies the user ID.

This is generally **infodba** or another user with administration privileges. If this argument is used without a value, the operating system user name is used.

Note If Security Services single sign-on (SSO) is enabled for your server, the **-u** and **-p** arguments are authenticated externally through SSO rather than being authenticated against the Teamcenter database. If you do not supply these arguments, the utility attempts to join an existing SSO session. If no session is found, you are prompted to enter a user ID and password.

-p

Specifies the password.

If used without a value, the system assumes a null value. This argument is mutually exclusive with the **-pf** argument.

If this argument is not used, the system assumes the *user-ID* value to be the password.

-pf

Specifies the password file. If used without a value, the system assumes a null value. If this argument is not used, the system assumes the *user-ID* value to be the password.

For more information about managing password files, see [Manage password files](#).

This argument is mutually exclusive with the **-p** argument.

-g

Specifies the group associated with the user.

If used without a value, the user's default group is assumed.

Caution For HTTP enabled sites, remote site operations log on using the default group for the user supplied with the **-u** argument. Any value supplied with the **-g** argument is ignored.

-pull

Specifies synchronization starts in **pull** mode, that is, from a replica site.

-class

Specifies the class of objects to be searched to determine what objects need synchronization. This does not mean that all objects of the given class are synchronized; only those that were modified since the last time they were exported to the given site(s) are synchronized. See restrictions [1](#) and [2](#).

-filename (fn)

Specifies the name of the input file containing IDs or names of objects to update. See restriction [1](#).

-itemKeyFile

Specifies the name of the input file containing the list of item key strings of the items you want to update. The following listing shows sample content of a file for updating a list of items:

```
item_id=export_001
item_id=M2Item1_001,object_name=M2Item_name1,object_type=M2Item1
```

If the item key file has 4GD object key strings, the corresponding 4GD class must be supplied using the **-classoffile(cof)=** argument. The following listing shows sample content of a file for updating 4GD data:

```
4gd_id=DE_Export_001
4gd_id=DE00001_ID,object_name=DE00001_Name,object_desc=DE00001_Desc
```

The **4g_id** entry maps to the corresponding unique ID of the 4GD class, for example:

```
Class=Cpd0DesignElement, 4gd_id=cpd0_design_element_id
```

-item_id (item)

Specifies the ID or template of items to update. See restriction [1](#).

-key

Specifies the item keys of the items to update, the template of the item keys, or the 4GD object key. It is mutually exclusive with the **-item_id** argument. Use the following format:

```
[keyAttr1=keyVal1] [,keyAttr2=keyVal2]...[,keyAttrN=keyValN]
```

To find the key of an object, use the **get_key_string** utility.

For more information, see the *Business Modeler IDE Guide*.

-OnlyVIS

Specifies synchronization of only visualization datasets attached directly or indirectly to a replicated item revision with status. See restriction [8](#).

Note If you use the **-OnlyVIS** argument the **-batch_size** value defaults to **500**.

-f

Specifies the function to be performed; define one of the following functions:

sync

Initiates the update process. See restrictions [2](#) and [4](#).

republish (repub)

Republishes objects that have been modified since last published. See restriction [2](#).

verify (veri)

Note Siemens PLM Software recommends that you always use the **-item_id=*** argument with the **verify** function. If you use the **-class=item** argument with the **verify** function, it processes only the items that have been modified after their last export.

When used with the **-update** argument, deletes the IXRs of objects for which replicas do not exist at the remote sites.

When used without the **-update** argument, generates a report. See restriction [2](#).

The report returns the following verification verdict codes:

0	Object does not exist.
1	Object exists as a master copy.
2	Object exists as a replica.
3	Object was replaced by a POM stub.

status (stat)

Displays the status of a given transaction ID.

If the **-site** argument is given, only status of the given sites is displayed. If no **-site** argument is given, the status returned depends on whether the command is given at the site that initiated the checkpointed transaction or the site is the receiving end of the transaction. If the command is given at the initiating site and no **-site** argument is given, the status of the local site and all target sites is returned.

commit_ixr (cmi)

Updates the export records at the owning site once the data is known to have been successfully imported at a target site.

Under normal conditions, the update of the export records are performed automatically by each subprocess that succeeds in completing the send operation to its assigned site. Use this function only if either of the following conditions occur:

- The failure occurs at the importing site, and the user performs the restart using the [item_import](#) utility.
- The failure occurs after the data is successfully imported by a target site, but a failure occurs just before or during the updating of the export records.

You must use at least one **-site** argument to identify the site or sites for which export records are to be updated.

You must execute this function at a node or host that has direct access to the transfer area where the export was performed (if at initiating site) or where the data was transmitted to by the owning site (if at receiving site). You must also have **read** access to the operating system directory where the export data is placed within the transfer area. If these conditions are not met, an error message is displayed to **stdout** and the utility returns a nonzero value.

cleanup_transaction (ct)

Removes transient data generated during a checkpoint transaction or fast sync transactions if use with the **-low_level** argument.

For checkpoint transactions, this transient data consists of the export data and supporting directories and files used to manage the transaction. You must execute this function at a node or host that has direct access to the transfer area where the export was performed (if at initiating site) or where the data was transmitted to by the owning site (if at receiving site). You must also have **delete** access to the operating system directory where the export data is placed within the transfer area. If these conditions are not met, an error message is displayed to **stdout** and the utility returns a nonzero value.

For fast sync transactions, you must specify either the **-transaction_id** or **-before_last_process_date** argument.

-before_last_process_date (blpd)

Specifies the date used to determine which fast sync transactions to clean up. The date must be supplied in the following format:

YYYY-MM-DD:HH:NN:SS

YYYY represents the four-digit year value. *MM* represents the two-digit month value. *DD* represents the two-digit day of the month. *HH* represents a two-digit hour value from 0 to 23. *NN* represents a two-digit minute value from 0 to 59 and *SS* represents a two-digit second value from 0 to 59. The *DD:HH:NN:SS* are optional. If they are not specified, the utility uses 12:00 AM of the specified date.

Valid only with the **-cleanup_transaction** argument.

list_transactions (lt)

Lists all uncleaned checkpoint transactions. An uncleaned transaction is one in which its transient data has not been deleted from the transfer area using the **cleanup_transaction** function.

- Active transactions can only be detected at the site that initiated it. The receiving end of a transaction is not able to tell if a transaction is active or not.
- The list of inactive transactions initiated by the local site and the list of transactions initiated by remote sites are based only on the contents of the transfer area of the node where this command is executed.
- The list of active transactions initiated by the local site is always complete because it is based on data stored in the local database.

You must execute this function at a node or host that has direct access to the transfer area where the export was performed (if at initiating site) or where the data was transmitted to by the owning site (if at receiving site). You must also have *read* access to the operating system directory where the export data is placed within the transfer area. If these conditions are not met, an appropriate error message is displayed to **stdout** and the utility returns a nonzero value.

-assert_extinct_ods (aeo)

Deletes all publication audit record (PAR) objects from the local database for an ODS that no longer exists. This removes any record about objects previously published to the ODS and makes it possible to delete the published objects at a later time. Only the **-site** and **-login** switches are required. This is valid only with the **-f=verify** argument. See restriction 5.

-assert_extinct_site (aes)

Deletes all import export record (IXR) objects from the local database for a site that no longer exists. This removes any record of objects previously exported to the site and makes it possible to delete the exported objects at a later time. Only the **-site** and **-login** arguments are required. This is valid only with the **-f=verify** argument. See restriction 5.

-replacement_site (rs)

Specifies the name of the site that replaces the site to be extinct. This argument is valid only with the **-assert_extinct_site** argument. All objects owned by the extinct site are redirected to the replacement site.

-stubs_only

Specifies that stubs are processed only when the **-replacement_site** argument is specified. This argument is valid only with the **-assert_extinct_site** argument.

-sync_file_stubs

Processes dataset files excluded from export and updates remote stubs if needed. This argument is valid only when the **-class=ImanFile** argument is specified.

-update (upd)

Performs a database update. Must be given in order for the **-f=sync**, **-f=republish**, or **-f=verify** to occur; otherwise, it only does a dry run and generates a report. See restriction 4.

-report

Generates a synchronization report. If a file name is not supplied, the report is displayed in a shell.

-site

Specifies the Teamcenter site to update. This argument can be used multiple times in the command line to synchronize with multiple name-identified sites.

-exclude_files (exf)

Excludes dataset files. See restriction 6.

-exclude

Excludes the specified relation type. This argument may be given multiple times and must use the database name (not the display name) of the relation type. See restriction 6.

-include

Includes the specified relation type. This argument may be given multiple times and must use the database name (not the display name) of the relation type. Use this argument to force the inclusion of a relation type that may have been excluded during the last export.

-exclude_folder_contents (efc)

Excludes the contents of a folder. Intended for use with NX part families where family members are stored in a folder that is related to the item.

-include_bom (bom)

Synchronizes all components of an assembly. This synchronization includes any newly added components to the existing assembly. See restriction 6.

-disable_modified_only (dmo)

Disables the default behavior of synchronizing subobjects inside an item only if they were modified since the last time the item was exported.

Normally, this argument is not used. See restriction 6.

For more information, see *Multi-Site Collaboration Guide*.

- revision-selector

Valid only if both the **-f=sync** and **-update** arguments are specified. Choose one of the following revision selectors:

-all_revisions

Synchronizes all revisions.

-latest_revision

Synchronizes only the latest revision, regardless of the release status. This is the default if no revision selector is specified and more than one site is to be synchronized. If synchronizing only one site, the default selector is **same_as_last_export**.

-latest_working

Synchronizes only the latest working (unreleased) revision.

-latest_released

Synchronizes only the latest released revision with any release status.

-latest_working_or_any

Synchronizes the latest working revision; if no working revision, synchronizes the latest released revision of any release status.

-release_status = *release-status-type*

Synchronizes only the latest released revision with the specified release status type.

-all_released_revs

Synchronizes all revisions with a release status including in-process item revisions.

-same_as_last_export

Synchronizes using the options used the last time the item was exported. This is the default if no revision selector is specified and only one site is being synchronized. If synchronizing multiple sites, the default selector is **latest_revision**.

Note

If the item was not exported in Teamcenter Engineering 7.0 (that is, the item was last exported or synchronized prior to 7.0), the latest revision used is the default.

-include_pfmembers

Identifies the related part family members to be exported when handling part family templates.

-include_pftemplates

Identifies the related part family template to be exported when handling part family members.

-pf_bom_treatment

Identifies the part family objects associated with the assemblies to be exported. The argument must be used in conjunction with the **-include_bom** argument. Valid arguments are:

-members

Includes part family member components present in the assembly.

-templates

Includes part family template rather than part family member components.

-all

Includes both the part family member components and templates.

-none

Includes neither the part family member components nor the templates.

-latest_ds_version (ldv)

Synchronizes only the latest version of datasets. See restriction 6.

-force

Synchronizes objects regardless of whether they were modified since the last time they were exported. See restriction 3.

-since

Synchronizes only those objects modified since the specified date and time, which must be specified in *YYYY-MM-DD:HH:NN* format, where *YYYY* is the year; *MM* is the month number from 1 to 12; *DD* is the day from 1 to 31; *HH* is the hour from 0 to 23, and *NN* is the minute from 0 to 59. *HH* and *NN* are optional and default to zero, which indicates 12 a.m. of the given date. This is valid only with the **-class** argument.

-verbose

Displays maximum amount of information when the utility is run in verbose mode. Typically, nonverbose utility sessions only display error messages. Do not abbreviate this argument to **-v**.

-log

Places detailed information in the **data_sync.log** file. The information includes the start and ending time for each step performed by the **data_sync** utility. Use this argument to analyze the performance of the utility.

-bp

Displays best practices information.

-checkpoint (cp)

Initiates a checkpoint transaction, that is, a transaction that can be restarted at the point of failing. This argument is valid only when both **-f=sync** and **-update** are specified. If specified without the **-update** argument, this argument is ignored.

Valid only with **-f=sync**.

If a noncheckpoint operation is initiated for multiple target sites and some target sites are not currently available based on a preliminary availability check, Teamcenter sends a message to **stdout** to notify the user about unavailable sites, removes unavailable sites from the target site list, and then performs the operation for the available sites.

compress_ind_files (cif)

Specifies compression mode to use to compress files in the export directory during a checkpoint transaction. If not specified, creates a single large ZIP file. This argument is valid only with the **-checkpoint** argument. Valid values are:

- **S**

Creates a single large ZIP file.

- **I**

Creates a ZIP file for each individual file, resulting in multiple ZIP files.

- **N**

No files are compressed.

-transaction_id (trid)

Specifies a 14-character transaction ID for a given checkpoint-related operation or fast sync transaction.

-4gd_id

Specifies a 4GD object identifier or 4GD object pattern. The **-class=class-name** argument must be specified with the **-4gd_id** argument.

The utility maps the **-4gd_id** argument to the corresponding unique ID of the 4GD class, for example:

```
Class=Cpd0DesignElement, 4gd_id=cpd0design_element_id
```

A 4GD partition object and 4GD subset definition objects do not have a unique 4GD class ID. Therefore, using **-4gd_id** for partition objects or subset definition objects may result in the update of multiple objects

To export unique partition object use multifield key attributes supplied in the **-key** argument, see the **-key** argument description.

-optionset

Specifies the name of the transfer option set when sending objects using low-level TC XML functionality (**-low_level** argument). Values of the options listed in the option set govern the object export. The option set must exist at the exporting site. If you do not specify, this argument, the utility uses the **MultiSiteOptSet** option set.

-de_incl_rlz_bom

Sends the source objects of a design element (Type:**Cpd0DesignElement**). You must specify the **-low_level** argument and specify a 4GD object using the **-4gd_id**, **-key**, or **-itemKeyFile** argument.

-workset_incl_relz_de

Sends the source objects of a design element (**Cpd0DesignElement**) in a Workset (**Cpd0Workset**). You must specify the **-low_level** argument and specify a 4GD object using the **-4gd_id**, **-key**, or **-itemKeyFile** argument.

-restart (rs)

Restarts a given transaction at the point of failure.

Valid only with the **-f=send** function.

-batch_objects (bo)

Specifies a list of comma-separated deferred classes. The list must not contain spaces.

-batch_file (bof)

Specifies the file name of a text file containing a list of deferred classes. Each class name is contained on a separate line.

-batch_size (bs)

Specifies the number of objects per batch. A new process is created for each batch. All workspace objects (not just items) that are synchronized are considered part of a batch. The default batch size is 2000. The maximum value you can specify is 99999. If you enter a value greater than 99999, the utility sets the value of **-batch_size** to the default.

-deferred_batch_size (dbs)

Specifies the number of objects per batch; a new process is created per batch. The default value is 2000. This value must be a positive integer. Use this argument to process thousands of objects to avoid memory and disk shortage problems.

The following classes are supported for deferred objects:

- **Dataset**
- **Folder**
- **Form**
- **ImanRelation**
- **MEAppearancePathNode**
- **NamedVariantExpression**
- **PSOccurrence**
- **VariantExpression**
- **VariantExpressionBlock**

-h

Displays help for this utility.

ENVIRONMENT

As specified in the [Manually configuring your environment for Teamcenter utilities](#).

FILES

As specified in the [Log files produced by Teamcenter](#).

RESTRICTIONS

1. One of the following arguments must be supplied: **-class**, **-filename**, or **-item_id**.
2. One of the following arguments must be supplied: **-f=sync**, **-republish**, or **-f=verify**.
3. The **-force** argument can only be used along with the **-filename** or the **-item_id** argument. It does not function when used in combination with the **-f=verify** argument.
4. Unless the **-update** argument is given, the **data_sync** utility generates only reports.
5. The **-assert_extinct_site** and **-assert_extinct_ods** options can only be used with the **-f=verify** argument.
6. The **-exclude_files**, **-exclude=**, **-include_bom**, **-disable_modified_only** and **-latest_ds_version** options can be used if both the **-f=sync** and **-update** arguments are supplied.
7. The **-classoffile** argument currently supports only the **Item**, **ItemRevision**, **Dataset**, **Form**, **Folder**, **Role**, **User**, **Group**, and **Person** classes.

8. The **-include** and **-update** arguments must be supplied with the **-OnlyVIS** switch.
9. To use this utility, you must be a user with system administration privileges or be granted authorization by a user with system administration privileges.

EXAMPLES

Note Required logon information is omitted from the following examples.

1. To generate a report of items that must be synchronized for a given site:

```
data_sync -class=Item -site=Site1 -f=sync
```

The report is output to **stdout**. No synchronization is performed.

2. To synchronize all items copied to a site and output a report to a file:

```
data_sync -class=Item -site=Site1 -f=sync -update -report=report.lst
```

Note The default revision selector, **-same_as_last_export**, is used.

3. To synchronize the latest released revisions of items:

```
data_sync -class=Item -site=Site1 -f=sync -update -latest_released
```

4. To synchronize all forms and datasets:

```
data_sync -class=Form -class=Dataset -site=Site1 -f=sync -update
```

5. To republish all previously published items to the **Mfg_ODS** ODS :

```
data_sync -class=Item -site=Mfg_ODS, -f=republish -update
```

6. To check if datasets copied to the **Design_Center** site still exist and delete the **IXR** from the master if a copy is no longer there:

```
data_sync -class=Dataset -site=Design_Center -f=verify -update
```

7. To force synchronization of a list of items specified in a text file copied to a site and output the report to a file:

```
data_sync -filename="/myhome/itemlist.txt" -classoffile=Item  
-site=Site1 -f=sync -update -force -report=report.lst
```

8. To force synchronization of a single item or items that match a template:

```
data_sync -item_id=Eng* -site=Site1 -f=sync -update -force  
-report=rep.lst
```

9. To destroy all the export records to a known extinct site:

```
data_sync -u=infodba -p=infodba -site=XSite -f=verify  
-update -assert_extinct_site
```

10. To destroy all the publication records to a known extinct ODS site:

```
data_sync -u=infodba -p=infodba -site=XSite -f=verify  
-assert_extinct_ods
```

11. To destroy export records, BVRs, and attachments of specific deleted replica item revisions:

```
data_sync -site=S1 -f=verify -update -fn=mylist -cof=ItemRevision
```

The **mylist** file has item revision names in the following format: **item123/A**

12. To destroy export records of specific deleted replica datasets:

```
data_sync -site=S1 -f=verify -update -filename=mylist
-classoffile=Dataset
```

The **mylist** file has dataset names in the following format: **dataset123**

13. To start synchronization in **pull** mode:

```
data_sync -pull -class=Item -site=S1 -update -report=report.lst
```

14. To force synchronize a list of items specified in a text file in **pull** mode:

```
data_sync -pull -filename="/myhome/itemlist.txt" -force
-update -report=report.lst
```

15. To generate a report of which items must be synchronized in **pull** mode:

```
data_sync -pull -filename="/myhome/itemlist.txt" -site=S1
-f=sync -report=report.lst
```

16. To synchronize visualization datasets that are under replicated Item Revision having status:

```
data_sync -OnlyVIS -since=2005-01-01:01:01 -site=Site1 -f=sync -update
-report=report.lst -include=IMAN_Rendering -include=IMAN_specification
```

17. To check if items copied to the **Design_Center** site still exist and delete the IXR from the master if a copy is no longer there, enter the following command on a single line:

```
data_sync -item_id=* -site=Design_Center -f=verify -update
```

18. To delete the IXRs of objects whose replicas do not exist at the remote sites, enter the following command on a single line:

```
data_sync -item_id=* -site=Site1 -f=verify -update -report=rep.lst
```

19. To generate a report of the IXRs, enter the following command on a single line:

```
data_sync -item_id=* -site=Site1 -f=verify -report=rep.lst
```

Both this and the previous example generate reports listing all objects including those that are no longer at the remote site.

20. To synchronize all items copied to a site and output a report to a file with newly added components to existing assembly:

```
data_sync -class=Item -site=Site1 -f=sync -update
-include_bom -report=report.rpt
```

21. To synchronize any particular item transferred to a replica site and output a report to a file with newly added components to existing assembly:

```
data_sync -u=infodba -p=infodba -g=dba -item_id=Item1
-site=Site1 -f=sync -update -include_bom -report=report.rpt
```

22. To synchronize all **imanfile** objects copied to a site and output a report to a file:

```
data_sync -class=imanfile -site=Site1 -f=sync -update -report=report.lst
```


23. To force synchronization of **imanfile** objects for all datasets specified in a text file copied to a site and output the report to a file:

```
data_sync -filename=/myhome/datasetlist-for-imanfiles.txt
          -classoffile=imanfile -site=Site1 -f=sync -update -report=report.lst
```

24. To synchronize files and initiate a checkpoint for three sites:

```
data_sync -f=sync -checkpoint -item_id=item123
          -site=Site2 -site=Site3 -site=Site4 -update
```

25. To force synchronization of files and initiate a checkpoint for three sites:

```
data_sync -f=sync -update -checkpoint -item_id=item123
          -site=Site2 -site=Site3 -site=Site4
```

26. To check the status of a given transaction:

```
data_sync -f=status -transaction_id=AhEZaOnRAAMfD
```

27. To restart a given transaction for a given site:

```
data_sync -f=sync -transaction_id=AhEZaOnRAAMfD
          -restart -site=Site3
```

28. To synchronize all 4th Generation Design (4GD) objects copied to a site and output report to a file:

```
data_sync -class=Cpd0DesignElement -site=Site1
          -sync -update -report=report.lst
```

29. To synchronize specific 4GD objects copied to a site and output report to a file:

```
data_sync -4gd_id=DE000001 -class=Cpd0DesignElement
          -site=Site1 -sync -update -report=report.lst
```

30. To clean up fast sync transactions prior to specific last process date, list the available transactions to get the last process dates:

```
data_sync -low_level -lt
```

Clean up the transactions:

```
data_sync -low_level -ct -blpd=2012-12-18:20:10:00
```

31. To use the **TC_cms_relation_optset_map** preference to include or exclude relations:

- a. Add the relation and option set to the **TC_cms_relation_optset_map** preference, for example:

```
IMAN_rendering, opt_rel_rendering
```

- b. In the PLM XML/TC XML Export Import Administration application, expand the **TransferOptionSet**, click **MultiSiteOptSet**, and add the **opt_rel_rendering** option with the default value set to **false**.
- c. Expand **ClosureRule**, click **MultiSiteDefaultCR**, and add the following clause:

```
CLASS:WorkspaceObject:CLASS:Dataset:RelationP2S:IMAN_rendering:
SKIP:$opt_rel_rendering==false;
```

Note This clause states, from any **WorkspaceObject**, find the dataset using **IMAN_rendering** relation, and when the relation is **opt_rel_rendering**, skip the dataset during export. It means the default is to always exclude the **IMAN_rendering** relation for an exported object.

- d. To include the **IMAN_rendering** in the synchronization using the **data_sync** utility, type:

```
data_sync -include=IMAN_rendering -low_level
```

IMPORTANT NOTES

1. When synchronizing items, all item revisions, BOM view revisions, BOM views, forms, and datasets associated with the item will also be synchronized. However, in some cases the item itself is not modified, so the last modification date is not updated and, therefore, cannot be used as the sole basis for synchronization. In most cases, it is necessary to specify all classes associated with an item to guarantee that complete synchronization is accomplished. This means that the command to run the **data_sync** utility should include several class switches, for example:

```
data_sync -class=Item -class=ItemRevision -class=PSBOMViewRevision
```

Note If your database contains a large number of replicated items (more than 10,000), you should synchronize one class at a time. When doing so, you should begin with the **Item** class, and then the **ItemRevision** class, followed by the **PSBOMViewRevision** class, and continue down the schema to dataset and forms classes.

2. The **PSBOMViewRevision** class must be specified instead of the **PSBOMView** class so that changes to the structure is synchronized.
3. When synchronizing an assembly, the **data_sync** utility does not automatically traverse the assembly tree. Rather, it synchronizes each subassembly or component individually on an as-needed basis. If you want the utility to traverse the assembly tree, use the **-include_bom** argument.
4. When synchronizing an assembly, **data_sync** transfers new components that are part of the assembly, when sending an assembly with the **-include_bom** argument set to true.
5. Because the **data_sync** utility never involves any transfer of ownership, there is no need to perform export recovery if the utility terminates prematurely.
6. When synchronizing, the utility performs an automatic verification. It checks if the object being synchronized still exists at the remote site prior to synchronizing it. If a replica no longer exists, the utility deletes the corresponding IXR.
7. The **-verbose** argument can be used to analyze the performance of the **data_sync** utility. The **-verbose** argument prints the system times at important stages during the process of synchronization.
8. Siemens PLM Software recommends that you synchronize only one site at a time rather than synchronizing multiple sites in a single run of the **data_sync** utility. This allows you to use the **-same_as_last_export** revision selector that uses the

same import/export options used to replicate the item. If you must synchronize multiple sites, create a script that loops through sites but only invokes the **data_sync** utility with only one site at a time.

USING FOLDERS WITH THE DATA_SYNC UTILITY

Folders can be used with the **data_sync** utility, as shown below:

```
data_sync -filename=/tmp/folderlist -classoffile=Folder...
```

If the content of the folder has changed since the last export, if references have been added or removed, the **data_sync** utility updates the remote copy to reflect the current state of the folder.

If no references have been added or removed from a folder since the last export, it is not considered to have been modified. Therefore, if the objects referenced in the folder have changed and need to be updated at the remote site using the **-classoffile=Folder** argument, use the **-force** argument.

GENERATING REPORTS

This example shows how to generate a report called **data_sync.rpt** against the Detroit site:

Enter the following command on a single line:

```
data_sync -class=Item -verify -report=data_sync.rpt -site="Detroit"
```

The results are as follows:

```
Object Date Last Modified Site Date Last Exported Type (Class)
-----
DS_0401_02A 1997-04-03 15:13:50 Detroit 1997-04-03 12:47:45 Text (Dataset)
DS_0401_02A;1 1997-04-03 15:13:40 Detroit 1997-04-03 12:47:48 Text (Dataset)
DS_0401_02A;2 1997-04-03 15:13:43 Detroit 1997-04-03 12:47:52 Text (Dataset)
0320_01/A 1997-03-24 15:34:44 Detroit 1997-03-24 15:33:57 Text (Dataset)
0320_01/A;1 1997-03-24 15:34:33 Detroit 1997-03-24 15:34:00 Text (Dataset)
0320_01/A;2 1997-03-24 15:34:38 Detroit 1997-03-24 15:34:04 Text (Dataset)
0320_01/A;3 1997-03-24 15:34:42 Detroit 1997-03-24 15:34:06 Text (Dataset)
sueD0324-4;1 1997-03-24 22:04:44 Detroit 1997-03-24 21:57:28 Text (Dataset)
sueD0324-4;2 1997-03-24 22:04:48 Detroit 1997-03-24 21:57:32 Text (Dataset)
```

ERROR CODES

Error code 100228 indicates that a Multi-Site Collaboration file transfer operation has failed. The most likely causes are a network connection failure or an abort (crash) of the IDSM process at the remote site. For the former, retry the operation. For the latter, examine the IDSM system log files at the remote site.

diff_xml

Uses the **bomwriter**-generated output files with the **grdvua_on** option specified at two days, compares the PLM XML files, and generates a difference XML file. This difference XML file contains all of the changes performed on the assembly structure. This XML file is then used to update the audit log file dataset.

SYNTAX

diff_xml **-u**= *user-name* {**-p**=*password* | **-pf**=*password-file*} [**-g**=*group*]
-item=*item-id* **-rev**=*revision-id* **-key**=*key-id* [**-h**]

ARGUMENTS**-u**

Specifies the user ID.

This is generally **infodba** or another user with administration privileges. If this argument is used without a value, the operating system user name is used.

Note

If Security Services single sign-on (SSO) is enabled for your server, the **-u** and **-p** arguments are authenticated externally through SSO rather than being authenticated against the Teamcenter database. If you do not supply these arguments, the utility attempts to join an existing SSO session. If no session is found, you are prompted to enter a user ID and password.

-p

Specifies the password.

If used without a value, the system assumes a null value. This argument is mutually exclusive with the **-pf** argument.

If this argument is not used, the system assumes the *user-ID* value to be the password.

-pf

Specifies the password file. If used without a value, the system assumes a null value. If this argument is not used, the system assumes the *user-ID* value to be the password.

For more information about managing password files, see [Manage password files](#).

This argument is mutually exclusive with the **-p** argument.

-g

Specifies the group associated with the user.

If used without a value, the user's default group is assumed.

-item

Specifies the item ID of the top node of the assembly structure.

-rev

Specifies the revision ID of the top node of the assembly structure.

-key

Specifies the key of the object. The **-key** argument can be used instead of the **-item** argument.

To find the key of an object, use the **get_key_string** utility.

For more information, see the *Business Modeler IDE Guide*.

-h

Displays help for this utility.

ENVIRONMENT

As specified in *Manually configuring your environment for Teamcenter utilities*.

FILES

As specified in *Log files produced by Teamcenter*.

RESTRICTIONS

None.

EXAMPLES

Open the Teamcenter menu shell with the database connection variables set and then execute the following command:

```
diff_xml -u=infodba -p=infodba -g=dba -item=000125 -rev=001
```

distributed_execute

Executes the **item_report** utility, both locally and remotely, and generates reports.

You can specify any command line parameters required for the **item_report** utility on the **distributed_execute** command line, and those arguments are passed to the **item_report** utility.

Note This utility does not support individual item ID input. You must use the **-itemidsfile=file** argument.

Siemens PLM Software recommends you first perform the **-distributed_func=traverse_items** step. This accumulates all traversed items from all specified sites and collects them in an output file (BOM traversal). You can use this file as input argument in subsequent steps, for example, report.

This utility does not collect logs at remote sites and return them to the local machine.

Siemens PLM Software also recommends you test this utility with emphasis on:

- Verifying the utility performs the same way locally and remotely.
- Receiving required report files and test miscellaneous combinations of command line parameters; any additional parameters specified are passed to the calling program.

SYNTAX

```
distributed_execute [-u=user-id {-p=password | -pf=password-file} -g=group]  
-distributed_func=function -itemidsfile=datafile  
-distributed_sites=site1,site2,site3 -outfile=file-name  
-delimiter=delimiter-character [-h]
```

ARGUMENTS**-u**

Specifies the user ID.

This is generally **infodba** or another user with administration privileges. If this argument is used without a value, the operating system user name is used.

Note If Security Services single sign-on (SSO) is enabled for your server, the **-u** and **-p** arguments are authenticated externally through SSO rather than being authenticated against the Teamcenter database. If you do not supply these arguments, the utility attempts to join an existing SSO session. If no session is found, you are prompted to enter a user ID and password.

-p

Specifies the password.

If used without a value, the system assumes a null value. This argument is mutually exclusive with the **-pf** argument.

If this argument is not used, the system assumes the *user-ID* value to be the password.

-pf

Specifies the password file. If used without a value, the system assumes a null value. If this argument is not used, the system assumes the *user-ID* value to be the password.

For more information about managing password files, see [Manage password files](#).

This argument is mutually exclusive with the **-p** argument.

-g

Specifies the group associated with the user.

If used without a value, the user's default group is assumed.

-distributed_func

Specifies one of the following functions:

- | | |
|-----------------------|--|
| traverse_items | Traverse BOM on all specified sites and produce union of all traversed IDs, argument maps to the item_report utility argument -traverseditemfile . |
| report | Generates a distributed IDSM-based report based on list of items input from an ID file, argument maps to item_report utility, generate the reports, and merger reports. |

-itemidsfile

Specifies a data file containing comma-separated values (CSVs) or carriage return/line feed separated item IDs. This argument is required.

-outfile

Specifies the output file. This argument is required.

-distributed_sites

Specifies a list of sites, both local and remote, on which this command is executed. This argument is required.

-delimiter

Specifies a delimiter character for the output file. The default value is the vertical bar (|). Ensure the delimiter in the site-based file and merge file input match.

-h

Displays help for this utility.

RESTRICTIONS

None.

EXAMPLES

- To execute **item_report** to generate a list of traversed objects for three sites (user, password, and group arguments are not shown in the example):

```
distributed_execute -distributed_sites=Site1,Site2,Site3
                   -outfile=trav.out -distributed_func=traverse_items -itemidsfile=item_id.txt
```

- To execute **item_report** to generate report files and merge file (user, password, and group arguments are not shown in the example):

```
distributed_execute -distributed_sites=Site1,Site2,Site3
                   -outfile=merge.out -distributed_func=report -itemidsfile=item_id.txt
```

dsa_util

Distributes system administration data, such as users and groups, from one site to another. When adding a new site, this allows you to enter the site information of all sites in the network so the new site can exchange data with them.

Caution Do not use this utility to share organization objects between sites if the same organization objects are shared through a global organization.

Note This utility should be used only for the initial migration of system objects. Siemens PLM Software recommends that you do *not* use this utility to maintain system objects.

Propagates Teamcenter administration data among multiple sites and allows administrators to:

- Manage system data from a central site.
- Support non-networked sites.
- Create reports of the results of a distribution operation.

SYNTAX

```
dsa_util [-u=user-id {-p=password | -pf=password-file} -g=group]
-f={distribute | export | import | list_classes | list_sites | check_sites |
set_logging_level -level={FATAL | ERROR | WARNING | INFO | DEBUG |
TRACE | OFF}}
[-site=remote-site1-name -site=remote-site2-name...]
[-class=class1-name -class=class2-name...]
[-filename=file-path-name]
[-report=report-file-name]
[-email=email-address]
[-attr1-name=attr1-value -attr2-name=attr2-value]
[-attr-name-listfile=file-path-name]
[-h={topics | topic-name} | -h [-f=function-name | -class=class-name]]
```

ARGUMENTS

Note Entries in parentheses are accepted abbreviations for arguments.

-u

Specifies the user ID.

This is generally **infodba** or another user with administration privileges. If this argument is used without a value, the operating system user name is used.

Note If Security Services single sign-on (SSO) is enabled for your server, the **-u** and **-p** arguments are authenticated externally through SSO rather than being authenticated against the Teamcenter database. If you do not supply these arguments, the utility attempts to join an existing SSO session. If no session is found, you are prompted to enter a user ID and password.

-p

Specifies the password.

If used without a value, the system assumes a null value. This argument is mutually exclusive with the **-pf** argument.

If this argument is not used, the system assumes the *user-ID* value to be the password.

-pf

Specifies the password file. If used without a value, the system assumes a null value. If this argument is not used, the system assumes the *user-ID* value to be the password.

For more information about managing password files, see [Manage password files](#).

This argument is mutually exclusive with the **-p** argument.

-g

Specifies the group associated with the user.

If used without a value, the user's default group is assumed.

Caution

For HTTP enabled sites, remote site operations log on using the default group for the user supplied with the **-u** argument. Any value supplied with the **-g** argument is ignored.

-f

Identifies the function to perform. Must be one of the following functions:

distribute (dist)	Sends system objects to the given sites.
export (exp)	Outputs system object information to the text file identified by the -filename argument. The output file can be edited and used with the distribute function in conjunction with the -filename argument. Equivalent to exporting system objects.
import (imp)	Imports system object information from the text file identified by the -filename argument and updates system objects in the local database. Equivalent to importing system objects.
list_classes (lc)	Lists the name of all classes supported by this utility.
list_sites (ls)	Lists all the sites that are defined in the local database.
check_sites (cs)	Checks the availability of all sites defined in the local database. A site is considered available for Distributed System Administration purposes if its IDSM server is ready.
set_logging_level (sll)	Sets the logging level at the remote site indicated by the -site argument to the level indicated by the -level argument. See restrictions 5 and 6 .

-level

Specifies the logging level for a given remote site in a remote procedure call (RPC)-based Multi-Site environment. Valid values are:

- **FATAL**

Logs only severe errors that cause premature program termination.

- **ERROR**

Logs other run-time errors or unexpected conditions.

- **WARNING**

Logs run-time situations that are undesirable or unexpected, such as use of deprecated APIs.

- **INFO**

Logs informational messages that highlight the progress of the application at a coarse-grained level, such as startup and shutdown events.

- **DEBUG**

Logs detailed information on the flow through the system. This information is useful for debugging an application.

- **TRACE**

Logs the most detailed information on system events and operation.

- **OFF**

Turns off logging.

See restrictions [5](#) and [6](#).

Set the logging level at the local site in the **logger.properties** file in the *TC_DATA* directory.

-site

Identifies the remote sites to which system objects are distributed. May be given multiple times in the same command line to distribute to multiple sites.

-class (cl)

Identifies the system class or classes to be processed. This argument can be given multiple times in the command line but only if the entire class is to be processed. No attribute switches are allowed when multiple classes are given.

Note All class names are case insensitive.

-filename (fn)

Specifies the path name of a text file to be used as input or output of system object information. If only the file name is given, the file is assumed to be in the user's current directory.

To prevent the system from appending the **.plmxml** file extension to the specified file name, Siemens PLM Software recommends that you specify a file name using the **.xml** file extension.

-report (rep)

Specifies the path name of a text file to which the local report is written.

-email (em)

Indicates the e-mail address to which the remote report is sent. It can be a single address or multiple addresses separated by a semicolon (;).

-attr-name=attr-value

Specifies the attribute name and value pair identifying a specific instance of the given system class. This argument can be given multiple times in the command line if necessary to locate a specific instance of a given class.

-attr-name_listfile

Specifies the path name of the text file containing the IDs or names of instances of the given class. Use to process multiple instances of a given class.

-h=topics

Displays a list of topics for which detailed help information is available.

-h=topic-name

Displays help information for a specific topic.

-h -f=function-name

Displays detailed help information for the given function.

-h -class=class-name

Displays class-specific help information for the given class. The class name must be one of the classes listed by the **list_classes** function. If the class name is set to **ALL_CLASSES**, displays help information for all supported classes.

-h

Displays help information on basic usage.

RETURN VALUES

Return value 0
upon success

Return value >1
upon failure

RESTRICTIONS

1. To use this utility, you must be a user with system administration privileges or be granted authorization by a user with system administration privileges.
2. Do not use this utility to share organization objects between sites if the same organization objects are shared through a global organization.
3. When exporting the user object, this utility does not export the license level of the user. The license level of the user is set to the lowest available license level at the importing site. System administrators at the importing site must manually set the license level and/or the license bundle of each user.
4. The **dsa_util** utility does not recognize externally managed users, groups, roles, persons, and group members with a **datasource** attribute value greater than **0** and convert them to remotely managed (for example, managed by an LDAP external directory at a remote site). Because the **dsa_util** utility is the only way user constructs can be converted to remotely managed, no user construct objects in the Organization user interface appear as remotely managed.
5. The **set_logging_level** function and **-level** argument are only for RPC-based Multi-Site environments. Use the JMX console to set the logging levels for the loggers in an HTTP (four-tier) Multi-Site environment.

6. To use this argument, you must add the source site (site where the utility is run) to the **IDSMDsa_sites_permitted_to_push_admin_data** preference value at the remote site.

EXAMPLES

To set the Multi-Site logging level to **DEBUG** at a remote site **203456177**, enter the following command on a single line:

```
dsa_util -u=infodba -p=infodba -f= set_logging_level -site=203456177 level=DEBUG
```

ensure_site_consistency

Allows users to perform corrective actions if the site ownership transaction is interrupted due to a system or network crash or a user-initiated process termination (such as the Windows Task Manager). In cases where legitimate error conditions are encountered (such as lack of transfer privilege or duplicate item IDs), there is no requirement to perform any corrective action; Teamcenter restores the data to consistent states under most non-crash conditions.

Note This utility should be run only at the exporting site; never run it at the importing site. The flag that marks an object as requiring this utility is always at the exporting site.

SYNTAX

```
ensure_site_consistency [-u=user-id {-p=password | -pf=password-file} -g=group]
-f=recovery | report | list_all_rec | clean_all_rec
{ [-item_id=item-id] | [-key={keyAttr1=keyVal1 [,keyAttr2=keyVal2],...[,keyAttrN=keyValN]}
[-class=class-name] ] | [-folder=folder-name] | [-filename=file-name] |
[-itemKeyFile=file-name] | [-search] | [-4gd_id=object-id -class=4gd-class-name] }
[-report=file-name]
[-mode={sst | gms} ]
[-h]
```

ARGUMENTS

-u

Specifies the user ID.

This is generally **infodba** or another user with administration privileges. If this argument is used without a value, the operating system user name is used.

Note If Security Services single sign-on (SSO) is enabled for your server, the **-u** and **-p** arguments are authenticated externally through SSO rather than being authenticated against the Teamcenter database. If you do not supply these arguments, the utility attempts to join an existing SSO session. If no session is found, you are prompted to enter a user ID and password.

-p

Specifies the password.

If used without a value, the system assumes a null value.

If this argument is not used, the system assumes the *user-ID* value to be the password.

This argument is mutually exclusive with the **-pf** argument.

-pf

Specifies the password file. If used without a value, the system assumes a null value. If this argument is not used, the system assumes the *user-ID* value to be the password.

For more information about managing password files, see [Manage password files](#).

This argument is mutually exclusive with the **-p** argument.

-g

Specifies the group associated with the user.

If used without a value, the user's default group is assumed.

-f

Specifies the function to perform. Valid values are:

report

Generates a list of objects that require recovery. The list is output to a text file identified by the value of the **report** argument. By default, the report contains the Global Services transfers (GMS) report followed by the Synchronous Site Transfer (SST) report. If the **-mode** argument is supplied with this argument, the utility generates a report on the mode specified.

The **clean_all_rec** value can be used with this value.

recovery

Performs recovery operations such as reclaiming site ownership, releasing transfer locks, and removing unwanted export records.

list_all_rec

Lists all inconsistent local/replica **IXR**, **ITXR**, and **PAR** records that exist for local/replica objects on a local site. This value must be used with the **report** value.

clean_all_rec

Deletes all inconsistent local/replica **IXR**, **ITXR**, and **PAR** records specified in the input that exist for local/replica objects at a local site. This argument deletes workspace objects only. It does not delete **VariantExpression**, **AbsOccData**, or **MEApprPathNode** objects.

Note

Inconsistent local/replica **IXR**, **ITXR**, and **PAR** records occur in the following situations:

- The object is local and the referencing auxiliary objects (**IXR**, **ITXR**, and **PAR**) are replicas.
- The object is replica and the referencing auxiliary objects (**IXR**, **ITXR**, and **PAR**) are replicas.
- The object is replica and the referencing auxiliary objects (**IXR**, **ITXR**, and **PAR**) are replicas.
- At a hub site, the object is local/replica and the referencing auxiliary objects (**IXR**, **ITXR**, and **PAR**) are replicas.

-folder

Specifies a folder that contains items on which to perform corrective action.

The use of a folder is intended for Workspace objects that do not have unique IDs, for example, datasets and forms. This is useful for failed remote checkins of multiple objects where many of the remotely checked-out objects do not have unique IDs, for example, datasets, forms, BVRs, and so forth.

-filename

Specifies a file name that contains a list of items on which to perform corrective action. The file should only contain item IDs. This argument is mutually exclusive with the **-folder**, **-item_id**, and **-search** arguments.

-class (cl)

Specifies the Teamcenter class of the object specified by the **-name** or **-4gd_id** argument. This argument is valid only with the **-name** or, when the **-low_level** argument is specified, with the **-4gd_id** argument. The default class is **Item**.

For organization objects, this argument accepts **Role**, **User**, **Group**, and **Person** classes.

-key

Specifies the keys of the objects on which to perform corrective action. Use the following format:

```
[keyAttr1=keyVal1] [,keyAttr2=keyVal2]...[,keyAttrN=keyValN]
```

To find the key of an object, use the [get_key_string](#) utility.

For more information, see the [Business Modeler IDE Guide](#).

-itemKeyFile

Specifies the name of the file containing the keys of the objects on which to perform corrective action.

-search

Finds all the objects that are flagged as requiring corrective action.

This argument is mutually exclusive with the **-folder**, **-filename**, and **-item_id** arguments.

- When used with the **report** function, the utility generates a report on all the objects that are found as requiring corrective actions.
- When used with the **recovery** function, the utility performs corrective actions on the objects that are found as requiring corrective actions.

-4gd_id

Specifies a 4GD object identifier or 4GD object pattern. The **-class=class-name** argument must be specified with the **-4gd_id** argument.

The utility maps the **-4gd_id** argument to the corresponding unique ID of the 4GD class, for example:

```
Class=Cpd0DesignElement, 4gd_id=cpd0design_element_id
```

A 4GD partition object and 4GD subset definition objects do not have a unique 4GD class ID. Therefore, using **-4gd_id** for partition objects or subset definition objects may result corrective action on multiple objects.

To export unique partition objects, use multifield key attributes supplied in the **-key** argument. See [Examples](#).

-item_id

Specifies the item ID.

- When used with the **report** function, the utility generates a report on the item specified by *item-ID*.

- When used with the **recovery** function, the utility performs corrective action on the item specified by *item-ID* only if the specified item is flagged as requiring corrective actions.

-report

Specifies the output file path for generating the report. Use this argument with either the **report** function or the **recovery** function.

- When used with the **report** function, the report lists the objects that require corrective action.
- When used with the **recovery** function, the report lists the objects where corrective action was taken.

-mode

Specifies the recovery method type of transfer failures that you want to recover or the type of report to generate when used with the **-report** argument. If you do not specify this argument, the utility uses standard Multi-Site Collaboration transfer failures or generates both a Synchronous Site Transfer (SST) and Global Services (GMS) report when used with the **-report** argument. You can specify one of the following valid values:

- **sst**
When used with the **recovery** function, **sst** recovers SST transaction failures.
When used with the **report** argument, the report lists the SST transaction objects that require corrective action.
- **gms**
When used with the **recovery** function, **gms** recovers GMS transaction failures.
When used with the **report** argument, the report lists the GMS transaction objects that require corrective action.

-h

Displays help for this utility.

ENVIRONMENT

As specified in the [Manually configuring your environment for Teamcenter utilities](#).

FILES

As specified in the [Log files produced by Teamcenter](#).

RESTRICTIONS

None.

EXAMPLES

- Generate a report on all the objects that are flagged as requiring corrective actions:

```
ensure_site_consistency -u=infodba -p=infodba -g=dba
-f=report -search -report=recovery_candidates.txt
```

- Generate a report on the item specified by *item_ID*:

```
ensure_site_consistency -u=infodba -p=infodba -g=dba
-f=report -item_id=000301 -report=recovery_item.txt
```


- Perform corrective actions on all the objects that are flagged as requiring corrective actions:

```
ensure_site_consistency -u=infodba -p=infodba -g=dba  
-f=recovery -search -report=recovery_fixup.txt
```

- Perform corrective actions on the item specified by *item_ID*:

```
ensure_site_consistency -u=infodba -p=infodba -g=dba  
-f=recovery -item_id=000301 -report=recovery_fixup.txt
```

- Perform corrective actions on a list of item IDs:

```
ensure_site_consistency -u=infodba -p=infodba -g=dba  
-f=recovery -filename=item_id_list.txt
```

The **item_id_list.txt** file should contain a list of item IDs, one item ID per line.

- Perform corrective actions on all objects under a given uniquely named folder:

```
ensure_site_consistency -u=infodba -p=infodba -g=dba  
-f=recovery -folder=RecoveryFolderFor26June2007
```

export_recovery

Recovers and restores exported objects to your database under certain conditions. Occasionally, when you export an object and transfer ownership the object may not be successfully imported at the destination site. This places the object in an undefined state where no one has ownership. The preferred method of correcting this situation is to have the destination site complete the import/export transaction by importing the object into the database from the importing site's **TC_transfer_area** (using interactive object import).

However, if this is not possible, the **export_recovery** utility is used to restore the object to the exporting database from the exporting site's **TC_transfer_area** using the **min** or **full** mode (effectively canceling the export/transfer ownership transaction). If no data is available at either site, recovery can be attempted by running the automode at the exporting site that was the last known owning site.

Use the **export_recovery** utility when an export with transfer of site ownership fails, resulting in objects within an item having inconsistent site ownership. The mode of recovery to use depends on whether there is a valid export directory. The directory must include the **objects.meta** file.

Siemens PLM Software recommends the following order for attempting export recovery procedures; you should try the succeeding procedure only if you cannot perform the previous one or if the previous one fails to restore site ownership:

- If a valid export directory exists (most likely in the **TC_transfer_area** of the exporting or importing site), use either **full** or **min** mode while specifying the valid export directory with the **-dir=** switch. If you attempt to recover at the exporting site, use **min** mode; if you attempt to recover at the importing site, use **full** mode.
- If a valid export directory does not exist, you must attempt recovery from a valid database copy that may be a replica or one with inconsistent site ownership. Use **export_recovery** in **auto** mode. Specify the **-include_bom** switch if appropriate. Specify **-exclude** and/or **-include** switches, if desired.
- If the **auto** mode fails to restore site ownership, perform the manual export recovery procedure:
 1. Define the **TC_EXPORT_COPY=TRUE** environment variable.
 2. Run **item_export** as the **infodba** user to transfer site ownership to any site.
 3. Run **export_recovery** in **min** mode specifying the directory output in step 2 as the **-dir=** parameter.
 4. If successful, delete the export directory from step 2.

SYNTAX

```
export_recovery [-u=user-id {-p=password | -pf=password-file} -g=group]
  -mode={ full | min | auto | find }
  [-item_id=item-id-to-restore]
  | [-key=[keyAttr1=keyVal1] [,keyAttr2=keyVal2]...[,keyAttrN=keyValN]]
  | [-folder=folder-name] | [-filename=file-name]
  | [-itemKeyFile=file-name] | [-dir=directory]
```

[-report=report-file] **[-remote_site=last-transfer-site]** **[-include_bom]**
[-real_owning_site=desired-owning-site]
[-exclude=relation-type1 -exclude=relation-type2 ...]
[-include=relation-type3 -include=relation-type4 ...]
[-ignore_am_rules] **[-update_lmd]** **[-bp]** **[-h]**

ARGUMENTS

-u

Specifies the user ID.

This is generally **infodba** or another user with administration privileges. If this argument is used without a value, the operating system user name is used.

Note

If Security Services single sign-on (SSO) is enabled for your server, the **-u** and **-p** arguments are authenticated externally through SSO rather than being authenticated against the Teamcenter database. If you do not supply these arguments, the utility attempts to join an existing SSO session. If no session is found, you are prompted to enter a user ID and password.

-p

Specifies the password.

If used without a value, the system assumes a null value. This argument is mutually exclusive with the **-pf** argument.

If this argument is not used, the system assumes the *user-ID* value to be the password.

-pf

Specifies the password file. If used without a value, the system assumes a null value. If this argument is not used, the system assumes the *user-ID* value to be the password.

For more information about managing password files, see [Manage password files](#).

This argument is mutually exclusive with the **-p** argument.

-g

Specifies the group associated with the user.

If used without a value, the user's default group is assumed.

-mode

Specifies the basic mode in which the utility operates. The value of this argument can be one of the following:

full

Restores objects from the export metafile, imports them in to your database and restores ownership to your site.

min

Restores ownership to your site without reimporting data from the metafile. Valid only if the metafile was generated with transfer of ownership.

auto

Restores ownership on the specified item without reimporting. You must specify the **-itemid** argument and either the **-remote_site** or **-real_owning_site** arguments when using this mode.

find

Searches for items with inconsistent site ownership and generates a report.

-dir

Defines the path of the directory containing the exported metafile and the data files. Required only with the **-mode=full** and **-mode=min** arguments.

-item_id

Specifies the ID of the item to process. Wildcards are allowed.

-folder

Defines the name of the Teamcenter folder containing the list of items to process.

-filename

Defines the full path of the file that contains the list of items to process.

-key

Specifies the keys of the items to process. Use the following format:

```
[keyAttr1=keyVal1] [,keyAttr2=keyVal2]...[,keyAttrN=keyValN]
```

To find the key of an object, use the **get_key_string** utility.

For more information, see the *Business Modeler IDE Guide*.

-itemKeyFile

Specifies the name of the input file containing the keys to process. The file format is:

```
-key = [keyAttr1=keyVal1][keyAttr2=keyVal2]...
-key = [keyAttr1=keyVal1][keyAttr2=keyVal2]...
-key = [keyAttr1=keyVal1][keyAttr2='keyVal2']...
```

-remote_site

Defines the last site for which a transfer of ownership was attempted. This argument is valid only with the **auto** mode.

-report

Specifies the full path of the report file. Valid only with **find** mode.

-real_owning_site

Changes the owning site of specified objects to the site designated. Valid only with the **-mode=auto** argument.

-include_bom

Includes assembly components, if any exist.

-exclude

Excludes the specified relation type and may be given multiple times. The database name (not display name) of the relation type must be used.

-include

Includes the specified relation type and may be given multiple times. The database name (not the display name) of the relation type must be used. Use this argument to force the inclusion of a relation type that is not specified by your **TC_relation_required_on_export** preference.

-ignore_am_rules

Ignores AM rules for recovery purposes.

-update_lmd

Updates the last modified user and date. Valid only with the **-mode=auto** argument.

-bp

Displays best practices information.

-h

Displays help for this utility.

ENVIRONMENT

As specified in the [Manually configuring your environment for Teamcenter utilities](#).

FILES

As specified in the [Log files produced by Teamcenter](#).

RESTRICTIONS

- To use this utility, you must be a user with system administration privileges or be granted authorization by a user with system administration privileges.
- At least one primary mode must be specified.
- Not more than one primary mode can be specified.
- For the **-mode=auto** and **-mode=find** options, exactly one object selection filter (**-itemid**, **-filename**, or **-folder**) must be specified.

EXAMPLES

In each of the following examples, the **-u=user-id** **-p=password** and **-g=group** arguments are assumed:

- To restore ownership on an item with the ID **MyCorruptItem**:

```
export_recovery -mode=auto -item_id=MyCorruptItem
               -remote_site=Manufacturing
```

- To restore ownership on objects contained in an export metafile without reimporting:

```
export_recovery -mode=min -dir=metafile_dir
```

- To reimport objects from the metafile and restore site ownership:

```
export_recovery -mode=full -dir=metafile_dir
```

- To generate a report of ownership inconsistencies:

```
export_recovery -mode=find -filename=suspect_itemlist.dat
               -report=report.dat
```

- To make an item (**xyz**) in the local site a replica that is owned by another site (**Site2**):

```
export_recovery -mode=auto -item_id=xyz -real_owning_site=Site2
```

- To restore ownership of an entire assembly:

```
export_recovery -mode=auto -item_id=Assy1 -remote_site=Site2
               -include_bom
```

idsminetd

Serves as the Integrated Distributed Services Manager (IDSM) launching program on UNIX systems. Located in the **\$TC_ROOT/bin** directory, it is run at system startup and services all inbound requests for a new IDSM.

For more information on IDSM, see the [Multi-Site Collaboration Guide](#).

SYNTAX

idsminetd [-u=user-id {-p=tcp-port-number | -pf=password-file} -g=group]
[-d] [-t] [-r=idsm-start-script]
[-n=RPC-program-number]

ARGUMENTS**-u**

Specifies the user ID.

This is generally **infodba** or another user with administration privileges. If this argument is used without a value, the operating system user name is used.

Note

If Security Services single sign-on (SSO) is enabled for your server, the **-u** and **-p** arguments are authenticated externally through SSO rather than being authenticated against the Teamcenter database. If you do not supply these arguments, the utility attempts to join an existing SSO session. If no session is found, you are prompted to enter a user ID and password.

-p

Specifies the port number on which the IDSM should run. Default is the system-assigned port number.

-pf

Specifies the password file. If used without a value, the system assumes a null value. If this argument is not used, the system assumes the *user-ID* value to be the password.

For more information about managing password files, see [Manage password files](#).

This argument is mutually exclusive with the **-p** argument.

-g

Specifies the group associated with the user.

If used without a value, the user's default group is assumed.

-d

Specifies debug mode for stand-alone testing. The server runs in the foreground.

-t

Enhances logging.

-r

Specifies the IDSM start script.

-n

Specifies the RPC program number the IDSM should use. The default RPC program number is used if this argument is omitted.

ENVIRONMENT

As specified in *Manually configuring your environment for Teamcenter utilities*.

FILES

As specified in *Log files produced by Teamcenter*.

RESTRICTIONS

The selected port number must be in the range 1025–65535 and must not conflict with other system services.

EXAMPLES

Under normal circumstances, this utility runs only at system startup. The following is an example of running a debug session:

```
idsminetd -d -t -p=33333 -r=/tmp/myscript
```

import_file

Imports files into the Teamcenter database according to a set of user-specified arguments. These arguments supply user identification information, dataset information, and (optionally) item information to be associated with the imported file. The arguments may be specified on the command line to import a single data file or in a file to import multiple data files (bulk import).

Depending on the arguments, each data file is copied (an **ImanFile** object is created), a dataset is created (or modified), and if specified, an item is created or modified to contain the dataset. In the absence of a specified item, the dataset is placed in the user's **Newstuff** folder.

Note The **import_file** utility does not support the creation of custom item types.

SYNTAX

```
import_file [-u=user-id {-p=password | -pf=password-file} -g=group]
-f=file-name | -i=file-name [-vb] [-log=file-name] -type=datasettype -d=dataset-name
-ref=named-reference [-de={n | e | a | r}] [-item=item-id | -itemkey=key-id]
[-itemRevUid=item-revision-uid]
[-relationType=relation-type] [-use_ds_attached_to_rev_only]
[-revision=item-rev-num] [-ie={n | y}] [-desc=string]
[-v=volume-name] [-h]
```

ARGUMENTS**-u**

Specifies the user ID.

This is generally **infodba** or another user with administration privileges. If this argument is used without a value, the operating system user name is used.

Note If Security Services single sign-on (SSO) is enabled for your server, the **-u** and **-p** arguments are authenticated externally through SSO rather than being authenticated against the Teamcenter database. If you do not supply these arguments, the utility attempts to join an existing SSO session. If no session is found, you are prompted to enter a user ID and password.

-p

Specifies the password.

If used without a value, the system assumes a null value. This argument is mutually exclusive with the **-pf** argument.

If this argument is not used, the system assumes the *user-ID* value to be the password.

-pf

Specifies the password file. If used without a value, the system assumes a null value. If this argument is not used, the system assumes the *user-ID* value to be the password.

For more information about managing password files, see [Manage password files](#).

This argument is mutually exclusive with the **-p** argument.

-g

Specifies the group associated with the user.

If used without a value, the user's default group is assumed.

-f

Imports a single file into Teamcenter. The full path must be provided if the file does not reside in the current working directory. The **-f** and **-i** arguments are mutually exclusive. See example 1.

-i

Imports multiple files into Teamcenter using a specified import file. The full path must be provided if the file does not reside in the current working directory. The **-f** and **-i** arguments are mutually exclusive. See example 2.

-vb

Runs utility in verbose mode. Displays maximum amount of information. Nonverbose sessions only display error messages.

-log

Creates a log of items and datasets created.

-type

Defines the dataset type in Teamcenter, for example, **TEXT** or **UGPART** datasets.

-d

Specifies the name of the dataset into which the file is imported.

-ref

Specifies the type of named reference associated with the file. The value specified by this argument may or may not be identical to the value specified by the **-type** argument.

For example, **TEXT** or **UGPART** type datasets have named references of **TEXT** and **UGPART**, respectively. However, **DirectModel** type datasets have a **JTPART** named reference. Each dataset type defines one or more named references to be associated with it. See restriction #2.

For more information, see the [My Teamcenter Guide](#).

-de

Indicates that a dataset exists. Used when a dataset of the same name already exists.

=n

Specifies that a new dataset be added even if one with the same name exists. If it does exist, it is added to the same item folder. If it does not exist, it is placed in the new item folder or the user's **Newstuff** folder.

=e

Specifies that the dataset should be added if it already exists and that this dataset type supports multiple instances of the same dataset.

=a

Specifies that the imported file be added as a named reference to the existing dataset. When this is done, a new dataset version that contains the additional imported named reference file is created.

=r

Specifies that a new dataset revision be created and the existing named reference be replaced with the new one. This option generates an error if the dataset has no existing named reference.

-item

Specifies the name of the item containing the dataset that references the imported file.

-itemkey

Specifies the key of the object. You can use the **-item** argument or the **-itemkey** argument.

To find the key of an object, use the [get_key_string](#) utility.

For more information, see the [Business Modeler IDE Guide](#).

-itemRevUid

Specifies the 14 character UID string of the item revision object where the file and dataset are to be attached.

-relationType

Specifies a relation to use when the **IMAN_specification** relation is not appropriate. If the **-relationType** argument is not specified, the **IMAN_specification** relation is used.

-use_ds_attached_to_rev_only

Specifies a dataset based on its name and type, but it considers datasets attached to the specified item and item revision only. The item and item revision are specified by the **-item** and **-revision** parameters, respectively.

This prevents the utility from referring to datasets with the same name and type but that are unattached or are attached to another item or revision.

The **-use_ds_attached_to_rev_only** parameter is particularly useful when used along with the **de=r** argument (that is, if you want to revise the existing dataset instead of creating a new one).

-revision

Specifies the item revision number and revision ID. See restriction #3.

-ie

Specifies behavior if the item already exists.

=n

Specifies that the dataset will not be added if the item already exists.

=y

Specifies that the dataset may be added if the item already exists. If the item exists, but the item revision does not, an item revision is created.

-desc

Specifies a user-defined text description of an item that is created by the import function. If the **import_file** utility is creating a new revision of an existing item, this is the description of the item revision.

-v

Specifies the full path of the Teamcenter volume where the imported file is placed.

ENVIRONMENT

As specified in the [Manually configuring your environment for Teamcenter utilities](#).

FILES

As specified in the [Log files produced by Teamcenter](#).

RESTRICTIONS

1. When importing a file, the file name cannot be longer than 31 characters.
2. To create a dataset in Teamcenter, the user must specify the dataset type and the named reference.
3. When importing a file as a dataset, you must specify the named reference using the **-ref** argument.
4. When importing a file into an item or item revision, you must specify the revision; otherwise, an error message displays indicating a missing revision.

EXAMPLES

- To import a single operating system file, **bike.dat**, into Teamcenter as a **UGPART** dataset named **my_bike_dataset**, enter the following on a single line:

```
$TC_ROOT/bin/import_file -user=user-id -p=password -g=group -f=bike.dat
-type=UGPART -d=my_bike_dataset -ref=UGPART
```

- To import multiple operating system files into Teamcenter, first create an input file that contains the following information:

```
-f=bike1.dat -d=my_bike1_dataset -type=UGPART -ref=UGPART
-f=bike2.dat -d=my_bike2_dataset -type=UGPART -ref=UGPART
.
-f=binkeN.dat -d=my_bikeN_dataset -type=UGPART -ref=UGPART
```

- Run the **import_file** utility using the input file from example 2, entering the following command on a single line:

```
$TC_ROOT/bin/import_file
-user=user-id -password=password -group=group -i=input-file
```

- Import the **d:\some_file.jt** file:

```
%TC_ROOT%\bin\import_file -user=user-id -p=password -group=group
-f=d:\some_file.jt -type=DirectModel -d=my_jt_file_dataset -ref=JTPART
```

- Import the **d:\WordDoc.doc** file:

```
%TC_ROOT%\bin\import_file -user=user-id -p=password -group=group
-f=d:\WordDoc.doc -type=MSWord -d=my_word_dataset -ref=word
```

- Import the **d:\ExcelFileTest.xls** file:

```
%TC_ROOT%\bin\import_file -user=user-id -p=password -group=group
-f=d:\ExcelFileTest.xls -type=MSEExcel -d=my_excel_dataset -ref=excel
```

- Import the **d:\myfile.txt** file:

```
%TC_ROOT%\bin\import_file -user=user-id -p=password -group=group
-f=d:\myfile.txt -type=Text -d=my_text_file_dataset -ref=Text
```

item_export

Exports a single item or multiple items in batch mode. It is the companion to the **item_import** utility. This utility supports part family templates and members and works with the **TC_relation_required_on_export** and **TC_relation_required_on_transfer** preferences.

SYNTAX

```
item_export [-u=user-id {-p=password | -pf=password-file} -g=group]
-dir=directory
{-item=item-id | -key=key-id [-rev=revision-selector] | -filename=input-file
| -itemKeyFile=file-name}
{-owning_site=site-name | -target_site=site-name1, site-name2, ...}
[-exclude=relation-type1 -exclude=relation-type2...]
[-reason=export-reason] [-latest_ds_version] [-include_bom]
[-batch_objects=list-of-deferred-classes]
[-batch_file=file-name-listing-deferred-classes]
[-deferred_batch_size=batch-size-for-deferred-objects]
[-preview] [-report=file-name] [-continue_on_error]
[-xfr_top_lvl_only] [-xfr_top_asm_only] [-xcl_files]
[-status=release-status] [-exclude_folder_contents]
[-classoffile=class-name] [-separator=separator-character]
[-dont_exclude_protected] [-email=email-address] [-script=script-name]
[revision-selector] [-include_bc] [-include_supercedures] [-v] [-h]
```

ARGUMENTS

Note Entries in parentheses are accepted abbreviations for arguments.

-u

Specifies the user ID.

This is generally **infodba** or another user with administration privileges. If this argument is used without a value, the operating system user name is used.

Note If Security Services single sign-on (SSO) is enabled for your server, the **-u** and **-p** arguments are authenticated externally through SSO rather than being authenticated against the Teamcenter database. If you do not supply these arguments, the utility attempts to join an existing SSO session. If no session is found, you are prompted to enter a user ID and password.

-p

Specifies the password.

If used without a value, the system assumes a null value. This argument is mutually exclusive with the **-pf** argument.

If this argument is not used, the system assumes the *user-ID* value to be the password.

-pf

Specifies the password file. If used without a value, the system assumes a null value. If this argument is not used, the system assumes the *user-ID* value to be the password.

For more information about managing password files, see [Manage password files](#).

This argument is mutually exclusive with the **-p** argument.

-g

Specifies the group associated with the user.

If used without a value, the user's default group is assumed.

-dir

Specifies the full path of the directory where the metafile and data files are stored.

-item

Specifies the ID of the item to be exported. Valid only if no input file is specified using the **-i** argument.

-key

Specifies the key of the object. You can use the **-item** argument or the **-key** argument.

To find the key of an object, use the [get_key_string](#) utility.

For more information, see the [Business Modeler IDE Guide](#).

-rev

Identifies the revision to be exported. This can be the revision ID or one of the following keywords:

```
LATEST
LATEST_WORKING
LATEST_RELEASED
LATEST_WORKING_OR_RELEASED
USE_STATUS
```

If the **USE_STATUS** keyword is given, you must specify a release status using the **-status** argument. This is valid only when you are not transferring site ownership.

If used with an input file (**-i** argument), the revision keyword is used for every item in the input file. Keywords cannot be specified using the command line; however, you can use revision selectors at the command line as discussed below.

Note The revision ID cannot be specified when using an input file.

-filename (fn)

Specifies the name of an input file that contains the list of item IDs to be exported.

The format of the text file must contain the **-item=** prefix to each item ID. For example, to export item IDs **002259**, **002260**, and **002261**, the input file contains the following entries:

```
-item=002259
-item=002260
-item=002261
```

Note This replaces the **-i** argument, which is supported for backward compatibility.

-keyFileName

Specifies the name of the input file containing the keys to be exported. The file format is:

```
-key = [keyAttr1=keyVal1][keyAttr2=keyVal2]...
-key = [keyAttr1=keyVal1][keyAttr2=keyVal2]...
-key = [keyAttr1=keyVal1][keyAttr2='keyVal2']...
```

-classoffile (cof)

Specifies the class of objects contained in the input file. If no class is specified, the default class is **Item**. Valid only with the **-filename** argument.

-separator (sep)

Specifies the character to separate the item and revision IDs in the file. The default is **/**.

-target_site (ts)

Specifies the export target site or sites. If more than one site is specified, sites must be separated by a comma and the entire string must be enclosed in quotes. Either the **-target_site** or **-owning_site** argument is required.

-owning_site (os)

Specifies the site to which ownership is transferred. Either the **-target_site** or **-owning_site** argument is required.

-exclude (exc)

Specifies the relation type to be excluded. This argument may be given multiple times, and the database name (not the display name) of the relation type must be specified. You cannot exclude the **IMAN_master_form** and **TC_ic_intent_rtype** relation types with or without ownership transfer. Also, you cannot exclude the **IMAN_RES_audit** with ownership transfer.

-exclude_folder_contents (efc)

Excludes the contents of a folder. Intended for use with NX Part families where family members are stored in a folder that is attached to the item.

-dont_exclude_protected (dxp)

Does not exclude export-protected objects. If set, any export-protected object within an item prevents the export of the entire item.

-reason (rea)

Specifies the reason for exporting to sites. Up to 240 characters.

-latest_ds_version (ldv)

Exports only the latest version of datasets; default is to export all versions. Valid only when site ownership is not being transferred.

-include_bom (bom)

Exports all components if the given item is an assembly.

-preview (pre)

Performs an export dry run and generates a report to the file specified by the **-report** argument. If the **-report** argument is not specified, the report is output to the screen.

-report (rep)

Outputs preview or completion reports to the specified file. If no report file name is specified, the report is output to the screen.

-continue_on_error (con)

Continues the export operation even if errors are detected on optional objects. Optional objects are attachments other than requirement, specification, or master form objects.

-xfr_top_lvl_only

Only transfers ownership on top-level items specified in the input file.

-xfr_top_asm_only

Transfers ownership only on the top-level assembly items, as specified in the input.

-xcl_files

Excludes export of files in datasets.

-include_pfmembers

Identifies the related part family members to be exported when handling part family templates.

-include_pftemplates

Identifies the related part family template to be exported when handling part family members.

-pf_bom_treatment

Identifies the part family objects associated with the assemblies to be exported. The argument must be used in conjunction with the **-include_bom** argument. Valid options are:

-members

Includes part family member components present in the assembly.

-templates

Includes part family template rather than part family member components.

-all

Includes both the part family member components and templates.

-none

Includes neither the part family member components nor the templates.

-status (stat)

Specifies the release status type to use for selecting the item revision to be exported.

-include_bc

Exports the change revision along with the **BOMChange** objects associated with the affected assemblies of the change revision.

-include_supercedures

Exports the change revision along with the supercedures associated with the **BOMChange** objects.

-email

Specifies the e-mail address to which the export report is sent.

The default address is stored in the Teamcenter user account.

-script

Specifies the name of the script in the **TC_BIN** directory that is executed after a successful export. If a script is already defined by the **TC_post_export_script** preference, the specified script overrides the preference entry.

For more information, see the *Preferences and Environment Variables Reference*.

-revision_selector

Determines which item revision is exported with the item. The valid selectors are as follows:

latest_revision (lt)	Exports the latest revision only, regardless of whether it is a working or released revision.
latest_working (lw)	Exports the latest working revision only.
latest_released (lr)	Exports the latest released revision only with any release status.
latest_working_or_any (lwoa)	Exports the latest working revision. If no working revision exists, it exports the latest released revision with any release status.
status (stat)	Specifies the release status to be exported.

If no revision selector is given, all revisions are exported.

Note Revision selectors should be capitalized only when used with the **-rev=** switch and should be in lower case when used as a switch.

-v

Runs utility in verbose mode to display maximum amount of information. Typically, nonverbose utility sessions only display error messages.

-batch_objects (bo)

Specifies a list of comma-separated deferred classes. If you use this argument with the **preview** argument, only nondeferred objects with the number of deferred objects appear in the report.

-batch_file (bof)

Specifies the file name of a text file containing a list of deferred classes.

-deferred_batch_size (dbs)

Specifies the number of objects per batch; a new process is created per batch. The default value is **1000**. This value must be a positive integer. Use this argument to process thousands of objects to avoid memory and disk shortage problems.

The following classes are supported for deferred objects:

- **Dataset**
- **Folder**
- **Form**
- **ImanRelation**
- **MEAppearancePathNode**

- **NamedVariantExpression**
- **PSOccurrence**
- **VariantExpression**
- **VariantExpressionBlock**

-h

Displays help for this utility.

ENVIRONMENT

As specified in the [Manually configuring your environment for Teamcenter utilities](#).

FILES

As specified in the [Log files produced by Teamcenter](#).

RESTRICTIONS

1. The **-item** argument is mutually exclusive with the **-i** and **-filename** arguments.
2. Either the **-target_site** or **-owning_site** argument must be specified and must not be a local site.
3. It is the responsibility of the user exporting objects to inform the system administrator which directories need to be exported and to which site.
4. It is the responsibility of the system administrator to set up the list of other sites which are known to the local site.
5. It is the responsibility of the system administrator to send directories of the exported objects to the receiving sites, users, volumes, and other systems.
6. Administration object types cannot be exported.

EXAMPLES

To restart a checkpoint transaction that failed during import:

```
item_export -transaction_id=AjEZaOnRAAAffD -restart
```

item_import

Imports multiple items (in batch mode) into the Teamcenter database. It is the companion to the **item_export** utility. This utility supports part family templates and members.

SYNTAX

```
item_import [-u=user-id {-p=password | -pf=password-file} -g=group]
             -dir=directory
             [-folder=folder-name] [-preview] [-report=file-name] [-filename=file-name]
             [-classoffile=class-name] [-list_metafile] [-include_pfmembers=part-family-members]
             [-include_pftemplate=part-family-templates]
             [-part_family_bom_treatment={members | templates | all | none}]
             [-script=pre-import-script] [-email=email-address]
             [-parallel=number-of-parallel-processes] [-continue_on_error] [-verbose]
             [-transaction_id=tid] [-restart] [-h]
```

ARGUMENTS

Note Entries in parentheses are accepted abbreviations for arguments.

-u

Specifies the user ID.

This is generally **infodba** or another user with administration privileges. If this argument is used without a value, the operating system user name is used.

Note If Security Services single sign-on (SSO) is enabled for your server, the **-u** and **-p** arguments are authenticated externally through SSO rather than being authenticated against the Teamcenter database. If you do not supply these arguments, the utility attempts to join an existing SSO session. If no session is found, you are prompted to enter a user ID and password.

-p

Specifies the password.

If used without a value, the system assumes a null value. This argument is mutually exclusive with the **-pf** argument.

If this argument is not used, the system assumes the *user-ID* value to be the password.

-pf

For more information about managing password files, see [Manage password files](#).

This argument is mutually exclusive with the **-p** argument.

-g

Specifies the group associated with the user.

If used without a value, the user's default group is assumed.

-dir

Specifies the path name to the directory containing the metafile and the data files to be imported.

-folder (fol)

Specifies the destination folder in which imported items are placed. If the folder does not exist or the argument is not supplied, the imported items are placed in the user's **Newstuff** folder.

-preview (pre)

Performs an import dry run and generates a dry run report to the file specified by the **-report** argument. If the **-report** argument is not specified, the report is output to the screen.

-report (rep)

Outputs a preview or completion report to the specified report file. If no report file name is specified, the report is output to the screen.

-filename (fn)

Specifies the name of the text file listing objects for selective import, one name per line.

-classoffile (cof)

Specifies the class of objects contained in the input file. If not specified, the default class is **Item**.

-list_metafile (lm)

Lists only the contents of the metafile; does not import objects.

-include_pfmembers

Identifies the related part family members to be imported when handling part family templates.

-include_pftemplate

Identifies the related part family template to be imported when handling part family members.

-part_family_bom_treatment

Identifies the part family objects associated with the assemblies to be imported. The argument must be used in conjunction with the **-include_bom** argument. Valid options are:

-members

Includes part family member components present in the assembly.

-templates

Includes part family template rather than part family member components.

-all

Includes both the part family member components and templates.

-none

Includes neither the part family member components nor the templates.

-script

Specifies the name of the script to be executed prior to import. If a script is defined by the **TC_post_export_script** preference, this argument overrides the preference setting. If specified as **NONE**, the script defined in the preference file is executed.

-email

Sends e-mail to the user at the specified e-mail address after completion. If no e-mail address is specified, the e-mail address in the user's Teamcenter user profile is used.

-parallel (par)

Specifies the number of processes to be started automatically. If this argument is not specified, the system imports the deferred objects with a sequential process.

-continue_on_error (con)

Specifies that the import operation proceeds when an error has occurred on an optional object, such as a reference or manifestation attachment.

-verbose (v)

Runs the utility in verbose mode to display the maximum amount of information. Typically, nonverbose utility sessions only display error messages.

-transaction_id (trid)

Specifies the transaction ID for a given checkpoint-related operation.

-restart (rs)

Restarts a given transaction at the point of failure.

-h

Displays help for this utility.

ENVIRONMENT

As specified in the [Manually configuring your environment for Teamcenter utilities](#).

FILES

As specified in the [Log files produced by Teamcenter](#).

RESTRICTIONS

None.

EXAMPLES

To restart a transaction at a site where a failure occurred:

```
item_import -transaction_id=transaction-id -restart
```

item_relink

Replaces the external references for a duplicate item and its corresponding replica in a Multi-Site Collaboration environment. The **item_relink** utility works in conjunction with the **item_rename** utility.

The **Bypass** option enables or disables special bypass object protections for Teamcenter administrators, allowing you to freely access any object in the database to perform maintenance. When running this utility, you must use the **Bypass** option, the user ID must be **infodba**, and the OS user account must have read-access to the NX part files to comply with the rules stated in **Bypass UG Part File Verification**.

Caution The **item_relink** utility is used only with Multi-Site Collaboration to process production data. Siemens PLM Software recommends that a full backup of your database be performed before running this utility. This allows you to restore the database if the data becomes corrupted.

- **Naming pattern**

Because the **item_relink** utility works in conjunction with the **item_rename** utility, the same naming pattern must be used in both utilities.

- **Bypass NX part file verification**

Each NX part file that is attached to a duplicate is checked against the corresponding NX part file that is attached to the replica. The **item_relink** utility compares the UID strings in the NX part files. The **ug_inspect** utility retrieves the UID strings from the part files. Therefore, it is very important to run the **item_relink** utility using the OS account that has read access to the part files. Usually, the Teamcenter user account, such as **infodba**, is used to run the utility. If UIDs are not the same, the relink process for the duplicate fails. If you are confident about the part files being reconciled, you can use the **-bypass_ugpart** command line argument to bypass this check. The **-bypass_ugpart** argument is ignored if the **item_relink** utility is run in verify mode.

- **Refile folder**

The **-refile** argument generates a refile folder used as an output folder. The refile folder contains all the assemblies or subassemblies that use the replicas.

After a duplicate is reconciled, the **item_relink** utility retrieves the items that reference the replica item revisions and adds them to the refile folder. If the refile folder does not exist in the database, the utility creates one.

If no items are added to the newly created refile folder, it is not saved in the database and no refile process is required. Otherwise, the refile folder is saved and used as an input folder during the refile process. The refile folder must reside in the **Home** folder of the **infodba** user to comply with the restriction in the **ugmanager_refile** utility.

For more information about the **ugmanager_refile** utility, see Teamcenter Integration for NX documentation in the NX online help collection.

- **Matching criteria**

To find the corresponding replica, construct the replica item ID based on the duplicate item ID and renaming pattern and then search the database for the replica.

To find the corresponding item revision, match the revision ID.

To find the corresponding BOM view, match the view type name.

To find the corresponding BOM view revision, match the view type name.

To find the corresponding secondary object, match the object name, object type, and relation type.

- **Matching results**

For each object that is attached to a duplicate item or duplicate item revision, if more than one object with the same object name, object type, and relation type are found in its corresponding replica item or replica item revision, the first occurrence is used.

If objects attached to a duplicate do not have corresponding objects found in replica, use the **-verify** switch to generate a report that lists any discrepancies. In this case, perform a detailed examination and make the necessary corrections and/or ownership change for the duplicate. If any discrepancies are detected during the relink process, the duplicate is not replaced. Instead, the duplicate is placed in the exception folder. An error message is logged on the report file for review.

- **Exception**

If unexpected Teamcenter internal errors occur or the duplicate contains objects not found in its corresponding replica, the utility stops processing the duplicate that has a problem, logs an error message to the report file, and then processes the next duplicate in the replacement folder. All duplicates that are not reconciled are placed in the **Item_ID_ConsolidationEXP** exception folder so you can further examine these duplicates.

SYNTAX

```
item_relink [-u=user-id {-p=password | -pf=password-file} -g=group]
-replace=folder-name -refile=folder-name
-update | -verify
[-prefix=prefix-removed-from-item-id | -suffix=suffix-removed-from-item-id | -f=file-name]
[-bypass_ugpart]
[-ignore_attachments]
[-relink_to_latest_rev]
-report | -report=file-name
[-h]
```

ARGUMENTS

-u

Specifies the user ID.

This is generally **infodba** or another user with administration privileges. If this argument is used without a value, the operating system user name is used.

Note If Security Services single sign-on (SSO) is enabled for your server, the **-u** and **-p** arguments are authenticated externally through SSO rather than being authenticated against the Teamcenter database. If you do not supply these arguments, the utility attempts to join an existing SSO session. If no session is found, you are prompted to enter a user ID and password.

-p

Specifies the password.

If used without a value, the system assumes a null value. This argument is mutually exclusive with the **-pf** argument.

If this argument is not used, the system assumes the *user-ID* value to be the password.

-pf

Specifies the password file. If used without a value, the system assumes a null value. If this argument is not used, the system assumes the *user-ID* value to be the password.

For more information about managing password files, see [Manage password files](#).

This argument is mutually exclusive with the **-p** argument.

-g

Specifies the group associated with the user.

If used without a value, the user's default group is assumed.

-replace

Specifies the name of the folder that holds items that are currently duplicates but should be replicas. This must be the same folder used by the **item_rename** utility.

-refile

Specifies the name of the folder that holds all the assemblies that use the replicas. This folder is the input folder to the **ugmanager_refile** utility.

For more information about the **ugmanager_refile** utility, see Teamcenter Integration for NX documentation in the NX online help collection.

-verify

Requests verification of compatibility between duplicates and replicas.

-update

Performs the link replacement.

-prefix

Specifies the prefix removed from the item ID of duplicates to form the new item IDs for replicas. This must be the same prefix used by the **item_rename** utility. See restriction 5.

-suffix

Specifies the suffix removed from the item ID of duplicates to form the new item IDs for replicas. This must be the same suffix used by the **item_rename** utility. See restriction 5.

-f

Specifies the file containing item ID cross reference records. The cross reference contains the duplicate item ID and the renamed duplicate item ID for each duplicate item ID. This data is contained in a single 80-byte line in the file. The **item_rename** utility also uses this file. See restriction 5.

-bypass_ugpart

Indicates whether NX part files are verified. If this switch is specified, no verification is performed. This switch is ignored if the **-verify** argument is specified.

-ignore_attachments

Prohibits linking of secondary objects (attachments). Use this argument when the replacement item already has links to all required attachments.

-relink_to_latest_rev

Links all revisions of a duplicate item to latest revision of the replica item. Ignores secondary objects (attachments) to avoid incorrect attachments. This argument cannot be specified for items with multiple views.

-report

Generates a report. Outputs the report to standard output if the file name is not supplied.

-h

Displays help for this utility.

ENVIRONMENT

As specified in the [Manually configuring your environment for Teamcenter utilities](#).

FILES

As specified in the [Log files produced by Teamcenter](#).

RESTRICTIONS

The following restrictions must be understood and adhered to when using the **item_relink** utility:

1. Must be run with the **Bypass** option, and the user ID must be **infodba**.
2. The OS user account must have read-access to the NX part files to comply with the rules stated in **Bypass UG Part File Verification**.
3. The **-replace** and **-refile** arguments must be supplied.
4. Either the **-rename** or **-verify** arguments must be supplied.
5. A default naming pattern is used if the **-prefix**, **-suffix**, or **-f** argument is not supplied.

EXAMPLES

The best practice is to run the **item_relink** utility with the **-verify** argument to do a comparison to find discrepancies between duplicates and replicas. If any exist, examine the discrepancies and make the necessary corrections. To ensure data integrity, Multi-Site Collaboration imposes strict rules on object replication. One of these rules is that only the master object can be modified. The replicated object must never be checked out for modification or submitted for release. Therefore, if the duplicates contain objects that have no corresponding replicas, the relink process for

these duplicates is not performed. However, if the replicated objects have increased with more attachments, the duplicates are overwritten.

- To verify the items in the replacement folder and generate a report called **relink.rpt**, enter the following command on a single line. The naming pattern must be the same as that used by the **item_rename** utility.

```
Item_relink -u=infodba -p=infodba -replace=replacement  
-refile=assm_refile -prefix=AAA -verify -report=relink.rpt
```

- After generating a replacement report, you may need to correct duplicates or change ownership. To replace the links, enter the following command on a single line:

```
Item_relink -u=infodba -p=infodba -replace=replacement  
-refile=assm_refile -prefix=AAA -update -report=relink.rpt
```

item_rename

Changes the item IDs for duplicate part numbers in a naming pattern in a Multi-Site Collaboration environment. The **item_rename** utility works in conjunction with the **item_relink** utility. The main reason for renaming duplicates is to avoid a naming conflict while bringing in copies of the master data that was previously created.

The **Bypass** option enables or disables special bypass object protections for Teamcenter administrators, allowing you to freely access any object in the database to perform maintenance. When running this utility, you must use the **Bypass** option and the user ID must be **infodba**.

- **Naming pattern**

You can use the **-prefix**, **-suffix**, or **-f** arguments to embed a renaming pattern for the duplicate data objects. If these arguments are not used, the system applies a default naming pattern. The default naming pattern adds the **DUP_** prefix to the duplicate item IDs. For example, if the duplicate item ID is **ABC123**, after the **item_rename** utility runs the duplicate item ID is **DUP_ABC123**.

The **-prefix** and **-suffix** switches enable you to add character strings to the item IDs to form new item IDs.

The **-f** switch supplies a file that contains a list of item ID cross-references. If the **-f** argument is specified, the system ignores the **-prefix** and **-suffix** switches.

- **Cross-reference file format**

The **-f** switch generates a file that contains a list of item ID cross-references, specifically the duplicate item ID and the renamed duplicate item ID. Each set of item IDs is contained in a single 80-byte line. The duplicate item ID precedes the renamed duplicate item ID. The duplicate item ID and the renamed duplicate item ID must be separated by at least one blank space, although more are allowed. Leading blanks may appear before the duplicate item ID or padding blanks may appear after the renamed duplicate item ID.

The system administrator manually creates the cross-reference file. The system administrator must know how to match the item ID replicas and the item ID duplicates.

- **Exception**

If any unexpected Teamcenter internal errors occur, the utility stops processing the duplicate that has a problem, logs an error message to the report file, and then processes the next duplicate in the replacement folder.

The **item_rename** utility is used only with Multi-Site Collaboration.

SYNTAX

```
item_rename [-u=user-id {-p=password | -pf=password-file} -g=group]
-replace=folder-name -rename | -verify [-prefix= prefix-added-to-item-id |
-suffix= suffix-added-to-item-id | -f=file-name] -report=file-name [-h]
```

ARGUMENTS**-u**

Specifies the user ID.

This is generally **infodba** or another user with administration privileges. If this argument is used without a value, the operating system user name is used.

Note If Security Services single sign-on (SSO) is enabled for your server, the **-u** and **-p** arguments are authenticated externally through SSO rather than being authenticated against the Teamcenter database. If you do not supply these arguments, the utility attempts to join an existing SSO session. If no session is found, you are prompted to enter a user ID and password.

-p

Specifies the password.

If used without a value, the system assumes a null value. This argument is mutually exclusive with the **-pf** argument.

If this argument is not used, the system assumes the *user-ID* value to be the password.

-pf

Specifies the password file. If used without a value, the system assumes a null value. If this argument is not used, the system assumes the *user-ID* value to be the password.

For more information about managing password files, see [Manage password files](#).

This argument is mutually exclusive with the **-p** argument.

-g

Specifies the group associated with the user.

If used without a value, the user's default group is assumed.

-replace

Specifies the name of the folder that holds the items that are currently duplicates but that should be replicas.

-prefix

Specifies the prefix added to the item ID of duplicates to form the new item IDs.

-suffix

Specifies the suffix added to the item ID of duplicates to form the new item IDs.

-f

Specifies the file containing item ID cross-reference records. The cross-reference is comprised of the duplicate item ID and the renamed duplicate item ID for each duplicate item ID. This data is contained in a single 80-byte line in the file. The **item_relink** utility also uses this file.

-verify

Requests verification of the existence of renamed items.

-rename

Performs the rename function.

-report

Generates a report and outputs it to standard output if the file name is not supplied.

-h

Displays help for this utility.

ENVIRONMENT

As specified in the [Manually configuring your environment for Teamcenter utilities](#).

FILES

As specified in the [Log files produced by Teamcenter](#).

RESTRICTIONS

The following restrictions must be understood and adhered to when using the **item_rename** utility:

1. Must use the **Bypass** option, and the user ID must be **infodba**.
2. The **-replace** argument must be supplied.
3. Either the **-rename** or **-verify** argument must be supplied.
4. A default naming pattern is used if either the **-prefix**, **-suffix**, or **-f** arguments are not supplied.

EXAMPLES

The best practice is to run **item_rename** with the **-verify** switch to do a quick search for any objects with the chosen naming pattern. If any exist, choose a different naming pattern for all objects.

- Enter the following command on a single line to verify the items in the replacement folder and generate a report called **rename.rpt**. Assume that the naming pattern adds the prefix **AAA** to the item ID.

```
Item_rename -u=infodba -p=infodba -replace=replacement  
-prefix=AAA -verify -report=rename.rpt
```

If any items in the database have the same item ID as the chosen naming pattern, error messages beginning with *****ERROR** are logged on the **rename.rpt** file.

- Change the naming pattern and run the **item_rename** utility again. Otherwise, use the same command line in step 1, and replace the **-verify** argument with the **-rename** argument to rename the items.

item_report

Generates detail reports of an item or multiple items at the site level. The site level reports can be merged to generate a combined status output.

Using this utility, a site can investigate item consistency and dual ownership. It also provides information about the last modified user, locked information and details about the checkout user (owning and remote checkout). Checkout information includes the checkout user, respective checkout date, and time information.

SYNTAX

```
item_report [-u=user-id {-p=password | -pf=password-file} -g=group]
[-f=report | merge] [-itemidsfile=data-file
| -itemKeyFile=file-name] | -grmtypefile=grm-file
| [-item_id=itemids
| -key=[keyAttr1=keyVal1] [,keyAttr2=keyVal2]...[,keyAttrN=keyValN]

[-mergelist=file1,file2,...] | [-mergefiles=merge-list-file-name]
[-sites_list=site1,site2,...] | [-sites_file=site-list-file-name]
[-file=file1] [-file=file2] [-file=file3]
[-remove_consistent] [-show_rco] [-include_bom]
[-includefoldercontent] [-delimiter=delimiter]
[-anchorfile=anchor-file-name] [-traverseditemfile=outname]
[-outfile=file-name]
[-skipItem]
{-start_creation_date=creation-date
-end_creation_date=creation-date |
-start_modification_date=modification-date
-end_modification_date=modification-date] |
[-start_release_date=release-date
-end_release_date=release-date] }
[-sort_by=item_id | item_name | date] [-dataset_version=latest | all]
[-out_item_revs_file=output-file] [-h]
```

ARGUMENTS

-u

Specifies the user ID.

This is generally **infodba** or another user with administration privileges. If this argument is used without a value, the operating system user name is used.

Note

If Security Services single sign-on (SSO) is enabled for your server, the **-u** and **-p** arguments are authenticated externally through SSO rather than being authenticated against the Teamcenter database. If you do not supply these arguments, the utility attempts to join an existing SSO session. If no session is found, you are prompted to enter a user ID and password.

-p

Specifies the password.

If used without a value, the system assumes a null value. This argument is mutually exclusive with the **-pf** argument.

If this argument is not used, the system assumes the *user-ID* value to be the password.

-pf

Specifies the password file. If used without a value, the system assumes a null value. If this argument is not used, the system assumes the *user-ID* value to be the password.

For more information about managing password files, see [Manage password files](#).

This argument is mutually exclusive with the **-p** argument.

-g

Specifies the group associated with the user.

If used without a value, the user's default group is assumed.

-function

Specifies one of the following functions:

report Parses object information to a report file. This is the default.

merge Parses input from a set of input files as specified by the **mergelist** argument.

-itemidsfile

Specifies a data file containing comma-separated values (CSVs) or carriage return/line feed separated item IDs.

-item_id

Specifies a list of comma-separated values of item IDs.

-key

Specifies the keys of the items on which to report. Use the following format:

```
[keyAttr1=keyVal1] [,keyAttr2=keyVal2]...[,keyAttrN=keyValN]
```

To find the key of an object, use the [get_key_string](#) utility.

For more information, see the [Business Modeler IDE Guide](#).

-itemKeyFile

Specifies the name of the file containing the keys on which to report. The file format is:

```
-key = [keyAttr1=keyVal1][keyAttr2=keyVal2]...  
-key = [keyAttr1=keyVal1][keyAttr2=keyVal2]...  
-key = [keyAttr1=keyVal1][keyAttr2='keyVal2']...
```

-grmtypefile

Specifies a data file containing comma-separated values (CSVs) or carriage return/line feed separated **grmrelations**.

-mergelist

Specifies a comma-separated list of files from individual sites in the same order as specified by the **sites_list** argument.

-mergefiles

Specifies a file containing a list of reports from individual sites in the same order as specified by the **sites_list** argument.

-sites_list

Specifies a comma-separated list of sites to be analyzed when using the **report** function. This option is ignored when using any of the date-range arguments.

-sites_file

Specifies a file containing a list of sites to be analyzed when using the **report** function. This option is ignored when using any of the date-range arguments.

-remove_consistent

Specifies that the utility does not output consistent items in the merged report.

-show_rco

Specifies that the utility displays remote checkout information. By default, this information is not displayed.

-include_bom

Specifies that the utility traverses to the end of the item revisions of **PSOccurrences**. By default, this action is not performed.

-includefoldercontent

Specifies that the utility includes the contents of folders. By default, folders are not processed.

-delimiter

Specifies the delimiter used in the output file. By default, the commercial at symbol (@) is used. Ensure the delimiter for site-based files matches the merge file input file.

-anchorfile

Specifies the output file of UIDs of revision anchors. This file is used as input to the **purge_dataset** utility.

-traverseditemfile

Specifies an output list of traversed item IDs when using the **include_bom** argument.

-outfile

Specifies an output file.

-skipItem

Specifies that items are to be skipped.

-start_creation_date

Specifies the creation *from* date. The date is entered in *dd-mmm-yyyy hh:mm:ss* format, for example, **01-Jan-2007 00:00:00**. This argument is used with the **end_creation_date** argument.

-end_creation_date

Specifies the creation *to* date. The date is entered in *dd-mmm-yyyy hh:mm:ss* format, for example, **Jan-2007 00:00:00**. This argument is used with the **start_creation_date** argument.

-start_modification_date

Specifies the modification *from* date. The date is entered in *dd-mmm-yyyy hh:mm:ss* format, for example, **Jan-2007 00:00:00**. This argument is used with the **end_modification_date** argument.

-end_modification_date

Specifies the creation *to* date. The date is entered in *dd-mmm-yyyy hh:mm:ss* format, for example, **Jan-2007 00:00:00**. This argument is used with the **start_modification_date** argument.

-start_release_date

Specifies the released *from* date. The date is entered in *dd-mmm-yyyy hh:mm:ss* format, for example, **Jan-2007 00:00:00**. This argument is used with the **end_released_date** argument.

-end_release_date

Specifies the released *to* date. The date is entered in *dd-mmm-yyyy hh:mm:ss* format, for example, **Jan-2007 00:00:00**. This argument is used with the **start_released_date** argument.

-dataset_version

Specifies whether **all** or the **latest** version of the dataset needs to be reported. If this argument is not specified, the utility uses the default value of **all**.

-sort_by

Specifies one of the following attributes by which the items are processed:

- **date**
- **item_id**
- **item_name**

If this argument is not specified, the utility uses the default value of **item_id**.

-out_item_revs_file

Specifies an output file for item revisions. You can use this file as an input to the **delete_pdm_data** utility. The following is an example of an output item revision file:

```
ABC000075/A
ABC000074/A
ABC000092/A
ABN000002/A
ABN000011/A
ABN000058/A
```

-h

Displays help for this utility.

RESTRICTIONS

None.

EXAMPLES

- To create reports for the latest dataset versions created between **01-jan-2007** and **01-jan-2008**, write the report file to **c:\temp\reports.txt** and write the reported item revisions to **c:\temp\itemrevs.txt**:

```
item_report -u=infodba -p=xxxxxx -g=dba
  -start_creation_date="01-Jan-2007 00:00:00"
  -end_creation_date="01-Jan-2008 00:00:00"
  -outfile=c:\temp\reports.txt -out_item_revs_file=c:\temp\itemrevs.txt
```

migrate_organization

Allows you to create a global organization for your Multi-Site environment. Organization objects that have been duplicated across multiple sites can be migrated to be replicas of a master organization site. You use the utility first to identify duplicate objects between two sites. After identifying objects that must be made replicas, use the utility make the identified duplicate objects replicas of the master organization site objects.

SYNTAX

```
migrate_organization [-u=user-id {-p=password | -pf=password-file} -g=group]
{-f=compare_organization -report=report-file-name | make_replica}
{ [-user_id=userid1,userid2, ... | -role_name=rolename1,rolename2, ... |
-group_name=group1,group2, ... | -person_name=person1,person2, ...] |
[-classoffile=class-name -filename=input-file] }
{-site=remote-site-name}
[-h]
```

ARGUMENTS**-u**

Specifies the user ID.

This is generally **infodba** or another user with administration privileges. If this argument is used without a value, the operating system user name is used.

Note

If Security Services single sign-on (SSO) is enabled for your server, the **-u** and **-p** arguments are authenticated externally through SSO rather than being authenticated against the Teamcenter database. If you do not supply these arguments, the utility attempts to join an existing SSO session. If no session is found, you are prompted to enter a user ID and password.

-p

Specifies the password.

If used without a value, the system assumes a null value. This argument is mutually exclusive with the **-pf** argument.

If this argument is not used, the system assumes the *user-ID* value to be the password.

-pf

Specifies the password file. If used without a value, the system assumes a null value. If this argument is not used, the system assumes the *user-ID* value to be the password.

For more information about managing password files, see [Manage password files](#).

This argument is mutually exclusive with the **-p** argument.

-g

Specifies the group associated with the user.

If used without a value, the user's default group is assumed.

-f

Specifies the name of the function. Following functions are valid:

compare_organization

Identifies the duplicate and missing organization objects at the remote site (specified by **-site** argument) and the current site. The remote site is specified using the **-site** argument and the output is written to the file specified by the **-report** argument.

make_replica

Makes the object specified by the **-user_id**, **-group_name**, **-role_name**, or **-person_name** argument replica objects at the target (remote) site. This command must be used only at the site that is intended to be the master site for organization objects. One class of object must be specified but only one class can be in a command. Alternatively, the **-classoffile** and **-file_name** arguments can be specified to use a file containing the objects to be replicated. See [restriction 4](#).

-report

Specifies the file name of the report generated by the **compare_organization** function.

-user_id

Specifies an ID of a user to change to a replica object at the remote site. You can specify multiple IDs separated by commas.

-group_name

Specifies the full name of a group to change to a replica object at the remote site. You can specify multiple group names separated by commas.

-role_name

Specifies the name of a role to change to a replica object at the remote site. You can specify multiple role names separated by commas.

-person_name

Specifies the name of a person to change to a replica object at the remote site. You can specify multiple person names separated by commas.

-classoffile

Specifies the class of objects contained in the input file. Valid only with the **-filename** argument. The **-classoffile** argument can specify:

- **User**
The file must contain user IDs of **User** class objects.
- **Role**
The file must contain role names of **Role** class objects.
- **Group**
The file must contain full group names of **Group** class objects.
- **Person**
The file must contain person names of **Person** class objects.

-filename

Specifies the name of a file that contains a list of identifiers for objects of a specified class. Valid only with the **-classoffile** argument.

Note All objects in the file must be separated by a new line character.

-site

Specifies the remote site name.

-h

Displays help for this utility.

ENVIRONMENT

As specified in *Manually configuring your environment for Teamcenter utilities*.

FILES

As specified in *Log files produced by Teamcenter*.

RESTRICTIONS

1. You must set the **IDSMDsaSitesPermittedToPushAdminDat** preference at the remote site specifying the sites that are allowed to push organization data to the local site.
2. You must preform the premigration tasks described in the *Multi-Site Collaboration Guide* before using this utility with the **make_replica** function.
3. After creating a global organization, you must follow the post migration requirements described in the *Multi-Site Collaboration Guide* to properly maintain your organization.
4. The **make_replica** function must be run only at the master site (owning site of global organization data).
5. To migrate a user, the related **Person** object must be migrated first.
6. To migrate a role, for each associated **GroupMember** object, its **User** object must already be migrated.
7. To migrate a group, all of the roles related to the group must be migrated first.
8. Only a top-level group can be migrated. Specifying a sub-group fails. All sub-groups of the specified group are automatically migrated.

EXAMPLES

- Generate a report (**cvrg_cgn.txt**) file showing the duplicate objects between the current site and the remote site (**cgn**).

```
migrate_organization -u=infodba -p=infodba -g=dba -f=compare_organization
-report=cvrg_cgn.txt -site=cgn
```

- Make the users identified in the **cgn_user_rpl.txt** file replica objects at the remote site (**cgn**).

```
migrate_organization -u=infodba -p=infodba -g=dba -f=make_replica
-file=cgn_user_rpl.txt -classoffile=User -site=cgn
```

- Make the identified group objects (**ptrain_dev**, **frame_prod**) replica objects at the remote site (**cgn**).

```
migrate_organization -u=infodba -p=infodba -g=dba -f=make_replica  
-group=ptrain_dev, frame_prod -site=cgn
```

migrate_saved_searches

Updates pre-Teamcenter 8.1 saved search data (which is in the form of user preferences) to the current data model. Current saved search functionality allows users to share saved searches with other users.

SYNTAX

migrate_saved_searches [-u=*user-id* {**-p**=*password* | **-pf**=*password-file*} **-g**=*group*] **-mode**=**upgrade** | **report** [**-owning_users**=**all** | *user-names*] [**-users_of_group**=**all** | *group-names*] [**-file**=*file-name-path*] [**-delete**=**yes** | **no**] [**-log**=*file-name-path*] [**-h**]

ARGUMENTS**-u**

Specifies the user ID.

This is generally **infodba** or another user with administration privileges. If this argument is used without a value, the operating system user name is used.

Note

If Security Services single sign-on (SSO) is enabled for your server, the **-u** and **-p** arguments are authenticated externally through SSO rather than being authenticated against the Teamcenter database. If you do not supply these arguments, the utility attempts to join an existing SSO session. If no session is found, you are prompted to enter a user ID and password.

-p

Specifies the password.

If used without a value, the system assumes a null value. This argument is mutually exclusive with the **-pf** argument.

If this argument is not used, the system assumes the *user-ID* value to be the password.

-pf

Specifies the password file. If used without a value, the system assumes a null value. If this argument is not used, the system assumes the *user-ID* value to be the password.

For more information about managing password files, see [Manage password files](#).

This argument is mutually exclusive with the **-p** argument.

-g

Specifies the group associated with the user.

If used without a value, the user's default group is assumed.

-mode

Determines which function the utility performs. Use **report** to generate a report of legacy saved searches. Use **upgrade** to migrate the legacy saved searches to the current data model.

-owning_users

Migrates the legacy saved search data for the specified users. Use **all** to migrate the data of all users. Enter multiple user names separated by commas. If left unset, the default is **all**.

-users_of_group

Migrates the legacy saved search data for all the uses in the specified groups. Use **all** to migrate the data of all groups. Enter multiple group names separated by commas. If left unset, the default is **all**.

-file

Specifies the path and file name to which the migration report is written. Use this argument with the **report** value. The default location is *current-working-directory/migrate_saved_search_date_report.txt*.

-delete

Determines whether to delete legacy saved search data from the database after migration. Use **yes** to delete the data. The default setting is **no**.

-log

Specifies the path and file name to which any migration errors are written. Use this argument with the **upgrade** value. The default location is *current-working-directory/migrate_saved_search_date_report.log*.

-h

Displays help for this utility.

ENVIRONMENT

As specified in [Manually configuring your environment for Teamcenter utilities](#).

FILES

As specified in [Log files produced by Teamcenter](#).

RESTRICTIONS

None.

EXAMPLES

To migrate all the saved searches belong to the users **john** and **dave**, and to delete the user preferences after migration:

```
migrate_saved_searches -u=infodba -p=password -g=dba -mode=upgrade
-owning_users=john,dave -delete=yes
```

To generates a report of all the saved searches belonging to all users in the **design** and **manufacturing** groups:

```
migrate_saved_searches -u=infodba -p=password -g=dba -mode=report
-users_of_group=design,manufacturing -file=C:\reports\des_mfg_mss.txt
```

To migrate all the saved searches belong to all the users in the database, and to delete the user preferences after migration:

```
migrate_saved_searches -u=infodba -p=password -g=dba -mode=upgrade
-owning_users=all -delete=yes
```

pdx_export

Exports data in PDX format.

SYNTAX

```
pdx_export [-u=admin-id {-p=password | -pf=password-file} -g=dba]
{ -item=item-id [-rev=item-revision] | -item_key=item-key -sitename=target-site}
[-optionset=transfer-option-set][-reason=reason-description]
[-immediate= {True | False} ] [-notify= {True | False} ]
[-emailaddrs=email-address1, email-ddress2, email-addressn]
[-revisionrule=revision-rule-name] [-bomlevel=level-of-BOM-structure]
[-vendors=vendor-name1, vendor-name2, vendor-namen] [-fileoutput=file-name]
[-usegs= {True | False} ]
[-h]
```

ARGUMENTS**-u**

Specifies the user ID. The user must have administrative privileges.

If this argument is used without a value, the operating system user name is used.

-p

Specifies the user's password.

If used without a value, the system assumes a null value. If this argument is not used, the system assumes the *user-ID* value to be the password.

This argument is mutually exclusive with the **-pf** argument.

-pf

Specifies the password file. If used without a value, the system assumes a null value. If this argument is not used, the system assumes the *user-ID* value to be the password.

For more information about managing password files, see [Manage password files](#).

This argument is mutually exclusive with the **-p** argument.

-g

Specifies the group associated with the user. The group value must be **dba** to run this utility.

-item

Specifies the identifier for the item object to export. This argument is mutually exclusive with the **-item_key** argument.

-rev

Specifies the revision of the item object to export. You can use this argument only when you include the **-item** argument.

-item_key

Specifies the a string identifier containing attributes that identify the item object to export. This argument is mutually exclusive with the **-item** argument.

-sitename

Specifies the name of the importing (target) site.

-optionset

Specifies the name of the transfer option set to use for the export.

-reason

Allows you to type a description of the export purpose. This argument is limited to 240 characters. If you enter more than 240 characters, the argument is truncated.

-immediate

Specifies whether to schedule the export or perform it immediately. Valid values are **True** or **False**.

-notify

Specifies whether to send a e-mail notification to the users specified in the **-emailaddrs** argument. Valid values are **True** or **False**.

-emailaddrs

Specifies a comma delimited list of e-mail addresses that are notified of the export.

-revisionrule

Specifies the name of the revision rule to use for the export.

-bomlevel

Specifies the level of BOM to traverse for the export.

-vendors

Specifies a list of vendor names used to filter the content of the exported data. Only objects associated with a vendor in this list are included in the export file. If this argument is not specified, objects associated with any vendor are included.

-file

Specifies the name of the output file containing the PDX export data. If the file exists, it is overwritten.

-usegs

Specifies whether to use Global Services to perform the export.

-h

Displays help for this utility.

ENVIRONMENT

As specified in *Configuring utilities* in the *Utilities Reference*.

FILES

As specified in *Log files* in the *Utilities Reference*.

RESTRICTIONS

You must be logged on as a member of the **dba** group.

plmxml_export

Exports objects from Teamcenter in PLM XML format. If there are files for export, as determined by transfer mode, a directory is created and the files are exported into that directory. The directory is named using the specified file name without the **.xml** extension.

This utility is also used to extract file information from the database into a PLM XML file to prepopulate FSC.

For more information, see the [load_fccache](#) and [fscadmin](#) utilities.

Caution Do not use this utility to export organization objects from sites if the same organization objects are shared through a global organization.

The Business Modeler IDE application maintains most of the objects that affect the data model. Any export operation that can result in one or more of the following objects is not recommended because the generated XML file should not be imported using **plmxml_import** utility to avoid data accuracy issues.

ActivityTypeDef	GRMRule	PropBusinessOperation
AliasTypeDef	HideTypeRule (Type-Display rule)	PropertyRule
AppearanceGroupTypeDef	IdContextRule	RelationTypeDef
ApplicationInterface	ImanTypeDef	StatusTypeDef
CannedMethodRule	ItemTypeDef	StorageMediaTypeDef
ChangeTypeDef	ListOfValues	ToolTypeDef
CompoundPropDefRule	NameFieldRule	TypeBusinessOperation
DatasetTypeDef	NamingRule	UOMTypeDef
DeepCopyRule	NoteTypeDef	ViewTypeDef
Extension	OccurrenceTypeDef	WorkAreaTypeDef
FolderTypeDef	OperationTypeDef	
FormTypeDef	ProcessTypeDef	

SYNTAX

```
plmxml_export [-u=user-id {-p=password | -pf=password-file} -g=group]
-xml_file=xml-file-name -transfermode=transfermode-name
{[-item=item-id | -key=[keyAttr1=keyVal1][keyAttr2=keyVal2]...[,keyAttrN=keyValN]}
[-rev=item-revision-id] [-export bom=yes | no]
[-rev_rule=revision-rule] [-svrule=saved-variant-rule]
| -class=class-name | -type=type-name | -ics_class=ics-class-name
| -imantypedef=iman-type | -uid=uid-of-object | -foldername=folder-to-export}
| [-template=workflow-template-name] [-template_stage=workflow-template-stage]
[-template_classification=workflow-template-classification]}
[-locales=language-ID] [-log=log-file-location] [-h]
```

ARGUMENTS

-u
Specifies the user ID.

This is generally **infodba** or another privileged user. If this argument is used without a value, the operating system user name is used.

Note If Security Services single sign-on (SSO) is enabled for your server, the **-u** and **-p** arguments are authenticated externally through SSO rather than being authenticated against the Teamcenter database. If you do not supply these arguments, the utility attempts to join an existing SSO session. If no session is found, you are prompted to enter a user ID and password.

-p

Specifies the password.

If used without a value, the system assumes a null value. This argument is mutually exclusive with the **-pf** argument.

If this argument is not used, the system assumes the *user-ID* value to be the password.

-pf

Specifies the password file. If used without a value, the system assumes a null value. If this argument is not used, the system assumes the *user-ID* value to be the password.

For more information about managing password files, see [Manage password files](#).

This argument is mutually exclusive with the **-p** argument.

-g

Specifies the group associated with the user.

If used without a value, the user's default group is assumed.

-xml_file

Specifies the full path of the file to which the data is exported.

-transfermode

Specifies the name of the transfer mode used to export the objects. This transfer mode specifies the traversal rules, filter rules, and property sets to be used for export. It determines what is exported from the system. If not specified, a default transfer mode is used. If **-transfermode** is set to **justDatasetsOut**, you must specify a revision ID using the **-rev** argument.

-item

Specifies the ID of the item to be exported.

-key

Specifies the keys of the items to be exported in PLM XML format. Use the following format:

```
[keyAttr1=keyVal1] [,keyAttr2=keyVal2]...[,keyAttrN=keyValN]
```

To find the key of an object, use the [get_key_string](#) utility.

For more information, see the [Business Modeler IDE Guide](#).

-rev

Specifies the revision ID of the item to be exported. If not specified, the configured revision (either specified or default) is exported for the item.

-rev_rule

Specifies the revision rule applied to export the BOM. This is also used to determine the configured revision if the **-rev** option is not specified.

If this option is not specified, the default revision rule is applied, as specified in the **TC_config_rule_name** preference.

-svrule

Specifies the name of the saved variant rule to be applied for the BOM window configuration.

-class

Exports all instances of a given class.

Note

- You cannot use the **-class** argument to export scope rules (**TransferModes**, **ClosureRules**, **FilterRules**, **PropertySets**, **ActionRules**, and **TransferOptionSets**). Use the **tcxml_export** utility.
- To view the persistent object manager (POM) schema, open the **Classes** view in the Business Modeler IDE.
- If **ListOfValues** and **BusinessRule** are specified, all instances of subclasses of specified class are also exported.

Options to export organization information using **-class** are shown in the following table:

Option	Description
Person	Export all persons
User	Export all users
Role	Export all roles
Group	Export all groups
GroupMember	Export all group members
POM_imc	Export all sites
ImanVolume	Export volumes
ListOfValues	Export all list of values
ListOfValuesString	Export string list of values
ListOfValuesDate	Export date list of values
ListOfValuesDouble	Export double list of values
ListOfValuesInteger	Export integer list of values
ListOfValuesChar	Export char list of values
ListOfValuesTag	Export reference list of values

Options to export workspace objects using **-class** are shown in the following table:

Option	Description
Item	Export all instances of item
ItemRevision	Export all instances of item revisions
Folder	Export all instances of folder
Form	Export all instances of forms
Dataset	Export all instances of datasets
Alias	Export all instances of alias
EPMJob	Export all instances of workflow jobs
PSBOMView	Export all instances of BOM view
PSBOMViewRevision	Export all instances of BOM view revision
Tool	Export all instances of tool

Options to export business rules using **-class** are shown in the following table:

Option	Description
BusinessRule	Export all business rules
NameRule	Export all naming rules
NameField	Export all naming field
HideTypeRule	Export all hide type rules of tool
ImanCompoundPropDef	Export all compound property rules of tool
ImanGRM	Export all GRM rules
TypeCannedMethod	Export all action rules

Other options using **-class** are shown in the following table:

Option	Description
FormTypeDef	Export all form type definition
ImanType	Export all instances of Teamcenter types
NoteType	Export all note types
PSViewType	Export all view types
UnitOfMeasure	Export all defined unit of measures
TaskType	Export all defined status
PSOccurrenceType	Export all occurrence types

-type

Exports all instances of a given type.

-ics_class

Exports the specified classification class, if it exists.

-imantypedef

Exports the definition of specified type.

Options and their results for **-imantypedef** are shown in the following table:

Option	Description
ListOfValues	Export list of values
ImanQuery	Export saved queries
Tool	Export tool definitions
TaskType	Export defined status
IdContext	Export identifier contexts
Status	Export defined status
StorageMedia	Export storage media
Note	Export PS occurrence note types
UnitOfMeasure	Export unit of measures defined
Occurrence	Export occurrence types
View	Export view types
RevisionRule	Export revision rules for PS configuration
Alias	Export alias and its types
Identifier	Export identifier and its types
MEWorkArea	Export workarea
MEOP	Export ME operation
ChangeTypeData	Export changeid/changetypen
Dataset	Export dataset type definition
ImanType	Export all Teamcenter types
ImanRelation	Export relations
AppearanceGroup	Export appearancegroup types

-export_bom

Specifies that the BOM is exported. This argument must be used in conjunction with the **-item** argument.

-uid

Exports the object specified by the UID.

-foldername

Exports the specified folder, if it exists.

-template

Specifies the name of the exported workflow template.

-template_stage

Specifies the stage type of the exported workflow template. This argument is used with the **-template** argument. Valid values are **OBSOLETE_STAGE**,

UNDER_CONSTRUCTION_STAGE, or **AVAILABLE_STAGE** (default). If this option is not specified, the default value is used.

-template_classification

Specifies the classification type of the exported workflow template. This argument is used with the **-template** argument. Valid values are **TASK_TEMPLATE** and **PROCESS_TEMPLATE** (default). If this option is not specified, the default value is used.

-locales

Specifies the languages for the export. Separate multiple languages by commas. The language IDs should follow the standard Java locale naming conventions (for example, **en_US**). If no locales are specified for export, the database scalar value (attribute master) is exported to PLM XML scalar fields.

-log

Specifies the full path of the export log file. If this option not specified, the log file is created in the **\$TC_TMP_DIR** directory. If **\$TC_TMP_DIR** is not defined, the log file is created in the system temporary directory (**C:\TEMP** on Windows or **/tmp** on UNIX).

-h

Displays help for this utility.

ENVIRONMENT

As specified in [Manually configuring your environment for Teamcenter utilities](#).

FILES

As specified in [Log files produced by Teamcenter](#).

RESTRICTIONS

- Siemens PLM Software recommends you do *not* use the following transfer modes with this utility:

BOMwriterExport
JTDataExportDefault
TIEPDXExportDefault
TIEUnconfiguredExportDefault
PLMXMLAdminDataExport

- Do not use this utility to export organization objects from sites if the same organization objects are shared through a global organization.

EXAMPLES

- The following command exports item **ABC00001**, revision **A**, to a PLM XML file using the **toPrimeSupplier** transfer mode context. The default revision rule is applied.

```
plmxml_export -u=infodba -p=password -g=dba -xml_file=abc00001_A.xml
-item=ABC00001 -rev=A -transfermode=toPrimeSupplier
```

- The following command exports item **ABC00002**, revision **A** to a PLM XML file using the **toEnterprise** transfer mode context by applying the specified revision rule and saved variant rule to apply to the BOM window:

```
plmxml_export -u=infodba -p=infodba -g=dba -xml_file=abc00002_A.xml
-item=ABC00002 -rev_rule="Latest Released" -svrule="AlphaRelease"
-transfermode=toEnterprise
```

- The following command exports all users to an XML file using the default context to export the data to PLM XML format:

```
plmxml_export -u=infodba -p=infodba -g=dba -xml_file=tcusers.xml  
-class=User
```

- The following command exports an object specified by the UID and uses the default context to export the data to PLM XML format. The UID should be a unique identifier in Teamcenter:

```
plmxml_export -u=infodba -p=infodba -g=dba -xml_file=myobj.xml  
-uid="QRw4LZ0glYomJAAAAAAAAAAAAAAAA"
```

- The following command creates a PLM XML file containing all external file references associated with the top level item selected for cache prepopulation.

```
plmxml_export -u=infodba -p=infodba -g=dba -item=ITEM -rev=A  
-export_bom=yes -transfermode=justDatasetsOut -out=tickets.plmxml
```

- The following command exports the item with ID **item1** to the PLM XML file **item1.xml**. The French and German translations of the localized properties on **item1** that are identified for export (from the property set) are also exported to text elements.

```
plmxml_export -u=infodba -p=infodba -g=dba -item=item1  
-locales=fr_FR,de_DE -xml_file=item1.xml
```


plmxml_import

Imports objects to Teamcenter from a specified PLM XML file. In cases where a transfer mode manages the import, the utility looks for files in the path specified by the **xml_file** argument. This utility is also used to import workflow templates. If the PLM XML file being imported contains translations of localizable properties in multiple languages, the translations of the supported languages are imported into the database.

Caution

Do not use this utility to import organization objects to sites if the same organization objects are shared through a global organization.

The Business Modeler IDE application maintains most of the objects that affect the data model. To avoid data accuracy issues, any PLM XML file that may contain one or more of the following objects should not be imported using the **plmxml_import** utility.

AliasTypeDef	GRMRule	ProcessTypeDef
ApplicationInterface	HideTypeRule (Type-Display rule)	PropBusinessOperation
AppearanceGroupTypeDef	IdContextRule	RelationTypeDef
ActivityTypeDef	ImanTypeDef	StatusTypeDef
CannedMethodRule	ItemTypeDef	StorageMediaTypeDef
ChangeTypeDef	ListOfValues	ToolTypeDef
CompoundPropDefRule	NamingRule	TypeBusinessOperation
DatasetTypeDef	NameFieldRule	UOMTypeDef
DeepCopyRule	NoteTypeDef	ViewTypeDef
Extension	OccurrenceTypeDef	WorkAreaTypeDef
FolderTypeDef	OperationTypeDef	
FormTypeDef	PropertyRule	

SYNTAX

```
plmxml_import [-u=user-id {-p=password | -pf=password-file} -g=group]
-xml_file=name-of-xml-file -transfermode=transfermode-name
[-log=log-file-name] [-import_mode=overwrite | ignore]
[[-apply_template | -ignore_originid]] [-h]
```

ARGUMENTS
-u

Specifies the user ID.

This is generally **infodba** or another privileged user. If this argument is used without a value, the operating system user name is used.

Note If Security Services single sign-on (SSO) is enabled for your server, the **-u** and **-p** arguments are authenticated externally through SSO rather than being authenticated against the Teamcenter database. If you do not supply these arguments, the utility attempts to join an existing SSO session. If no session is found, you are prompted to enter a user ID and password.

-p

Specifies the password.

If used without a value, the system assumes a null value. This argument is mutually exclusive with the **-pf** argument.

If this argument is not used, the system assumes the *user-ID* value to be the password.

-pf

Specifies the password file. If used without a value, the system assumes a null value. If this argument is not used, the system assumes the *user-ID* value to be the password.

For more information about managing password files, see [Manage password files](#).

This argument is mutually exclusive with the **-p** argument.

-g

Specifies the group associated with the user.

If used without a value, the user's default group is assumed.

-xml_file

Specifies the full path of the file name from which the data is imported.

-transfermode

Specifies the name of a transfer mode used to import the objects. This transfer mode specifies the traversal rules, filter rules, and property sets to be used for import. It determines what is imported from the system. If not specified, a default transfer mode is used. In addition the following transfer mode values can be applied to import workflow templates.

workflow_template_import

Use to create a new template. If the template already exists in the database, the command is ignored.

workflow_template_overwrite

Use to overwrite an existing workflow template. The version existing in the database is overwritten by the imported version. If the workflow template does not already exist, a new workflow template is created.

-log

Specifies the full path of the import log file. If this option not specified, the log file is created in the **\$TC_TMP_DIR** directory. If **\$TC_TMP_DIR** is not defined, the log file is created in the system temporary directory (**C:\TEMP** on Windows or **/tmp** on UNIX).

-import_mode

Specifies the mode in which import is handled for PLM XML Import/Export configuration objects. In **overwrite** mode, objects that already exist in the database are overwritten. In **ignore** mode, the imported object is ignored if the imported object already exists in the database.

The classes that function with this argument are:

Teamcenter class name	SDK class name
TransferMode	plmxml60::TransferMode
ClosureRule	plmxml60::ClosureRule
PropertySet	plmxml60::PropertySet
Filter	plmxml60::FilterRule
PIEActionRule	Exported as UserData under plmxml60::TransferMode
Person	plmxml60::Person
TCCalendar	plmxml60::Calendar
User	plmxml60::User
Group	plmxml60::Organisation
Discipline	plmxml60::Discipline
Role	plmxml60::Role
POM_imc	plmxml60::Site
RevisionRule	plmxml60::RevisionRule
ListOfValues	plmxml60::ListOfValues
ImanQuery	plmxml60::SavedQueryDef

-apply_template

When the **workflow_template_overwrite** transfer mode is specified, and the imported workflow template contains changes from the existing workflow template, this argument applies those changes to all active workflow processes based on the workflow template.

For more information about how the workflow template changes are applied, see the [Workflow Designer Guide](#).

This argument must be used with the **workflow_template_overwrite** transfer mode.

The **-apply_template** and **-ignore_originid** arguments are mutually exclusive.

-ignore_originid

Prevents the check of the **origin_id** property and forces the imported workflow template to overwrite the current one. Changes to active workflow processes are not applied.

The **-apply_template** and **-ignore_originid** arguments are mutually exclusive.

-h

Displays help for this utility.

ENVIRONMENT

As specified in [Manually configuring your environment for Teamcenter utilities](#).

FILES

As specified in [Log files produced by Teamcenter](#).

RESTRICTIONS

- Siemens PLM Software recommends you do *not* use the following transfer modes with this utility:

JTDataImportDefault
TIEImportDefault

- Do not use this utility to export organization objects to sites if the same organization objects are shared through a global organization.

EXAMPLES

- To import all objects in the **abc.xml** file using the default context, enter the following command on a single line:

```
plmxml_import -u=infodba -p=infodba -g=dba -xml_file=abc.xml
```

If errors are detected during the import operation, a log file named **abc_log.txt** is created.

- To import the **wkf_templates.xml** workflow template without overwriting existing templates, enter the following command on a single line:

```
plmxml_import -u=infodba -p=infodba -g=dba -xml_file=wkf_templates.xml  
-transfermode=workflow_template_import
```

- To import the **wkf_templates.xml** workflow template that overwrites the existing **wkf_templates.xml** workflow template, creating an updated version of the template in the database, enter the following command on a single line:

```
plmxml_import -u=infodba -p=infodba -g=dba -xml_file=wkf_templates.xml  
-transfermode=workflow_template_overwrite
```

- To import the **wkf_templates.xml** workflow template that overwrites the existing **wkf_templates.xml** workflow template, creating an updated version of the template in the database, and applies all changes from the imported version to all active workflow processes, enter the following command on a single line:

```
plmxml_import -u=infodba -p=infodba -g=dba -xml_file=wkf_templates.xml  
-transfermode=workflow_template_overwrite -apply_template
```

plmxml_tm_edit_xsl

Lists, exports, attaches, or detaches an **.xslt** file to a given transfer mode.

If you are using multiple **.xslt** files, you must run the utility for each file. The files are applied in the order in which you include them in and run the utility.

SYNTAX

```
plmxml_tm_edit_xsl [-u=user-id {-p=password | -pf=password-file} -g=group]
-transfermode=transfermode-name
-action= | list | export | attach | detach | detach_all
-xsl_file=xslt-filename
[-h]
```

ARGUMENTS

-u

Specifies the user ID.

This is generally **infodba** or another user with administration privileges. If this argument is used without a value, the operating system user name is used.

Note

If Security Services single sign-on (SSO) is enabled for your server, the **-u** and **-p** arguments are authenticated externally through SSO rather than being authenticated against the Teamcenter database. If you do not supply these arguments, the utility attempts to join an existing SSO session. If no session is found, you are prompted to enter a user ID and password.

-p

Specifies the password.

If used without a value, the system assumes a null value. This argument is mutually exclusive with the **-pf** argument.

If this argument is not used, the system assumes the *user-ID* value to be the password.

-pf

Specifies the password file. If used without a value, the system assumes a null value. If this argument is not used, the system assumes the *user-ID* value to be the password.

For more information about managing password files, see [Manage password files](#).

This argument is mutually exclusive with the **-p** argument.

-g

Specifies the group associated with the user.

If used without a value, the user's default group is assumed.

-transfermode

Specifies the transfer mode to which the **.xslt** file is exported, attached, or detached.

-action

Performs one of the following actions on the transfer mode:

list

Lists all **.xslt** files associated with the transfer mode.

export

Exports the **.xslt** file to the operating system.

attach

Attaches the **.xslt** file to the transfer mode.

detach

Detaches and removes the **.xslt** file from the transfer mode.

detach_all

Detaches and removes all of the **.xslt** files from the transfer mode

-xslt_file

Specifies the **.xslt** file.

This option is required if the **-action** option is set to **export**, **attach**, or **detach**.

-h

Displays help for this utility.

ENVIRONMENT

As specified in [Manually configuring your environment for Teamcenter utilities](#).

FILES

As specified in [Log files produced by Teamcenter](#).

RESTRICTIONS

None.

step_export

Exports Teamcenter data from the database to STEP-compliant physical files.

There are two ways to use this utility: single-line and batch mode. The single-line method uses the **-item**=*item-id* argument and other optional arguments to export one object at a time; batch mode uses the **-i**=*input-file* argument and an input file to export several objects at once.

SYNTAX

```
step_export [-u=user-id {-p=password | -pf=password-file} -g=group]
  {-i=input-file
  | [-item=item-id | -key=[keyAttr1=keyVal1][,keyAttr2=keyVal2]...[,keyAttrN=keyValN]]
  [-item_rev=item-rev-id] [-ds=dataset] [-rel=relation-name]}
-fmt=AP203 | AP214 | IMAN [-full_assembly]
[-all_ds_versions] [-f=file-name] [-cmt=comments] [-h] [-v]
```

ARGUMENTS

-u

Specifies the user ID.

This is generally **infodba** or another user with administration privileges. If this argument is used without a value, the operating system user name is used.

Note If Security Services single sign-on (SSO) is enabled for your server, the **-u** and **-p** arguments are authenticated externally through SSO rather than being authenticated against the Teamcenter database. If you do not supply these arguments, the utility attempts to join an existing SSO session. If no session is found, you are prompted to enter a user ID and password.

-p

Specifies the password.

If used without a value, the system assumes a null value. This argument is mutually exclusive with the **-pf** argument.

If this argument is not used, the system assumes the *user-ID* value to be the password.

-pf

Specifies the password file. If used without a value, the system assumes a null value. If this argument is not used, the system assumes the *user-ID* value to be the password.

For more information about managing password files, see [Manage password files](#).

This argument is mutually exclusive with the **-p** argument.

-g

Specifies the group associated with the user.

If used without a value, the user's default group is assumed.

-i

Specifies the input file containing the list of objects to export. The complete path name must be provided.

-item

Specifies the item ID of the item being exported.

-item_rev

Specifies the item rev ID of the item revision being exported.

-ds

Specifies the dataset being exported.

-rel

Specifies the name of the Teamcenter relation containing the dataset.

-key

Specifies the keys of the items to export. Use the following format:

`[keyAttr1=keyVal1] [,keyAttr2=keyVal2]...[,keyAttrN=keyValN]`

To find the key of an object, use the [get_key_string](#) utility.

For more information, see the [Business Modeler IDE Guide](#).

-fmt

Specifies the data output format: **AP203**, **AP214**, or **IMAN**.

-full_assembly

Specifies that the full assembly, including product structure and all component parts that constitute the assembly, are exported.

-all_ds_versions

Specifies that all version of the datasets are included in the export.

-f

Specifies the output file name. The complete file specification (full path and file name) must be supplied unless the desired location is the current working directory.

-cmt

Describes the data being exported. This comment is placed in the **file_description** section of the output file.

-v

Runs utility in verbose mode, displaying maximum amount of information. Typically, nonverbose utility sessions only display error messages.

-h

Displays help for this utility.

ENVIRONMENT

As specified in [Manually configuring your environment for Teamcenter utilities](#).

FILES

As specified in [Log files produced by Teamcenter](#) and the following:

`$ROSE_DB/*.rose`

.rose files are STEP schema files used by the STEP Translator. The **step_export** utility must be able to write these files in this directory.

RESTRICTIONS

Either the **-item=***item-id* or the **-i=***input-file* argument must be supplied.

EXAMPLES

To export several objects in batch mode (using an input file), perform the following:

1. Create an input file and add one line for each object you want to export in the following format:

Note Ensure that you separate each argument with a semicolon (;) and put each object on its own line.

```
-item=item-id;-item_rev=item-rev-id;-ds=dataset;-rel=relation-name  
-item=item-id;-item_rev=item-rev-id;-ds=dataset;-rel=relation-name
```

2. Run the **step_export** utility using the **-i=input-file** argument:

```
$TC_ROOT/bin/step_export -u=infodba -p=password -g=dba -i=input-file
```

step_import

Imports product information from STEP-compliant physical files into the Teamcenter database.

SYNTAX

step_import [-u=*user-id* {-p=*password* | -pf=*password-file*} -g=*group*]
{-f=*file-name* | -i=*input-file*} [-h] [-v]

ARGUMENTS**-u**

Specifies the user ID.

This is generally **infodba** or another user with administration privileges. If this argument is used without a value, the operating system user name is used.

Note

If Security Services single sign-on (SSO) is enabled for your server, the **-u** and **-p** arguments are authenticated externally through SSO rather than being authenticated against the Teamcenter database. If you do not supply these arguments, the utility attempts to join an existing SSO session. If no session is found, you are prompted to enter a user ID and password.

-p

Specifies the password.

If used without a value, the system assumes a null value. This argument is mutually exclusive with the **-pf** argument.

If this argument is not used, the system assumes the *user-ID* value to be the password.

-pf

Specifies the password file. If used without a value, the system assumes a null value. If this argument is not used, the system assumes the *user-ID* value to be the password.

For more information about managing password files, see [Manage password files](#).

This argument is mutually exclusive with the **-p** argument.

-g

Specifies the group associated with the user.

If used without a value, the user's default group is assumed.

-f

Specifies a single STEP file. The complete file specification (full path and file name) must be supplied unless the file is in the current working directory.

-i

Specifies the input file containing list of STEP files to batch process. The complete file specification (full path and file name) must be supplied unless file is in the current working directory.

-h

Displays help for this utility.

-v

Runs utility in verbose mode, displaying maximum amount of information. Typically, nonverbose utility sessions only display error messages.

ENVIRONMENT

As specified in [Manually configuring your environment for Teamcenter utilities](#).

FILES

As specified in [Log files produced by Teamcenter](#) and the following:

`$ROSE_DB/*.rose`

.rose files are STEP schema files used by the STEP Translator. The **step_export** utility must be able to write these files in this directory.

RESTRICTIONS

Either the **-f=file-name** or the **-i=input-file** argument must be supplied.

EXAMPLES

None.

sync_form_util

Creates or modifies the **ParticipatingSitesForm** type used to control replication of released assemblies to designated sites.

SYNTAX

```
sync_form_util [-u=user-name {-p=password | -pf=password-file} -g=group]
{-item_id=root-item -rev=
-f={create | add | mv} [forminfo
-project=project-id -sitelist={site1, site2, ..., siten}}
[-h]
```

ARGUMENTS

Note Entries in parentheses are accepted abbreviations for arguments.

-u

Specifies the user ID.

This is generally **infodba** or another user with administration privileges. If this argument is used without a value, the operating system user name is used.

Note If Security Services single sign-on (SSO) is enabled for your server, the **-u** and **-p** arguments are authenticated externally through SSO rather than being authenticated against the Teamcenter database. If you do not supply these arguments, the utility attempts to join an existing SSO session. If no session is found, you are prompted to enter a user ID and password.

-p

Specifies the password.

If used without a value, the system assumes a null value. This argument is mutually exclusive with the **-pf** argument.

If this argument is not used, the system assumes the *user-ID* value is the password.

-pf

Specifies the password file. If used without a value, the system assumes a null value. If this argument is not used, the system assumes the *user-ID* value to be the password.

For more information about managing password files, see [Manage password files](#).

This argument is mutually exclusive with the **-p** argument.

-g

Specifies the group associated with the user.

If used without a value, the user's default group is assumed.

Caution For HTTP enabled sites, remote site operations log on using the default group for the user supplied with the **-u** argument. Any value supplied with the **-g** argument is ignored.

-item_id

Specifies the top item in the structure context object (SCO) assembly.

-rev

Specifies the top item revision used to configure the structure.

-f

Specifies the function to perform.

create

Creates and attaches the **ParticipatingSites** form.

add

Adds the site(s) specified in the **-sitelist** argument to the **SiteList** attribute of the **ParticipatingSites** form.

mv

Removes the site(s) specified in the **-sitelist** argument from the **SiteList** attribute.

forminfo

Retrieves the **ParticipatingSites** form information.

-project

Specifies the name of the project associated with the participating sites.

-sitelist

Specifies a comma delimited list of sites to receive the replicated SCO assembly.

-h

Displays help for this utility.

ENVIRONMENT

As specified in [Manually configuring your environment for Teamcenter utilities](#).

FILES

As specified in [Log files produced by Teamcenter](#).

EXAMPLES

Note Required logon information is omitted from the following examples.

- A participating site form for the **B1_Y** project:

```
sync_form util -item_id=ABC00002 -rev="Latest Released" -f=create -project=B1_Y
-sitelist=CologneEng, AnnArborEng, CambridgeEng
```

sync_on_demand

Synchronizes or reports the synchronization state of a specified component, assembly, or object. The synchronization state indicates whether the replica is up-to-date and whether any object that has been added to the master has been replicated by the site running the utility.

The site that runs the **sync_on_demand** utility and the site that owns the component, assembly, or object being synchronized must both be instances of Teamcenter that support on-demand synchronization. If an assembly contains components from sites that do not support on-demand synchronization, the state of those components are reported as **unknown**.

Component synchronization allows you to determine the state of, or synchronize, all objects associated with the specified revision, such as BVR and attachments.

Assembly synchronization allows you to determine the state of, or synchronize, an entire assembly.

Object synchronization allows you to determine the state of individual objects, such as a dataset or form. You can also select item or item revisions for object synchronization, however the state or objects associated with the item or item revision is not reported or affected.

This utility uses the IDSM process at the remote replica's owning site to accomplish the report task and remote import to accomplish the synchronization.

SYNTAX

```
sync_on_demand [-u=user-id {-p=password | -pf=password-file} -g=group]
-f =sync | report [-uid_report=uid-report-file-name]
-type=object | component | assembly
{-rev_rule=revision-rule | -rev=rev-id} [-assy_level=number]
{[-item_id=item-id | -key=[keyAttr1=keyVal1] [,keyAttr2=keyVal2]... [,keyAttrN=keyValN]
  | template | -folder=folder-name
  | -name=wso-name | -filename=file-name
  | -itemKeyFile=file-name}
[-class=wso-class-name | -classoffile=class-name]
[-exclude=relation-type1] -exclude=relation-type2 ...]
[-include=relation-type3 -include=relation-type4... ]
[-exclude_folder_contents] [-exclude_protected_objects]
[-exclude_protected_comp] [-batch_size=number-of-objects-per-batch]
[-report_file=report-file-name] [-error_report=error-file-name]
[-separator=uid-separator-characters] [-h]
```

ARGUMENTS

Note Entries in parentheses are accepted abbreviations for arguments.

-u

Specifies the user ID.

This is generally **infodba** or another user with administration privileges. If this argument is used without a value, the operating system user name is used.

Note If Security Services single sign-on (SSO) is enabled for your server, the **-u** and **-p** arguments are authenticated externally through SSO rather than being authenticated against the Teamcenter database. If you do not supply these arguments, the utility attempts to join an existing SSO session. If no session is found, you are prompted to enter a user ID and password.

-p

Specifies the password.

If used without a value, the system assumes a null value. This argument is mutually exclusive with the **-pf** argument.

If this argument is not used, the system assumes the *user-ID* value to be the password.

-pf

Specifies the password file. If used without a value, the system assumes a null value. If this argument is not used, the system assumes the *user-ID* value to be the password.

For more information about managing password files, see [Manage password files](#).

This argument is mutually exclusive with the **-p** argument.

-g

Specifies the group associated with the user.

If used without a value, the user's default group is assumed.

-f

Specifies the function to perform. Valid values are **sync** or **report (rep)**. Where **sync** performs a synchronization on the specified object, component, or assembly, and **report** returns the synchronization state.

-uid_report

Returns a report on the specified object, component, or assembly that contains the UIDs of objects enclosed within square brackets ([]) by default. The characters used to enclose the UIDs are configurable through the **-separator** argument. This file can be processed by a custom script to create an input file that the **data_share** utility uses to import objects from a remote site. Using UIDs for remote import increases performance by eliminating the remote query required to determine an item or item revision UID.

-separator (sep)

Designates the start and end characters used to enclose the UID of objects in a **uid_report** file. If this argument is not specified, the UIDs are enclosed in square brackets ([]).

-type

Specifies the type of Teamcenter object on which to perform the function. Valid values are:

- **object (obj)**
- **component (comp)**
- **assembly (assy)**

-rev_rule

Specifies the name of the revision rule used to perform the synchronization or report function. This name must specify an existing revision rule at the local site. This value is passed to the owning site where it is used to determine the item revision to report status for or synchronize. You must supply this argument if the target object is an assembly.

-rev

Specifies the revision ID of the revision to report the status of or synchronize.

-assy_level (al)

Specifies the number of levels of the assembly to report. This argument is valid only for the report function.

-item_id (item)

Specifies the ID or template of items to report the status of or synchronize. Mutually exclusive with the **-name**, **-folder**, and **-filename** arguments.

-key

Specifies the keys of the items to export. Use the following format:

```
[keyAttr1=keyVal1] [,keyAttr2=keyVal2]...[,keyAttrN=keyValN]
```

To find the key of an object, use the [get_key_string](#) utility.

For more information, see the [Business Modeler IDE Guide](#).

-folder (fl)

Specifies the folder that contains the object to report the status of or synchronize. If the folder is not unique, the first folder found that matches this value is used. Mutually exclusive with the **-name**, **-filename**, and **-item_id** arguments.

-name

Specifies the name of a single workspace object to be processed. If not an item, use the **-class** option to specify the class of the object. Mutually exclusive with the **-folder**, **-filename**, and **-item_id** options.

-filename (fn)

Specifies the name of the input file containing IDs or names of objects to report the status of or synchronize. Mutually exclusive with the **-name**, **-folder**, and **-item_id** arguments. If the input file contains names, the **-classoffile** argument is required.

-keyFileName

Specifies the name of the output file containing the keys to export. The file format is:

```
-key = [keyAttr1=keyVal1] [keyAttr2=keyVal2]...
-key = [keyAttr1=keyVal1] [keyAttr2=keyVal2]...
-key = [keyAttr1=keyVal1] [keyAttr2='keyVal2']...
```

-class (cl)

Specifies the Teamcenter class of the object specified by the **-name** argument. This argument is valid only with the **-name** argument. If not specified, **Item** is the default class.

-classoffile (cof)

Specifies the class of objects in the input file. This argument is valid only with the **-filename** argument. If not specified, **Item** is the default class. **Folder** is a valid class for synchronization.

-exclude

Excludes the specified relation type. This argument may be given multiple times and must use the database name (not the display name) of the relation type.

-include

Includes the specified relation type. This argument may be given multiple times and must use the database name (not the display name) of the relation type. Use this argument to force the inclusion of a relation type that may have been excluded during the last export.

-exclude_folder_contents (efc)

Excludes the contents of a folder. Intended for use with NX part families where family members are stored in a folder that is related to the item.

-exclude_protected_object (epo)

Excludes export protected objects.

-exclude_protected_comp (epc)

Excludes export protected components. This argument is valid only when the **-type** argument is set to **assembly**.

-batch_size (bs)

Specifies the number of objects per batch; a new process is created per batch. Default batch size is 1000. Must be a positive integer. This is useful when processing thousands of objects, because it helps avoid memory and disk space shortage problems.

-report_file

Generates a report that is output to the specified file.

If the function specified is **report**, the report contains the synchronization state of each object, component, or components of an assembly.

If the function specified is **sync**, the report shows all successful imports and any errors that occur during the synchronization.

If the function specified is **report** and this argument is not supplied, the report is displayed in a shell (**std out**).

-error_report

Generates a error report that is output to the specified file. This file contains only error information. It provides no synchronization information. If a file name is not supplied, the report is displayed in a shell. This argument is normally used with the **-sync** argument to provide error information during the synchronization process.

-h

Displays help for this utility.

EXAMPLES

Note Required logon information is omitted from the following examples.

- To synchronize an assembly and output any errors that occur to **stdout**:

```
sync_on_demand -f=sync -type=assembly -rev_rule="Latest Working"  
-item_id=Item100/A -error_report
```

The assembly components are synchronized and errors are output to **stdout**.

- To generate a report called **assy_sync.rpt** for the **Item100/A** assembly:

Enter the following command on a single line:

```
sync_on_demand -f=report -type=assy -rev_rule="Latest Working"
               -item_id=Item100/A -report=assy_sync.rpt
```

An example of the contents of the **assy_sync.rpt** file on the owning site:

Component	Owning Site	Sync State	Master LMD	Replica LMD
Item100/A	Site1		09/25/05	09/25/05
Item100/A-view	Site1		09/25/05	09/25/05
Item101/B	Site2	out of date	10/25/05	09/25/05
Item101/B-view	Site2	out of date	10/25/05	09/25/05
Item102/A	Site2		09/25/05	09/25/05
Item102/A-view	Site2		09/25/05	09/25/05
Item103/C	Site1		09/25/05	09/25/05
Item104/B	Site3	unknown		

The report contains components up to the highest level that is out-of-date within a branch. For example, in the sample report, the BVR of component **Item101/B** is out of date and no further expansion is done to show its children in this branch. The branch with **Item102/A** has an up to date BVR so it is expanded until a leaf node is encountered or until a BVR with an out-of-date or unknown status is found.

- To generate a report called **comp_sync.rpt** for the **Item100** component:

Enter the following command on a single line:

```
sync_on_demand -f=report -type=comp -rev_rule="Latest Working"
               -item_id=Item100/A -report=comp_sync.rpt
```

An example of the contents of the **comp_sync.rpt** file on the owning site:

Object String	Type	Relation	Owning Site	Sync State	Master LMD	Replica LMD
Item100	Item			out of date	10/25/05	09/25/05
Item100	Item Master	IMAN_master_form	Site1		09/25/05	09/25/05
Item100-view	BOMView					
Item100/A	Item Revision	Revision	Site1	out of date	10/25/05	09/25/05
Item100/A	Rev Master	IMAN_master_form	Site1		09/25/05	09/25/05
Item100/A-spec	UGMASTER	IMAN_specification	Site1	out of date	10/25/05	09/25/05
Item102/B-ref	Text	IMAN_reference	Site2	unknown		
Item100/A-view	BOMViewRev		Site1		09/25/05	09/25/05

- To create a file (**uids_list.txt**) containing UIDs that can be processed for use as input for the **data_share** utility:

```
sync_on_demand -f=report -uid_report=uids_list.txt
```

- To create both a report file (**uids_list.txt**) containing UIDs and a synchronization report file (**sync.txt**):

```
sync_on_demand -f=report -report_file=sync.txt -uid_report=uids_list.txt
```

- To create a report file (**uids_list.txt**) containing UIDs enclosed by brackets ({ }):

```
sync_on_demand -f=report -uid_report=uids_list.txt -sep="{ }"
```

tcxml_export

Exports objects from Teamcenter in TC XML format. If there are files for export, the utility creates an FMS read file ticket and saves it in the output XML file for each file.

SYNTAX

```
tcxml_export [-u=user-id [-p=password | -pf=password-file] [-g=group] ]
  -file=output-xml-file {-item=item-id [-rev=revision-id] | -folder=folder-name |
  -class=POM-classname | -uid=uid-of-object |
  -item_key=attr-name1=value, attr-name2=value, ...}
  [-transfermode=transfer-mode-name | -optionset=transfer-option-set-name]
  [-targetsites=list-of-target-site-ids]
  [-transferownership]
  [-sync]
  [-incrementalChangeDelta]
  [-reason=reason-for-export]
  [-revrule =revision-rule]
  [-bomlevel =desired-bom-level]
  [-svrule]
  [-processUnconfiguredByOccEff]
  [-processSuppressedOcc]
  [-processUncofiguredVariants]
  [-processUnconfiguredChanges]
  [-requiredLang=locale-code-1, locale-code-2, ..., locale-code-n]
  [-allowedLang=locale-code-1, locale-code-2, ..., locale-code-n]
  [-low-level {-inputfile=file-with-item-ids | -inputuidfile=file-with-uids}
  [-force_retraverse]]
  [-h]
```

ARGUMENTS**-u**

Specifies the user ID.

This is generally **infodba** or another user with administration privileges. If this argument is used without a value, the operating system user name is used.

Note

If Security Services single sign-on (SSO) is enabled for your server, the **-u** and **-p** arguments are authenticated externally through SSO rather than being authenticated against the Teamcenter database. If you do not supply these arguments, the utility attempts to join an existing SSO session. If no session is found, you are prompted to enter a user ID and password.

-p

Specifies the password.

If used without a value, the system assumes a null value. This argument is mutually exclusive with the **-pf** argument.

If this argument is not used, the system assumes the *user-ID* value to be the password.

-pf

Specifies the password file. If used without a value, the system assumes a null value. If this argument is not used, the system assumes the *user-ID* value to be the password.

For more information about managing password files, see [Manage password files](#).

This argument is mutually exclusive with the **-p** argument.

-g

Specifies the group associated with the user.

If used without a value, the user's default group is assumed.

-file

Specifies the output XML file name. The value can be either an absolute path (full path name) or a relative path name.

-item

Specifies the ID of the item to be exported.

-rev

Specifies the revision ID of the item to be exported. If this argument is not specified, the configured revision (either specified or default) is exported for the item.

-folder

Specifies the folder containing objects to be exported.

-class

Specifies a class name. Instances of this POM class are exported. The following workspace object names are valid:

- **Item**
- **ItemRevision**
- **Folder**
- **Dataset**
- **Alias**
- **ImanFile**
- **ImanRelation**
- **ReleaseStatus**
- **IdContext**
- **Identifier**
- **PSBOMView**
- **PSBOMViewRevision**
- **TransferMode**
- **TransferOptionSet**

-uid

Specifies the UID of an object (one object only).

-item_key

Specifies a comma-delimited list of attributes used to identify the object to be exported.

-transfermode

Specifies the transfer mode name used to export the objects. If this argument is not specified, the utility uses a default transfer mode. See restriction 2.

-optionset

Specifies the transfer option set name used to export the objects. Mutually exclusive with the **-transfermode** argument. If you specify both, the command does not fail. However, the transfer mode indicated by this **-optionset** argument takes precedence over the transfer mode specified by the **-transfermode** argument. See restriction 3.

-targetsites

Specifies a comma-delimited list of destination site IDs. If used with the **-transferownership** argument, must contain only one site ID.

-transferownership

Indicates that this export transfers the ownership of exported objects to the given target site. You must specify only one site ID in the **-targetsites** argument if you use this argument.

-sync

Indicates that this export is for data synchronization.

-incrementalChangeDelta

Exports modified objects tracked by configured incremental change as a partial structure export.

Because incremental change data is configured data, you must specify a configured transfer option set for the **-optionset** argument. The **-processUnconfiguredChanges** and **-processSuppressedOcc** arguments are ignored if they are included.

-reason

Specifies the reason for this export.

-revrule

Specifies the revision rule to use to configure the exported BOM with the specified item as the top line.

-bomlevel

Specifies the level in the BOM.

-svrule

Specifies the saved variant rule to use to configure the exported BOM.

-processUnconfiguredByOccEff

Exports **BOMLine** objects that are not configured for occurrence effectivity.

-processSuppressedOcc

Exports suppressed **BOMLine** objects.

-processUnconfiguredVariants

Exports **BOMLine** objects that are not selected by **BOMLine** object's variant conditions.

-processUnconfiguredChanges

Exports **BOMLine** objects configured out of the BOM by incremental change.

-requiredLang

Specifies a list of comma separated locale values. This list is used to ensure that localized attributes in the exported data have at least one representation that can be used as the attribute master language at the importing site. It also defines a priority order for the exporter to determine the attribute master language. The valid locale values must match the Java locale naming convention that consists of two groups of two-character identifiers separated by an underscore character (_) for a particular combination of language and region. For example, **zh_CN** represents Simplified Chinese in China and **en_US** represents English in the United States

-allowedLang

Specifies a list of comma separated locale values. This list is used to get additional representations for localized attributes in the exported data for use at the importing site. The valid locale values must match the Java locale naming convention that consists of two groups of two-character identifiers separated by an underscore character (_) for a particular combination of language and region. For example, **zh_CN** represents Simplified Chinese in China and **en_US** represents English in the United States

-h

Displays help for this utility.

**FAST EXPORT
ARGUMENTS**

The following arguments support low-level fast export functions used for site consolidation activities. The use of these arguments requires the **SITCONS_AUTH_KEY** environment variable be set to a valid license key. You must obtain a key from GTAC.

Export files created using the low-level export do not contain **GSID** attributes.

-low_level

Performs fast import using POM-level APIs and a DBA user. You must specify this argument to use any of the fast export arguments.

You can use any of the following standard **texml_export** arguments, in addition to the other fast export arguments, when you specify the **-low_level** argument.

- folder**
- file**
- targetsites**
- transfermode**
- optionset**
- reason**

-inputfile

Specifies a file that contains a list of item IDs indicating items to export using fast low-level export. You must include either this argument or the **-inputuidfile** argument when using the **-low_level** argument.

If your Teamcenter environment uses multifield key identifiers, you must specify the multifield key values for the **item_id** attribute as a list of comma-separated values in the input file, for example:

```
item_id=M2Item1_001,object_name=M2_Item_name1,object_type=M2Item1
```

The input file may contain both multifield key and standard item ID values, for example:

```
Item_id=Ace1
Item_id=lor1,object_name=fixedPl,object_type=type
Item_id=lor2,object_descr=acmetool,object_type=type
```

-inputuidfile

Specifies a file that contains a list of UIDs indicating items to export using fast low-level export. You must include either this argument or the **-inputfile** argument when using the **-low_level** argument.

-force_retraverse

Forces retraversal of previously replicated or exported objects during fast low-level export.

ENVIRONMENT

As specified in the [Manually configuring your environment for Teamcenter utilities](#).

FILES

As specified in the [Log files produced by Teamcenter](#).

RESTRICTIONS

1. Not all PLM data is supported. For a list of objects that are supported, see the [Data Exchange Guide](#).
2. If you specify the **TransferMode** or **TransferOptionSet** object as the **-class** argument value, you are not required to specify the **-transfermode** or **-optionset** arguments. A predefined transfer mode is used for exporting these objects and if these arguments are specified they are ignored.
3. To export related dataset files, you must specify the **-transfermode** argument. The arguments value must be set to an option set containing closure rules that traverse dataset files related to the primary object. The **TIEExportDefaultTM** transfer mode contains a standard option set that can be used for this purpose.

EXAMPLES

- Select an item and export the item and its attachments using default export transfer mode. The output XML file, **exportitem.xml**, is created in the directory where this command is executed.

```
tcxml_export -u=infodba -p=infodba -g=dba -item=item_id
-file=exportitem.xml
```

- Select an item revision and export its attachments using default export transfer mode. The output XML file, **itemrev.xml**, is created in the directory where this command is executed.

```
tcxml_export -u=infodba -p=infodba -g=dba -item=item_id
-rev=item_rev -file=itemrev.xml
```

- Export the contents of the **exportObjects** folder using default export transfer mode. If objects in the folder are supported objects, they are also exported. The output XML file, **folder.xml**, is created in the directory where this command is executed.

```
tcxml_export -u=infodba -p=infodba -g=dba -folder=exportObjects
-file=folder.xml
```


- Export item **000001** using the **TIEUnconfiguredExportDefault** transfer mode.

```
tcxml_export -u=infodba -p=infodba -g=dba -item=000001
             -file=exportitem.xml -optionset=TIEUnconfiguredExportDefault
```

- Synchronize item **000001**.

```
tcxml_export -u=infodba -p=infodba -g=dba -item=000001
             -file=itemsync.xml -optionset=TransferOptionSet
             -sync -reason=ItemIsOutDated
```

- Export the **Latest Working** revision of the **Top1** item using the **TIEConfiguredExportDefault** transfer option set.

```
tcxml_export -u=infodba -p=infodba -item=Top1
             -optionset=TIEConfiguredExportDefault -revrule="Latest Working"
             -file=D:\temp\Top1_HL.xml
```

- Export a partial structure that includes only the changes to the **Top1** assembly that are tracked by configured incremental change:

```
tcxml_export -u=infodba -p=infodba -item=Top1
             -targetsites=-2054508072 -optionset=TIEConfiguredExportDefault
             -revrule="Latest Working" -sync -incrementalChangeDelta
             -file=/tmp/0001_delta.xml
```

- Fast export the objects identified in the **inp.txt** file using the **VARIANTRULE1** saved variant rule to site **56781234**.

```
tcxml_export -u=infodba -p=infodba -low_level -inputfile=d:\Temp\inp.txt
             -file=d:\Temp\top1_ll.xml -optionset=TIEConfiguredExportDefault
             -svrule=VARIANTRULE1 -targetsites=-2054508072
```

- Fast export the **Latest Working** revisions of the objects, including suppressed BOM lines and BOM lines configured out by incremental changes.

```
tcxml_export -u=infodba -p=infodba -low_level -inputfile=d:\Temp\inp.txt
             -file=d:\Temp\top1_ll.xml -optionset=TIEConfiguredExportDefault
             -revrule="Latest Working" -svrule=VARIANTRULE1 -processSuppressedOcc
             -processIncrementalChanges
```

The following are site consolidation examples:

- To export the objects to a specified site.

```
tcxml_export -u=infodba -p=infodba
             -optionset=SiteConsolidationDefault -targetsites=-2054508072
             -inputfile=d:\input.txt -file=d:\out.xml -low_level
```

- To synchronize the objects already exported (low level) to a specified site.

```
tcxml_export -u=infodba -p=infodba -sync
             -file=d:\out.xml -low_level
```

Special cases for **tcxml_export**:

- Export in-process workflow Items:
 1. Export the item in the workflow.
 2. In the rich client, search for workflow job corresponding to that item.
 3. Copy this job and paste it into a new folder with unique name.

4. Export this folder to export the job.

Note When a workflow process is exported with transfer ownership to the target site, the **My Worklist® Inbox® Tasks to Perform** folder at the target site does not display the task. To display the task, the affected users must delete the **Inbox** and restart the rich client.

Similarly, to display tasks in **ResourcePool Inbox**, delete its **Tasks to Track** and **Tasks to Perform** subfolders.

- Export a collaboration context object (CCO):
 1. In the rich client, search for the CCO.
 2. Copy the CCO and paste it into a new folder with a unique name.
 3. Export this folder to export the CCO and the entire structure context.
 4. Export the associated product structures.
- Low-level synchronization process example:
 1. Create the tables and triggers using an SQL script before exporting the data.

You can use an edited version of the following sample script, located in your **TC_ROOT/install/sitecons** directory, to manage the tables in separate tables spaces for the site consolidation tables. Before you run the script, edit the highlighted parameter values for your site.

```

sitcons_create_tablespace.sql
/* Copyright 2009.
   Siemens Product Lifecycle Management Software Inc.
   All Rights Reserved. */
/* This is a sample script that can be used by the administrator.
   The following parameters namely datafile, size, autoextend on maxsize,
   extent management local uniform size are the variables that need to be
   changed as required. Also, the tablespace can be named as desired */
/* Creates a separate table space to be used later for ACCT_TABLE creation */
create tablespace TCSITCONS datafile
'D:\oracle\product\10.2.0\oradata\test\TCSITCONS.dbf'
size 10M
autoextend on maxsize 100M
extent management local uniform size 64K;
/* Creates the table ACCT_TABLE using above table space */
create table ACCT_TABLE(
exp_obj_uid varchar2(15) PRIMARY KEY,
led date,
island anchor uid varchar2(15),
state NUMBER)
tablespace TCSITCONS;
create table SCRATCH_TABLE(
puid varchar2(15),
lsd date,
trigger condition NUMBER)
tablespace TCSITCONS;
CREATE INDEX "SCRATCH_TABLE_INDEX" ON "SCRATCH_TABLE" (puid);
create or replace trigger fast_sync_add_trigger
before insert on PPOM_OBJECT
referencing new as newRow
for each row
BEGIN
insert into scratch_table values (:newRow.puid, :newRow.plsd, '8');
END;
/
create or replace trigger fast_sync_delete_trigger
after delete on PPOM_OBJECT
referencing old as oldRow
for each row
BEGIN
insert into scratch_table values (:oldRow.puid, :oldRow.plsd, '9');
END;
/

```

2. Export the data using the **texml_export** utility in low-level mode:

```
tcxml_export -u=infodba -p=infodba -optionset=SiteConsolidationDefault
-targetsite=-2054508072 -inputfile=d:\input.txt -file=d:\out.xml -low_level
```

3. Import the data at the target site using the **tcxml_import** utility in low-level mode:

```
tcxml_import -u=infodba -p=infodba -g=dba -file=d:\out.xml -low_level
```

4. Copy the import-generated **out_import_results.txt** file to the source site and run the **tcxml_confirm_export** utility in low-level mode:

```
tcxml_confirm_export -u=infodba -p=infodba -g=dba
-file=d:\out_import_results.txt -low_level
```

5. Add, delete, and change the low-level exported data as needed.

6. Run the **tcxml_export** utility with the **-sync** argument.

```
tcxml_export -u=infodba -p=infodba -g=dba -file=d:\sync.xml -low_level -sync
-optionset=SiteConsolidationDefault -inputfile=d:\input.txt
```

7. Import the generated XML file using **tcxml_import** low-level mode.

8. Repeat the earlier step to copy the import-generated **out_import_results.txt** file to the source site and run the **tcxml_confirm_export** utility in the low-level mode.

```
tcxml_confirm_export -u=infodba -p=infodba -g=dba
-file=d:\out_import_results.txt -low_level
```

Note

Fast synchronization operations depend on the timestamp on the local server machine where an object is saved and edited. Therefore, modifying objects on different machines with different system time may influence the identification of out-of-sync status.

tcxml_import

Imports objects into Teamcenter from a TC XML file.

Additionally, this utility is used in conjunction with the [attribute_export](#) utility to update the values of object properties in your Teamcenter database in bulk.

For more information about updating property values in bulk, see the [System Administration Guide](#).

SYNTAX

```
tcxml_import  
[-u=user-id{-p=password | -pf=password-file} -g=group]  
-file=xml-file-name  
[ [-xsl=xsl-file-name]  
[-errorcontinue={yes | no} ]  
[ [-site=site-name]  
[-transfermode=transfer-mode]  
[-optionset=option-set-name] ] |  
[ [-scope_rules [-scope_rules_mode=ignore | overwrite] ] ]  
[-requiredLang=locale-code-1, locale-code-2, ..., locale-code-n]  
[-allowedLang=locale-code-1, locale-code-2, ..., locale-code-n] ] |  
[-low_level] |  
[-bulk_load -site=site-name ]  
[-bypassSiteCheck]  
[-h]
```

ARGUMENTS**-u**

Specifies the user ID.

This is generally **infodba** or another user with administration privileges. If this argument is used without a value, the operating system user name is used.

Note If Security Services single sign-on (SSO) is enabled for your server, the **-u** and **-p** arguments are authenticated externally through SSO rather than being authenticated against the Teamcenter database. If you do not supply these arguments, the utility attempts to join an existing SSO session. If no session is found, you are prompted to enter a user ID and password.

-p

Specifies the password.

If used without a value, the system assumes a null value. This argument is mutually exclusive with the **-pf** argument.

If this argument is not used, the system assumes the *user-ID* value to be the password.

-pf

Specifies the password file. If used without a value, the system assumes a null value. If this argument is not used, the system assumes the *user-ID* value to be the password.

For more information about managing password files, see [Manage password files](#).

This argument is mutually exclusive with the **-p** argument.

-g

Specifies the group associated with the user.

If used without a value, the user's default group is assumed.

-file

Specifies the input TC XML file containing the objects to imported into Teamcenter.

-xsl

Specifies the input XSL file to apply to the TC XML file before import.

-errorcontinue

Indicates whether to continue import after encountering an error. The default value is **yes**. If you specify **no** and the utility encounters an error, the utility rolls back all of the changes performed during the current import.

-site

Specifies the master exporting site from which input TC XML data is generated. This argument is mutually exclusive with the **-scope_rules** and **-scope_rules_mode** arguments.

-transfermode

Specifies the transfer mode name that is to be used for import. If this argument is not specified, the utility uses the **TIEImportDefault** transfer mode. This argument is mutually exclusive with the **-scope_rules** and **-scope_rules_mode** arguments.

-optionset

Specifies the option set name that contains options to use during import. This argument is mutually exclusive with the **-scope_rules** and **-scope_rules_mode** arguments.

-scope_rules

Specifies the input XML data contains scope rules, that is, transfer modes, closure rules, filter rules, property sets, actions rules, or transfer options sets. This argument is mutually exclusive with the **-site**, **-transfermode**, and **-optionset** arguments. See [Restrictions](#).

-scope_rules_mode

Specifies the import behavior when a rule is imported that already exists in the database. If set to **ignore** the rule is not imported. If set to **overwrite**, the existing database rule is overwritten. If you do not specify this argument, **ignore** behavior is used. If you do not specify the **-scope_rules** argument, this argument is invalid.

-optionset

Specifies the option set name that contains options to use during import.

-requiredLang

Specifies a list of comma separated locale values. This list is used to ensure that localized attributes in the imported data have at least one representation that can be used as the attribute master language at the importing site. It also defines a priority order for the exporter to determine the attribute master language. The valid locale values must match the Java locale naming convention that consists of two groups of two-character identifiers separated by an underscore character (__) for a particular

combination of language and region. For example, **zh_CN** represents Simplified Chinese in China and **en_US** represents English in the United States

-allowedLang

Specifies a list of comma separated locale values. This list is used to get additional representations for localized attributes in the imported data for use at the importing site. The valid locale values must match the Java locale naming convention that consists of two groups of two-character identifiers separated by an underscore character (__) for a particular combination of language and region. For example, **zh_CN** represents Simplified Chinese in China and **en_US** represents English in the United States.

-low_level

Performs a fast import using POM level APIs. Fast import is normally used for consolidation of site databases. You must specify only the **-file** argument; all other arguments are ignored during fast imports. To use this argument, you must set a license key value for the **SITCONS_AUTH_KEY** environment variable. You must obtain the key from GTAC. For more information about consolidating sites see the [Site Consolidation Guide](#).

-bulk_load

Performs a fast import of legacy data from a file containing low-level TC XML formatted data. You must specify only the **-file** and **-site** arguments; all other arguments are ignored when bulk loading low-level data. If you specify any other optional arguments they are ignored. To use this argument you must set a license key value for the **SITCONS_AUTH_KEY** environment variable. You must obtain the key from GTAC.

For more information about loading bulk data, see the [Data Exchange Guide](#).

-bypassSiteCheck

If specified, this switch bypasses the check that prevents the utility from updating objects at the same site.

If specified, this switch bypasses the check that prevents the update of replica objects

Note In a multi-site environment, the utility must be run at each of the sites to update the objects belonging to the respective sites.

Users have the capability of performing updates on replica objects, eliminating the need to re-replicate all updated objects

Caution Use this argument only with the **-bulk_load** argument for bulk update of attributes of local objects and import of archived audit records.

-h

Displays help for this utility.

ENVIRONMENT

As specified in the [Manually configuring your environment for Teamcenter utilities](#).

FILES

As specified in the [Log files produced by Teamcenter](#).

RESTRICTIONS

- When importing transfer option sets, if a local option set at the exporting site is specified for import, the utility assigns the local site as the site reference and

the imported option becomes local to the importing site. If a remote option at the exporting site is specified for import, the site reference of that option set is expected to exist in the database at the importing site. If it is not, the import fails. To avoid this failure, you must manually create the option set before attempting the import.

- Not all Teamcenter data is supported for import. For a list of objects that are supported, see the [Data Exchange Guide](#).

EXAMPLES

- The following example imports objects specified by the **-file** argument:

```
tcxml_import -file=xml-file-name -u=userid -p=password
```

- The following example imports objects specified by the **-file** argument according to the rules given in transfer mode specified by the **-transfermode** argument:

```
tcxml_import -file=xml-file-name -u=userid -p=password
             -transfermode=transfer-mode-name
```

- The following example imports objects specified by the **-file** argument and uses the value specified by the **-site** argument as the exporting site for this import:

```
tcxml_import -file=xml-file-name -u=userid -p=password
             -site=site-id
```

- The following example imports objects specified by the **-file** argument and uses the value specified by the **-xsl** argument to apply transformation on the input XML file:

```
tcxml_import -file=xml-file-name -u=userid -p=password
             -xsl=xsl-file
```

- The following example imports objects specified by the **-file** argument and uses the value specified by the **-errorcontinue** argument to determine whether to roll back the import if an error occurs:

```
tcxml_import -file=xml-file-name -u=userid -p=password
             -errorcontinue=yes
```

- The following example imports objects specified by the **-file** argument and uses the value specified by the **-optionset** argument to access the options that must be used during import:

```
tcxml_import -file=xml-file-name -u=userid -p=password
             -optionset=option-set-name
```

- The following example imports objects specified **-file** argument with localizable attribute values in **en_US** and **fr_FR** locales.

```
tcxml_import -u=infodba -p=infodba -g=dba -file=exportitem.xml
             -requiredlanguage=en_US -allowedlanguage=fr_FR
```

- The following example imports archived audit records.

```
tcxml_import -u=infodba -p=infodba -g=dba -bulk_load -bypassSiteCheck
             -file=c:/temp/AuditExport.xml
```

tcxml_validate

Validates the contents of a low-level TC XML formatted file. This file may be an a file exported from a Teamcenter site or constructed by a custom solution that maps legacy system data to the Teamcenter unified data model. You can use this to validate the file information prior to importing it into a Teamcenter site or to compare the Teamcenter data created by importing the file to the contents of the imported file. This utility requires a license key. For information about obtaining and setting the license key, contact GTAC.

SYNTAX

```
tcxml_validate [-u=user-id {-p=password | -pf=password-file} -g=group]  
{-file=input-xml-file -action={pre | post}}  
[-h]
```

ARGUMENTS**-u**

Specifies the user ID.

This is generally **infodba** or another user with administration privileges. If this argument is used without a value, the operating system user name is used.

Note

If Security Services single sign-on (SSO) is enabled for your server, the **-u** and **-p** arguments are authenticated externally through SSO rather than being authenticated against the Teamcenter database. If you do not supply these arguments, the utility attempts to join an existing SSO session. If no session is found, you are prompted to enter a user ID and password.

-p

Specifies the password.

If used without a value, the system assumes a null value. This argument is mutually exclusive with the **-pf** argument.

If this argument is not used, the system assumes the *user-ID* value to be the password.

-pf

Specifies the password file. If used without a value, the system assumes a null value. If this argument is not used, the system assumes the *user-ID* value to be the password.

For more information about managing password files, see [Manage password files](#).

This argument is mutually exclusive with the **-p** argument.

-g

Specifies the group associated with the user.

If used without a value, the user's default group is assumed.

-file

Specifies the input TC XML file name. The value can be either an absolute path (full path name) or a relative path name.

-action

When set to **pre**, checks the named file for conformance to the importing sites schema, logical data model, and business constraints. The output of the utility is written to a log file that uses the naming convention *input-xml-file-name_validator.log*. This file uses a pipe (|) character as a field delimiter and can be imported into Microsoft Excel for easy reading. This file contains any errors and discrepancies found by the utility.

Note The **post** action is not available at this time. Use the **pre** action.

When set to **post** compares the named file contents to the data created when the file was imported to verify the objects where imported properly.

-h

Displays help for this utility.

ENVIRONMENT

As specified in [Manually configuring your environment for Teamcenter utilities](#).

FILES

As specified in [Log files produced by Teamcenter](#).

RESTRICTIONS

You must acquire a license key from GTAC and set the **SITCONS_AUTH_KEY** environment variable to this value to use this utility.

EXAMPLES

The following command validates a file (**0001.xml**) and creates a log file (**0001_validator.log**) containing the results of the validation.

```
tcxml_validate -u=infodba -p=infodba -file=D:\temp\0001.xml -action=pre
```

tcxml_xfer_ownership

DESCRIPTION

Transfers the ownership of a set of Teamcenter objects. You can use this utility to change ownership of all objects owned by a site during site consolidation, change ownership of related objects during transition from Teamcenter Enterprise to Teamcenter (flip-the-switch ownership transfer), or change object ownership when importing objects into Teamcenter from another data management system.

For performance reasons during site consolidation, this utility changes ownership of objects using SQL calls, not the POM API. Rich client refresh works only when POM calls are used to change an attribute, including ownership changes. Therefore, ownership changes that the utility enacts during an active rich client session are not reflected in the interface. You must restart all active rich clients to get the changes. This is normally not an issue because user access must be prohibited during the *time critical period* when this utility is used.

A valid license key is required to use this utility for site consolidation ownership changes.

The **perform_highlevel** argument enables the flip-the-switch ownership transfer feature of this utility. By default, during flip-the-switch ownership transfer, this utility uses the POM API to change object ownership. Therefore, ownership changes that the utility enacts during an active rich client session are reflected in the interface. You are not required to restart the active rich clients to get the changes.

For better performance, you can set the **TIE_flip_using_SQL** preference to **true** to use SQL calls during flip-the-switch ownership transfers. If you use this preference setting, you must restart all active rich clients to get the changes.

A valid license key is not required to use this utility when you enable the flip-the-switch feature.

SYNTAX

```
tcxml_xfer_ownership -u=user-name {-p=password | -pf=password-file} [-g=group]
-action={extract | perform | update_status |
perform_highlevel | perform_highlevel_dryrun}
[-inputfile=file-with-object-list]
[-change_ownership_to=site-ID]
[-dryrun [-startDate=MM/DD/YYYY -endDate=MM/DD/YYYY] [-source_extinct]]
[-file=status-output-file]
[-report=report-file-name]
[-h]
```

ARGUMENTS**-u**

Specifies the user ID.

This is generally **infodba** or another user with administration privileges. If this argument is used without a value, the operating system user name is used.

Note

When Security Services single sign-on (SSO) is enabled for your server, the **-u** and **-p** arguments are authenticated externally through SSO, rather than being authenticated against the database.

- If you do not supply these arguments, the utility attempts to join an existing SSO session.
- If no session is found, you are prompted to enter a user ID and password.

-p

Specifies the user's password.

If used without a value, the system assumes a null value. If this argument is not used, the system assumes the *user-ID* value to be the password.

This argument is mutually exclusive with the **-pf** argument.

-pf

Specifies the password file. The file must be a single-line ASCII file containing the password in clear text. Teamcenter Environment Manager prompts you for a password and creates the password file during installation.

If used without a value, the system assumes a null value. If this argument is not used, the system assumes the *user-ID* value to be the password.

This argument is mutually exclusive with the **-p** argument.

-g

Specifies the group associated with the user.

If used without a value, the user's default group is assumed.

-action

Specifies whether to extract, perform, or update status.

extract

Generates the file for ownership transfer.

The **-change_ownership_to**, **-report**, or **-dryrun** arguments are supported for this action.

perform

Transfers the ownership at target site.

The **-inputfile**, **-file**, **-source_extinct**, or **-dryrun** arguments are supported for this action.

update_status

Updates the status at source site.

The **-inputfile**, **-change_ownership_to**, **-source_extinct**, **-report**, or **-dryrun** arguments are supported for this action.

perform_highlevel

Performs a high-level (flip-the-switch) ownership transfer at the target site

The **-inputfile** and **-change_ownership_to** arguments are supported for this action. Any other arguments are ignored.

This action generates two outputs files, the *input-file-name* **flip_objects.log** file lists all objects transferred and the *input-file-name* **results.xml** file contains the flip status of objects given in input file.

perform_highlevel_dryrun

Generates a report to preview the ownership change of objects transferred by a high-level (flip-the-switch) ownership transfer at the target site.

The **-inputfile** and **-change_ownership_to** arguments are supported for this action. Any other arguments are ignored.

-inputfile

Specifies the name of the file containing the objects for ownership change.

-change_ownership_to

Specifies the site ID of the target site. This argument is required when the utility is run at the source site. If this argument is omitted, the utility uses the local site as the target site. If either the **-dryrun** or the **perform_highlevel_dryrun** argument is specified, the ownership transfer is not performed.

-dryrun

Generates a report to preview the ownership change of objects transferred by the site consolidation tools, and identifies inconsistencies.

To preview the objects at the target site that are owned by the source site and are inconsistent with the source site due to modifications made to them at the target site post-replication, set the **TIE_DRYRUN_VALIDATION=TRUE** preference at the target site.

The **-startDate** and **-endDate** arguments are valid only with the **-dryrun** argument. The utility generates a dry run report for objects transferred within the date range specified by the **-startDate** and **-endDate** values.

-startDate

Specifies the starting date for replicas and inconsistencies reported by the **-dryrun** argument. This argument is used in conjunction with the **-endDate** argument and is ignored if the **-dryrun** argument is not specified. The date supplied must be in supplied in a *MM/DD/YYYY* format.

-endDate

Specifies the ending date for replicas and inconsistencies reported by the **-dryrun** argument. This argument is used in conjunction with the **-startDate** argument and is ignored if the **-dryrun** argument is not specified. The date supplied must be in supplied in a *MM/DD/YYYY* format.

-source_extinct

Generates the list of replica items at the target site which are owned by the source site and which were not transferred by site consolidation tools.

This argument can be used only when the **-dryrun** argument is used and is ignored otherwise.

-file

Specifies the name of the output file (with an absolute path) that contains the status of ownership transfer of the objects at the target site. This argument is valid only for the site consolidation **-action=perform** argument.

-report

Specifies the name of the report file or extract file. This argument is valid only for site consolidation actions.

Note To use the dry run mode to check objects of an island owned by the source site that are not covered in the accountability table for ownership transfer, set the **TIE_DRYRUN_VALIDATION** preference to **TRUE** at the target site.

For debug information, set the following environment variables:

- **TC_SLOW_SQL=1**
- **TC_SQL_DEBUG=BJPT**

ENVIRONMENT

As specified in the [Manually configuring your environment for Teamcenter utilities](#).

FILES

As specified in the [Log files produced by Teamcenter](#).

RESTRICTIONS

None.

EXAMPLES

- To run at the source site and extract to a file the objects for ownership transfer:

```
tcxml_xfer_ownership -u=infodba -p=infodba -g=dba
-action=extract -report=D:\Temp\ownership_transfer_extract.txt
-change_ownership_to=target_site_id [-dryrun]
```

- To transfer the ownership or validate the inconsistencies for ownership transfer at the target site:

```
tcxml_xfer_ownership -u=infodba -p=infodba -g=dba
-action=perfrom -inputfile=ownership_transfer_extract.txt
-file=ownership_transfer_status.txt [-dryrun] [-source_extinct]
```

- To update the ownership transfer status or to generate report for a dry run at the source site:

```
tcxml_xfer_ownership -u=infodba -p=infodba -g=dba
-action=update_status -inputfile=ownership_transfer_status.txt
-report=ownership_transfer_update_status.txt
-change_ownership_to=target_site_id [-dryrun]
```

- To use flip-the-switch ownership transfer to change ownership at the target site:

```
tcxml_xfer_ownership -u=infodba -p=infodba -g=dba
-action=perform_highlevel -inputfile=flip_the_switch_input.txt
-change_ownership_to=target_site_id
```

- To use flip-the-switch ownership transfer to determine ownership change at the target site without actually changing ownership:

```
tcxml_xfer_ownership -u=infodba -p=infodba -g=dba
-action=perform_highlevel_dryrun -inputfile=flip_the_switch_input.txt
-change_ownership_to=target_site_id
```

- The following example shows the contents of a sample file used as the input for a flip-the-switch ownership transfer (**-inputfile=flip_the_switch_input.txt**):

```
<?xml version="1.0" encoding="UTF-8"?>
<!-- flip_the_switch_input.txt file -->
<TcFlipXML
```

```

    author="super user" date="2010-03-24" time="16:10:50" language="en" schemaVersion="2.0" >
    <GSIdentity class="Cmponent" context="" factor="" atomic="false"
      label="udzjduesvsun0sugms2--aRZ" split_token="" sub-label="" system="213456789"
      transient_island_id="udzjdvcsvsun0sugms2--aRZ" elemId="1" />
    <GSIdentity class="CmpnMstr" context="" factor="" atomic="false"
      label="udzjdvcsvsun0sugms2--aRZ" split_token="" sub-label="" system="213456789"
      transient_island_id="udzjdvcsvsun0sugms2--aRZ" elemId="2" />
    <GSIdentity class="Assembly" context="" factor="" atomic="false"
      label="udzjdwksvsun0sugms2--aRZ" split_token="" sub-label="" system="213456789"
      transient_island_id="udzjdwksvsun0sugms2--aRZ" elemId="3" />
    <GSIdentity class="AssmMstr" context="" factor="" atomic="false"
      label="udzjdxisvsun0sugms2--aRZ" split_token="" sub-label="" system="213456789"
      transient_island_id="udzjdwksvsun0sugms2--aRZ" elemId="4" />
  </TcFlipXML>

```

You can construct the input file manually by copying the **GSIdentity** elements from the TC XML files used for importing the objects.

upload_plmxml_struct

Uses the item ID and revision ID of the top node of the assembly and the revision rule and executes the **bomwriter** utility with options specified in the arguments. The PLM XML file generated is then attached as a named reference to the **DirectModelAssembly** dataset.

SYNTAX

```
upload_plmxml_struct [-u=user-id {-p=password | -pf=password-file} -g=group]
[-item=item-id | -key=[keyAttr1=keyVal1][,keyAttr2=keyVal2]...[,keyAttrN=keyValN]]
-rev=revision-id -rev_rule=revision-rule [-h]
```

ARGUMENTS

-u

Specifies the user ID.

This is generally **infodba** or another user with administration privileges. If this argument is used without a value, the operating system user name is used.

Note

If Security Services single sign-on (SSO) is enabled for your server, the **-u** and **-p** arguments are authenticated externally through SSO rather than being authenticated against the Teamcenter database. If you do not supply these arguments, the utility attempts to join an existing SSO session. If no session is found, you are prompted to enter a user ID and password.

-p

Specifies the password.

If used without a value, the system assumes a null value. This argument is mutually exclusive with the **-pf** argument.

If this argument is not used, the system assumes the *user-ID* value to be the password.

-pf

Specifies the password file. If used without a value, the system assumes a null value. If this argument is not used, the system assumes the *user-ID* value to be the password.

For more information about managing password files, see [Manage password files](#).

This argument is mutually exclusive with the **-p** argument.

-g

Specifies the group associated with the user.

If used without a value, the user's default group is assumed.

-item

Specifies the item ID of the top node of the assembly structure.

-key

Specifies the key of the of the top node of the assembly structure.

```
[keyAttr1=keyVal1] [,keyAttr2=keyVal2]...[,keyAttrN=keyValN]
```

To find the key of an object, use the [get_key_string](#) utility.

For more information, see the [Business Modeler IDE Guide](#).

-rev

Specifies the revision ID of the top node of the assembly structure.

-rev_rule

Specifies the revision rule to use to apply on the assembly and create the structure output.

-h

Displays help for this utility.

ENVIRONMENT

As specified in [Manually configuring your environment for Teamcenter utilities](#).

FILES

As specified in [Log files produced by Teamcenter](#).

RESTRICTIONS

None.

EXAMPLES

Open the Teamcenter menu shell with the database connection variables set and execute the following command:

```
upload_plmxml_struct -u=infodba -p=infodba -g=dba -item=000125  
-rev=001 -rev_rule="Latest Working"
```


validate_and_replicate_assembly

Validates an assembly with checks based on parameters and generates a PLM XML file containing a list of all components in a configured assembly. The utility gets a list of participating sites using the project that the assembly root item is assigned to and invokes the replication mechanism to update the assembly components to all the participating sites. Use this utility only from a translator task to generate PLM XML output for an assembly that is being released.

SYNTAX

```
validate_and_replicate_assembly [-u=user-name {-p=password |  
-pf=password-file}  
-g=group] -item_id=assembly-root -rev=revision -revision_rule=revision-rule  
[-variant_rule=rule-to-configure-BOM] [-check_precise] [-check_no_stubs]  
[-check_all_released] [-h]
```

ARGUMENTS

Note Entries in parentheses are accepted abbreviations for arguments.

-u

Specifies the user ID.

This is generally **infodba** or another user with administration privileges. If this argument is used without a value, the operating system user name is used.

Note If Security Services single sign-on (SSO) is enabled for your server, the **-u** and **-p** arguments are authenticated externally through SSO rather than being authenticated against the Teamcenter database. If you do not supply these arguments, the utility attempts to join an existing SSO session. If no session is found, you are prompted to enter a user ID and password.

-p

Specifies the password.

If used without a value, the system assumes a null value. This argument is mutually exclusive with the **-pf** argument.

If this argument is not used, the system assumes the *user-ID* value is the password.

-pf

Specifies the password file. If used without a value, the system assumes a null value. If this argument is not used, the system assumes the *user-ID* value to be the password.

For more information about managing password files, see [Manage password files](#).

This argument is mutually exclusive with the **-p** argument.

-g

Specifies the group associated with the user.

If used without a value, the user's default group is assumed.

Caution For HTTP enabled sites, remote site operations log on using the default group for the user supplied with the **-u** argument. Any value supplied with the **-g** argument is ignored.

-item_id

Specifies the top item in the assembly structure.

-rev

Specifies the top item revision used to configure the structure.

-revision_rule

Specifies the name of the revision rule used to configure the assembly.

-variant_rule

Specifies the variant rule used to configure the assembly. If not specified, the default variant rule set at the site level is used.

-check_precise

Performs a validation check on the complete assembly to ensure that all sub assemblies are precise.

-check_no_stubs

Performs a validation check on the complete assembly to ensure that there are no stubs in the assembly.

-check_all_released

Performs a validation check on the entire assembly to ensure that all components are released.

-h

Displays help for this utility.

ENVIRONMENT

As specified in [Manually configuring your environment for Teamcenter utilities](#).

FILES

As specified in [Log files produced by Teamcenter](#).

EXAMPLES

Note Required logon information is omitted from the following examples.

The following command validates that the **assy_root** assembly is a precise assembly with no stubs and all components are released. It also replicates the assembly to all participating sites:

```
validate_and_replicate_assembly -u= -p= -g=dba -item_id=assy_root -rev=A
-revision_rule=Latest Released -check_precise -check_no_stubs
-check_all_released
```

Chapter

8 *Visualization utilities*

harvest_mmv_index

Creates or updates the spatial cell index for a particular product specified by an item revision. The spatial cell index describes the spatial structure of the massive product data and is used by viewer clients to quickly load and render the product geometry.

The utility traverses the product structure and locates all unconfigured occurrences that contain a JT dataset. Each occurrence is identified by the revision independent path from the root to this occurrence. The bounding box information is computed for each occurrence, and the bounding box is rolled up to the root coordinate system by applying the accumulated transform. The rolled-up bounding boxes are used to create a spatial tree, which is saved as an **.mmv** file. This **.mmv** file is uploaded as a named reference of a **Fnd0SpatialHierarchy** dataset.

Syntax

```
harvest_mmv_index [-u=user-id {-p=password | -pf=password-file} -g=group]  
[-item=item-id | -key=item-key | -itemid=root-item][-rev=revision]  
[-rev_rule=revisionRuleList][-log=file-name][-h=]
```

Arguments

-u

Specifies the user ID.

This is generally **infodba** or another privileged user. If this argument is used without a value, the operating system user name is used.

Note If Security Services single sign-on (SSO) is enabled for your server, the **-u** and **-p** arguments are authenticated externally through SSO rather than being authenticated against the Teamcenter database. If you do not supply these arguments, the utility attempts to join an existing SSO session. If no session is found, you are prompted to enter a user ID and password.

-p

Specifies the password.

If used without a value, the system assumes a null value. This argument is mutually exclusive with the **-pf** argument.

If this argument is not used, the system assumes the *user-ID* value to be the password.

This argument is mutually exclusive with the **-pf** argument.

-pf

Specifies the password file. The file must be a single-line ASCII file containing the password in clear text. Teamcenter Environment Manager prompts you for a password and creates the password file during installation.

If used without a value, the system assumes a null value. If this argument is not used, the system assumes the *user-ID* value to be the password.

This argument is mutually exclusive with the **-p** argument.

-g

Specifies the group associated with the user.

If used without a value, the user's default group is assumed.

-item

Specifies the item ID for the root of the BOM structure.

-key

Specifies a string used to identify an item instead of specifying an item using the **-item** and **-rev** arguments.

To find the key of an object, use the [get_key_string](#) utility.

For more information, see the [Business Modeler IDE Guide](#).

-itemuid

Specifies the UID to identify the item of the product to be harvested. If this option is specified, the **-item** and **-key** options cannot be specified.

-rev

Specifies the item revision for the root of the BOM structure.

-rev_rule

Specifies the revision rules that are supplied for the product that is to be harvested. Each specified revision rule is delimited by a comma. If no revision rule is supplied, the product is harvested as unconfigured.

-log

Specifies the name for the log file. By default, the log file is *harvest_spatial_index_time.log*.

-h

Displays help for this utility.

Environment

As specified in [Manually configuring your environment for Teamcenter utilities](#).

Files

As specified in [Log files produced by Teamcenter](#).

Restrictions

None.

Examples

- To harvest a specific revision of a product (**item_id=000004**), configured with revision rules **Latest Working** and **Latest Released**, type the following command on a single line:

```
harvest_mmv_index -u=adminjones -p=passjones -g=admin -item=000004 -rev=001  
-rev_rule="Latest Working, Latest Released"
```

- To harvest spatial cell indexes for a specific revision of an unconfigured product (**item_id=000004**) using the multifield key, type the following command on a single line:

```
harvest_mmv_index -u=adminjones -p=passjones -g=admin -key=000004-key -rev=001
```

Chapter

9 *Repeatable Digital Validation
(RDV) utilities*

bomwriter

Emits a bill of materials (BOM), in a variety of file formats, to a nominated file optionally restricted to selected areas of the BOM.

For example, you can use this utility to export product structure information from Teamcenter to a PLM XML file that can be consumed by several applications. If the size of the product structure is very large (several thousand occurrences) you may run the export overnight. In such situations, consider using the [PLMXML_sdk_threshold](#) preference to minimize memory consumption during the export.

For more information about using this preference to serialize PLM XML objects, therefore reducing memory consumption during large exports, see the [Preferences and Environment Variables Reference](#).

SYNTAX

```
bomwriter [-u=user-id {-p=password | -pf=password-file} -g=group]
[-noprompt] [-h | -help]
{-bookmark=bookmark-file |
-item_list=input-file-name |
-item=item-name | -key=item-key}
[-rev=revision]
[-selected=input-file-name]
[-subselected=input-file-name]
[-output_file=output-file-name]
[-revision_rule=configuration-rule]
[-show_substitutes=true | false]
[-show_unconfigured=true | false]
[-show_unconfigured_occ_eff=true | false]
[-show_variants=true | false]
[-view=view-type-name]
[-descendants=true | false]
[-flatten=true | false]
[-packed_window=true | false]
[-transient_unpack=true | false]
[-smstring=true | false]
[-svrule=saved-variant-rule-name]
-format=format where format is one of the following:
    index
    psup
    plmxml
    ajt
```

ARGUMENTS**-u**

Specifies the user ID.

This is generally **infodba** or another privileged user. If this argument is used without a value, the operating system user name is used.

Note If Security Services single sign-on (SSO) is enabled for your server, the **-u** and **-p** arguments are authenticated externally through SSO rather than being authenticated against the Teamcenter database. If you do not supply these arguments, the utility attempts to join an existing SSO session. If no session is found, you are prompted to enter a user ID and password.

-p

Specifies the password.

If used without a value, the system assumes a null value. This argument is mutually exclusive with the **-pf** argument.

If this argument is not used, the system assumes the *user-ID* value to be the password.

-pf

Specifies the password file. If used without a value, the system assumes a null value. If this argument is not used, the system assumes the *user-ID* value to be the password.

For more information about managing password files, see [Manage password files](#).

This argument is mutually exclusive with the **-p** argument.

-g

Specifies the group associated with the user.

If used without a value, the user's default group is assumed.

-noprompt

Specifies that if autologin fails, the system will not prompt for an interactive login. This is a standard autologin option.

-help

Displays help for this utility.

-bookmark

Specifies a simple bookmark file from which to extract the default root item ID, root revision ID, and revision rule (overruled by any revision rule specified on the command line). The parsing of this file is primitive: one complete XML element per line is expected.

-item_list

Specifies a list of item IDs, one per line, with optional output file names on the same line. If no file name clause is provided, the default is **itemid.format** without **+** options. These items are processed successively.

-item

Specifies the item ID for the root of the BOM structure.

-key

Specifies a string used to identify an item instead of specifying an item using the **-item** and **-rev** arguments.

To find the key of an object, use the [get_key_string](#) utility.

For more information, see the [Business Modeler IDE Guide](#).

-rev

Specifies the item revision for the root of the BOM structure.

-view

Specifies the view to be used.

-output_file

Specifies the output file. The **stdout** file is the default.

-format

Specifies the output file format with optional modifiers, for example:

```
-format=ajt+native+asm_jt_file
```

-format=index

Format used with the **-selected** or **-subselected** option. No modifiers.

-format=psup

psup format with the **+props=xxx** modifier representing comma-separated BOM line properties. You can also specify a delimiter using the **delimiter** option. The default delimiter is a comma (,).

For example, to specify the semicolon as a delimiter:

```
-format=psup+delimiter=;
```

+props=xxx

Comma-separated BOM line properties are exported.

+delimiter=x

Single-character delimiter is used to separate each column in the output.

-format=plmxml

plmxml format with the following modifiers:

+strict

Minor errors are fatal.

+type= xxx

Use nondefault builder.

+tmode= xxx

Use nondefault transfer mode.

+transform=[None | Cumulative | Absolute]

Use specified **transformType** on Occurrence.

+locales=language_codes

Specifies the languages for the export. Separate multiple languages by commas (for example, **en_US**, **fr_FR**). The language IDs follow the standard locale naming conventions (for example, **en_US**). If no locales are specified for export, the database scalar value (attribute master) is exported to PLM XML scalar fields.

+ua= xxx

User attributes specifier.

The syntax is a combination of a target specifier followed by a key specifier followed by a property specifier, all separated by commas. For example:

```
-format=plmxml+ua=target:<target name>,key:<key name>,prop:<prop name>...
```

The target, key, and property values are controlled by the **INTEROP_ExtraPLMXMLInstanceAttributes** preference. If there is more than one key/property value specified for a given target, you can specify each without repeating the **target** keyword. The properties are exported as **UserValue** under the **UserData** element under the **target** element.

In the following example, the **UserData** element appears under the **Occurrence** element as it is the target.

```
<UserValue value="prop_value" title="key"></UserValue>
```

In the following example, the user exports the **bl_rev_owning_user** property of **BOMLine** under the **Occurrence** element with a title of **OwningUser**. The same applies to **bl_rev_owning_group**.

```
-format=plmxml+ua=target:Occurrence,key:OwningUser,prop:bl_rev_owning_user,  
key:OwningGroup,prop:bl_rev_owning_group
```

+revid_off

Turn off revision ID in the PRV name attribute.

+varuid_on

Turn on variant UID.

+grdvua_on

Turn on user attributes for GRDVA in instance element.

-format=ajt

ajt format with the following modifiers:

+nt

ajt file attribute in Windows format. For example:

```
d:\folder\tk0404c2_mod_5q8050016xwq6.jt
```

+unix

ajt file attribute in UNIX format. For example:

```
/folder/tk0404c2_mod_5q8050016xwq6.jt
```

+native

ajt file attribute in machine-native (UNIX or Windows) format.

+uidtag

ajt file attribute in **uidtag** format. For example:

```
BVHRD95$1V1P$n$hD.jt
```

+identity

Causes a missing transform to become an identity transform rather than a fatal error.

+strict

Minor errors are fatal.

+skip_fake_part

Skips dummy part for subassemblies.

+asm_jt_file

Outputs the associated JT files (if any) information for any intermediate lines in the assembly.

-selected

Specifies the input file to nominate particular lines as selected, defaults to the root line if none is selected. If the specified input file is empty, the output should contain configuration information only (no BOM lines), where supported.

Run **bomwriter** once with the **-f=index** argument, edit the resulting file to remove lines that should not be selected, and run **bomwriter** again with the same parameters, but **-f** to the format you want and **-s** indicating the edited **-f=index** file.

Note

Review the index output file carefully before using it as the input for this option. If the output contains any special characters, for example: new line, the **-selected** option may not function properly.

-subselected

Specifies the subselected items file. Edit **-f=index** for format.

-descendants

Specifies whether descendants of selected lines should be included in the output, using **true** or **false** values. Defaults to **true**. Selected lines and ancestors of selected lines are always included.

-flatten

Presents the selected lines as a tree in which the root node is the immediate parent of all selected lines. The transforms of the selected lines are combined with the transforms of their ancestor's lines to compensate for their disappearance. Valid values are **true** and **false**. Defaults to **false**.

The **-flatten** argument reverses the default for the **-descendants** argument, because the **-flatten** argument never needs descendent lines.

-packed_window

Use a packed BOM window. Valid values are **true** and **false**. The default value is **false**.

Do not use with the PLM XML format.

-transient_unpack

Use transient unpacking. Valid values are **true** and **false**. The default value is **false**.

Do not use with the PLM XML format.

-smstring

Use **smstring** output, printed to **stdout**. Valid values are **true** and **false**. The default value is **false**.

-revision_rule

Specifies a named revision rule. Defaults to the site default, frequently the **latest working** revision.

-show_substitutes

Specifies that substitutes be shown. Defaults to the site default value.

-show_unconfigured

Specifies that unconfigured lines be shown. Defaults to the site default value.

-show_unconfigured_occ_eff

Specifies that lines unconfigured by occurrence effectivity be shown. Defaults to the site default value.

-show_variants

Specifies that unconfigured variants be shown. Defaults to the site default value.

-svrule

Specifies the BOM is to be configured based on the given saved variant rule.

-h

Displays help for this utility.

ENVIRONMENT

As specified in [Manually configuring your environment for Teamcenter utilities](#).

FILES

As specified in [Log files produced by Teamcenter](#).

RESTRICTIONS

None.

EXAMPLES

- Run the **bomwriter** utility with the following arguments to write an **AJT** file for compilation (using **asciitotjt**) on the current platform where some, but not all, BOM lines have transform matrices:

```
bomwriter -bookmark=somefile.bkm -format=ajt+native+identity
-output_file=somefile.ajt
```

- Run the **bomwriter** utility with the following arguments to write selected parts of a BOM window in **AJT** format (without descendants). First, create the index file from which to select parts:

```
bomwriter -item=XYZ001 -format=index -output_file=xyz001.index
```

- Edit the **selected.index** file to remove various lines and then run the utility as follows:

```
bomwriter -item=XYZ001 -format=ajt+native -selected=xyz001.index
-descendants=false -output_file=xyz001.ajt
```

- Output associated JT files information for any intermediate lines for item **XYZ001**.

```
bomwriter -item=XYZ001 -format=ajt+native+asm_jt_file
-output_file=xyz001.index -rev=A
```

- The following command calls the **bomwriter** utility with PLM XML format output on **item1** and produces the **Export_WithTranslations.plmxml** file. It also writes the **bl_item_object_desc** BOM line property as **Item_Desc UserData** under the **Occurrence** element. The French and German translations of the localized properties on **item1** are also exported to **Text** elements.

```
bomwriter -u=user -p=password -g=group -item=item1  
-output_file=Export_WithTranslations.plmxml  
-format=plmxml+ua=target:Occurrence,key:"Item_Desc",prop:  
"bl_item_object_desc"+locales=fr_FR,de_DE
```

harvester.pl

Updates the QPL database with the latest changes in the product structure. The QPL database is used when performing spatial and attribute queries. Use the **-run_level** argument to run this utility in a setting similar to a UNIX **crontab** command, allowing you to achieve a finer level of granularity and control with this utility.

Note To access the Teamcenter volumes containing JT data, this utility and the FMS service must all be run by the same user.

Below is a list of the **harvester.pl** components and a description of their functionality:

update_structure

Updates the top-level cache in the NX part file with the latest changes in the product structure.

ug_spacemap

Creates the cell occupancy map in binary format. Attribute schema and the attribute details of the NX file are created in XML format.

QPL upload

Uploads the spatial and attribute data into the QPL database.

SYNTAX

```
$QPL_ROOT/scripts/harvester.pl [-u=user-id {-p=password | -pf=password-file} -g=group] -run_level=value -k -l -r -v -h
```

ARGUMENTS

-u

Specifies the user ID.

This is generally **infodba** or another user with administration privileges. If this argument is used without a value, the operating system user name is used.

Note If Security Services single sign-on (SSO) is enabled for your server, the **-u** and **-p** arguments are authenticated externally through SSO rather than being authenticated against the Teamcenter database. If you do not supply these arguments, the utility attempts to join an existing SSO session. If no session is found, you are prompted to enter a user ID and password.

-p

Specifies the password.

If used without a value, the system assumes a null value. This argument is mutually exclusive with the **-pf** argument.

If this argument is not used, the system assumes the *user-ID* value to be the password.

-pf

Specifies the password file. If used without a value, the system assumes a null value. If this argument is not used, the system assumes the *user-ID* value to be the password.

For more information about managing password files, see [Manage password files](#).

This argument is mutually exclusive with the **-p** argument.

-g

Specifies the group associated with the user.

If used without a value, the user's default group is assumed.

-run_level

Defines which component of the harvester process is to be run:

1= update_structure

2= ug_spacemap

3= update_structure + ug_spacemap

4= QPL upload

5= update_structure + QPL upload

6= ug_spacemap + QPL upload

7= update_structure + ug_spacemap + QPL upload

-k

Preserves the intermediate files created by the **ug_spacemap** utility.

-l

Specifies symbolic links followed in UNIX.

-r

Recursively searches the search directory specified in the **options.txt** file.

-v

Logs the current status of the executables run by this utility in to the terminal.

-h

Displays help for this utility.

ENVIRONMENT

The environment variables are provided by the variables defined in the **\$QPL_ROOT/qpldata/options.txt** file and the **\$QPL_ROOT/qpldata/config.txt** file. This script uses all the variables defined in the two aforementioned files as environment variables. For more information regarding the variables, see the **options.txt** file in the **\$QPL_ROOT/qpldata** directory.

FILES

As specified in [Log files produced by Teamcenter](#).

RESTRICTIONS

None.

EXAMPLES

The following examples demonstrate use of this utility in a command line and also in a **crontab** setting, which is most likely to be used in a production environment.

- The following command runs the utility at run level **7**, which runs all possible executables. These are run on bookmark files located in the search directory pointed to by the **QPL_H_root_dirs** variable in the **options.txt** file in the **\$QPL_ROOT/qpldata** directory:

harvester.pl

- The following command runs the utility at run level **3** on the bookmark file named **BOOKMARK1.bkm** in the directory from which the utility is being run:

```
harvester.pl BOOKMARK1.bkm -run_level=3
```

- The following command runs the utility at run level **7** on bookmark files found in the directory named *anyDir*, which is a subdirectory existing in the current directory:

```
harvester.pl anyDir
```

- The following example illustrates using the **harvester.pl** script in a **crontab** setting:

```
10 * * 1-6 harvester.pl BOOKMARK1.bkm -run_level=3
14 * * 1-6 harvester.pl BOOKMARK1.bkm -run_level=3
18 * * 1-6 harvester.pl BOOKMARK1.bkm -run_level=3
22 * * 1-6 harvester.pl BOOKMARK1.bkm -run_level=3
2 * * 1-6 harvester.pl BOOKMARK1.bkm -run_level=4
12 * * 1-6 harvester.pl BOOKMARK2.bkm -run_level=3
16 * * 1-6 harvester.pl BOOKMARK2.bkm -run_level=3
20 * * 1-6 harvester.pl BOOKMARK2.bkm -run_level=3
0 * * 1-6 harvester.pl BOOKMARK2.bkm -run_level=3
4 * * 1-6 harvester.pl BOOKMARK2.bkm -run_level=4
```

harvester_jt.pl

Updates the QPL database with the latest changes in the product structure based on the tessellated representation of the assembly. The QPL database is used when performing spatial and attribute queries.

Note OpenGL runtime libraries are required for this utility to run.

A list of the **harvester_jt.pl** utility components and a description of their functionality follows:

bomwriter

Creates a textual representation of the product structure in the form of an **.ajt** file. The file contains information regarding all the BOM line attributes in the product structure. This file is then read by the **asciitojt** utility to create the **.jt** file representation of the entire structure.

asciitojt

Creates the **.jt** file representation of the entire structure from the text file created by the **bomwriter** utility.

SpaceMapJt

Creates the voxel representation of the assembly from the binary **.jt** file created by the **asciitojt** utility.

QPL utilities

Creates the cell occupancy map and the XML representation of the attributes from the output of the **SpaceMapJt** utility, and then uploads them to the QPL database

SYNTAX

```
$QPL_ROOT/scripts/harvester_jt.pl [-u=user-id {-p=password |  
-pf=password-file} -g=group] -zone_min=x1, y1, z1 -zone_max=x2, y2, z2 -k -l -r -v
```

ARGUMENTS**-u**

Specifies the user ID.

This is generally **infodba** or another user with administration privileges. If this argument is used without a value, the operating system user name is used.

Note

If Security Services single sign-on (SSO) is enabled for your server, the **-u** and **-p** arguments are authenticated externally through SSO rather than being authenticated against the Teamcenter database. If you do not supply these arguments, the utility attempts to join an existing SSO session. If no session is found, you are prompted to enter a user ID and password.

-p

Specifies the password.

If used without a value, the system assumes a null value. This argument is mutually exclusive with the **-pf** argument.

If this argument is not used, the system assumes the *user-ID* value to be the password.

-pf

Specifies the password file. If used without a value, the system assumes a null value. If this argument is not used, the system assumes the *user-ID* value to be the password.

For more information about managing password files, see [Manage password files](#).

This argument is mutually exclusive with the **-p** argument.

-g

Specifies the group associated with the user.

If used without a value, the user's default group is assumed.

-zone_min

Defines the minimum coordinates of the two-dimensional plane to be considered while generating the occupancy map. It is provided in an x1, y1, z1 format.

-zone_max

Defines the maximum coordinates of the two-dimensional plane to be considered while generating the occupancy map. It is provided in an x2, y2, z2 format.

-k

Preserves the intermediate files created by the **ug_spacemap** utility.

-l

Symbolic links followed in UNIX.

-r

Recursively searches the search directory specified in the **options.txt** file.

-v

Logs the current status of the executables run by the **harvester_jt.pl** utility into the terminal.

-h

Displays help for this utility.

ENVIRONMENT

The environment variables are provided by the variables defined in the **\$QPL_ROOT/qpldata/options.txt** and the **\$QPL_ROOT/qpldata/config.txt** files. This script uses all the variables that are defined in the two aforementioned files as environment variables. For more information regarding the variables, refer to the **options.txt** file in the **\$QPL_ROOT/qpldata** directory.

FILES

As specified in [Log files produced by Teamcenter](#).

RESTRICTIONS

None.

EXAMPLES

The following examples demonstrate use of this utility in a command line:

- The following command runs the utility on bookmark files located in the search directory pointed to by the **QPL_H_root_dirs** variable in the **options.txt** file in **\$QPL_ROOT/qpldata** directory:

```
harvester_jt.pl
```

- The following command runs the utility on the bookmark file named **BOOKMARK1.bkm** in the directory from which the utility is being run:

```
harvester_jt.pl BOOKMARK1.bkm
```

- The following command runs the utility on bookmark files found in the directory named *anyDir*, which is a subdirectory existing in the current directory:

```
harvester_jt.pl anyDir
```

qpl_convert_occs2tc

Converts the NX subfile-ID instance paths to Teamcenter occurrence unique identifier (UID) instance paths in the attribute data file. This attribute data file is generated by the **ug_spacemap** utility. The utility updates the attribute data file as well as the schema file. It is mandatory to provide a name for the attribute file as well as the schema file as part of the input parameters.

The utility uses the **UG ENTITY HANDLE** attribute on a BOM line to generate the subfile-ID string from the BOM line.

Entity handles are generally in the following format:

```

0                               n           n + 8           n + 16
-----xxxxxxxxxxxxxxxx-----end

```

xxx is the subfile ID and **n + 16** is the end of the string. It is in hexadecimal representation.

This method generates the instance paths of BOM lines in terms of NX subfields and Teamcenter occurrence UIDs and adds them to the **sfid_uid** mapper. These strings are generated by concatenating the subfile-IDs (or Teamcenter UIDs) of BOM lines up until the top BOM line.

SYNTAX

```

qpl_convert_occs2tc [-u=user-id {-p=password | -pf=password-file} -g=group]
[-item=item-name | -key=item-key]
-rev=revision
-revision_rule=revision-rule-name
-view=view-name
-attr_file=attribute-file-name
-schema_file=schema-file-name
-h

```

ARGUMENTS

-u

Specifies the user ID.

This is generally **infodba** or another privileged user. If this argument is used without a value, the operating system user name is used.

Note

If Security Services single sign-on (SSO) is enabled for your server, the **-u** and **-p** arguments are authenticated externally through SSO rather than being authenticated against the Teamcenter database. If you do not supply these arguments, the utility attempts to join an existing SSO session. If no session is found, you are prompted to enter a user ID and password.

-p

Specifies the password.

If used without a value, the system assumes a null value. This argument is mutually exclusive with the **-pf** argument.

If this argument is not used, the system assumes the *user-ID* value to be the password.

-pf

Specifies the password file. If used without a value, the system assumes a null value. If this argument is not used, the system assumes the *user-ID* value to be the password.

For more information about managing password files, see [Manage password files](#).

This argument is mutually exclusive with the **-p** argument.

-g

Specifies the group associated with the user.

If used without a value, the user's default group is assumed.

-item

Specifies the item ID for the root of the BOM structure.

-key

Specifies a string used to identify an item instead of specifying an item using the **-item** and **-rev** arguments.

To find the key of an object, use the [get_key_string](#) utility.

For more information, see the [Business Modeler IDE Guide](#).

-rev

Specifies the item revision for the root of the BOM structure.

-revision_rule

Specifies a named revision rule. Defaults to the site default, frequently the **latest working** revision.

-view

Specifies the view to be used.

-attr_file

Specifies the file to contain the converted attributes.

-schema_file

Specifies the file to contain the converted schema.

-h

Displays help for this utility.

ENVIRONMENT

As specified in [Manually configuring your environment for Teamcenter utilities](#).

FILES

As specified in [Log files produced by Teamcenter](#).

RESTRICTIONS

None.

rdv_context_download

Evaluates or reevaluates a structure context object (SCO) and writes DesignContext information and user attributes into a PLM XML or AJT (**BOMWriter**) file. The contents of the SCO can represent the entire product or a subassembly. The information written into the PLM XML file depends on the specified input transfer mode.

The output file is used to facilitate the sharing of data between users, programs, or sites using Multi-Site Collaboration. You can download this file to the operating system or import it into Teamcenter as a dataset.

This utility can process a single SCO, a list of SCOs, or all SCOs in a specified folder. If you process a list or folder of SCOs, a separate PLM XML file is generated for each SCO. It can also accept a saved query as input.

If you use Multi-Site Collaboration, the [validate and replicate assembly](#) utility can consume the PLM XML files created by this utility and replicate them across sites.

If the output PLM XML file is in **BOMWriter** format, you can launch it in Lifecycle Visualization.

SYNTAX

```
rdv_context_download [-u=user-id {-p=password | -pf=password-file} -g=group]
[-item_id=item-ID] | -key=[keyAttr1=keyVal1][,keyAttr2=keyVal2]...[,keyAttrN=keyValN]
[-rev_id=revision-ID] [-variant_rule_name=variant-rule-name]
[-revision_rule_name=revision-rule-name] [-engg_change_id=engineering-change-ID |
-engg_change_key=[keyAttr1=keyVal1][,keyAttr2=keyVal2]...[,keyAttrN=keyValN]
[-folder_name=folder-name] [-process_name=process-name]
[-zone_name=zone-name] [-zone_type=BOX | PLANE]
[-operator=BOX(Within | Outside | Interferes) | Plane(Above | Below | Intersects)]
[-sco_download=yes | true | YES | TRUE]
[-user_attribute=user-attribute]
[-sco_name(s)=StructureContextObjectName(s) |
-saved_query_name=Teamcenter-saved-query-name |
-sco_folder=Teamcenter-folder-name] [-transfer_mode=transfer-mode-name]
[-output_format=BOM_writer FormatAJT | PLMXML |
PIEPIE-PLMXML] [-file_name=file-name-without-extension]
[-absolute_path=path-of-file-to-be-stored-without-extension] [-h]
```

ARGUMENTS

-u

Specifies the user ID.

This is generally **infodba** or another user with administration privileges. If this argument is used without a value, the operating system user name is used.

Note

If Security Services single sign-on (SSO) is enabled for your server, the **-u** and **-p** arguments are authenticated externally through SSO rather than being authenticated against the Teamcenter database. If you do not supply these arguments, the utility attempts to join an existing SSO session. If no session is found, you are prompted to enter a user ID and password.

-p

Specifies the password.

If used without a value, the system assumes a null value. This argument is mutually exclusive with the **-pf** argument.

If this argument is not used, the system assumes the *user-ID* value to be the password.

-pf

Specifies the password file. If used without a value, the system assumes a null value. If this argument is not used, the system assumes the *user-ID* value to be the password.

For more information about managing password files, see [Manage password files](#).

This argument is mutually exclusive with the **-p** argument.

-g

Specifies the group associated with the user.

If used without a value, the user's default group is assumed.

-item_id

Specifies the item ID.

-key

Specifies the key of the item. Use the following format:

```
[keyAttr1=keyVal1] [,keyAttr2=keyVal2]...[,keyAttrN=keyValN]
```

To find the key of an object, use the [get_key_string](#) utility.

For more information, see the [Business Modeler IDE Guide](#).

-rev_id

Specifies the revision ID.

-variant_rule_name

Specifies the name of the variant rule configuring the structure.

-revision_rule_name

Specifies the name of the revision rule configuring the structure, which defaults according to the [TC_config_rule_name](#) preference.

-engg_change_id

Specifies the engineering change item ID. The utility configures an RDV context based on change attachments and the latest engineering change revision.

-engg_change_key

Specifies the key of the engineering change item. The utility configures an RDV context based on change attachments and the latest engineering change revision.

To find the key of an object, use the [get_key_string](#) utility.

For more information, see the [Business Modeler IDE Guide](#).

-folder_name

Configures a context based on attachments of the *folder/envelope/engineering-change-revision-name*.

The attachments include the following:

- One product item revision
- One or more component item IDs
- Optional revision rule, overwrites the **-r** argument
- Optional variant rule, overwrites the **-v** argument

Note Search results are affected by the following preferences:

- [TC_config_rule_name](#)
- [WebDesignContextDefaultSearchDistance](#)
- [PortalDesignContextMaxMatchingObjects](#)
- [PortalDesignContextMaxMatchingBOMLines](#)

For more information, see the [Preferences and Environment Variables Reference](#).

-process_name

Specifies the name of the workflow processes that have not yet completed.

-zone_name

Specifies the name of the zone. When both the **-zone_name** and **-zone_type** arguments are specified, the utility performs a search according to the preference settings.

-zone_type

Specifies the type of the zone, either **BOX** or **PLANE**. The **-zone_name** argument must be used in conjunction with the **-zone_type** argument.

-operator

Specifies the zone type operator. Valid values for the **BOX** zone type are **Within**, **Outside**, or **Interferes**; the default value is **Within**. Valid values for the **PLANE** zone type are **Above**, **Below**, and **Intersects**; the default value is **Intersects**. The **-zone_name** and **-zone_type** arguments must be specified in conjunction with the **-operator** argument.

Note This utility performs a proximity search if the **-zone_name** argument is not specified and performs a name zone search if the **-zone_type** argument is not specified.

-sco_download

Specifies if the SCO should be downloaded.

-user_attribute

Specify user attributes in the same format described for the [bomwriter](#) utility.

-sco_name

Specifies the name of the structure context object from which a PLM XML is generated. You can specify more than one SCO as a comma-separated list, for example:

SCO1,SCO2,SCO3

-saved_query_name

Specifies the name of a Teamcenter saved query that retrieves a set of SCOs to process for given search criteria.

-sco_folder

Specifies the name of a Teamcenter folder that contains a set of SCOs to process.

-transfer_mode

Specifies the name of a transfer mode to generate the PLM XML file. If no mode is specified, the SCOs are evaluated with the default transfer mode.

-output_format

Specifies the output file format, either BOM writer PLM XML or AJT, or PIE PLM XML. The default format is BOM writer PLM XML.

-file_name

Specifies the name of the output file to which the data is output. Provide the file name without an extension.

-absolute_path

Specifies the full path of the AJT or PLM XML file where the data is to be stored. If not specified, the utility looks at the value of the **RDVContextDownloadDirectory** preference. Otherwise, the current working directory is used as the default path.

Note The **absolute_path** must be specified without an extension. For example, **/users/x_user/tempfile** for UNIX or **c:\temp\tempfile** for Windows.

-h

Displays help for this utility.

ENVIRONMENT

As specified in [Manually configuring your environment for Teamcenter utilities](#) and the **RDV_debug** environment variable. If this variable is set, the Teamcenter **syslog** file contains additional debugging information.

FILES

As specified in [Log files produced by Teamcenter](#).

RESTRICTIONS

You must have the necessary create or modify privileges to successfully run this utility. If you do not have the appropriate privileges, data is not imported and an error message is written to the log file.

EXAMPLES

- The following example creates a PLM XML file called **TL109375** and uses the **TC_config_rule_name** preference to determine the revision rule:

```
$TC_ROOT/bin/rdv_context_download -item_id=TL109375 -rev_id=004
```

- The following example creates a PLM XML file named **TL109375**, but enforces revision configuration using the **Beta or less w/pdi** revision rule:

The following code sample shows the sample AJT file created in this example.

```
#####
# DirectModel ASCII file - version 1.0
# Written by BOMWriterFormatAJT Teamcenter
Wednesday, 01/21/2004 05:56:39PM
0 ASM "AC7192-Product Book 2.asm;0;0:"
  ATTR Type="STRING" Key="DB_PART_NO" Value="AC7192"
  ATTR Type="STRING" Key="DB_PART_REV" Value="A"
  ATTR Type="STRING" Key="PLM_ITEMREV_UID" Value="Q1_o4OCyn6c4PB"
  ATTR Type="STRING" Key="DB_OCC_UID" Value="AAAAAAAAAAAAA"
  1 ASM "0728-045.asm;1885772752;-1684155971:"
    ATTR Type="STRING" Key="DB_PART_NO" Value="0728-045"
    ATTR Type="STRING" Key="DB_PART_REV" Value="A"
    ATTR Type="STRING" Key="PLM_ITEMREV_UID" Value="QX9o4OCyn6c4PB"
    ATTR Type="STRING" Key="DB_OCC_UID" Value="0z$o4OCyn6c4PB"
    Matrix [ 1.000 0.000 0.000 0.000 ]
            [ 0.000 0.998 -0.067 0.000 ]
            [ 0.000 0.067 0.998 0.000 ]
            [ -5.646 0.375 -0.150 1.000 ]
  2 PRT "1602-023.prt;-1968592985;273560760:"
2 PRT "1602-023.prt;-1968592985;-1738192227:"
2 PRT "1602-017.prt;1215126988;-365534238:"
2 PRT "1602-016.prt;-935130874;-619873730:"
2 PRT "1602-016.prt;-935130874;-650811331:"
1 PRT "H02976.prt;-1812813847;-640191115:"
1 PRT "E6895.prt;-1559507236;-2122085832:"
1 PRT "H02592.prt;1415580139;939946980:"
1 PRT "E6820.prt;-963798381;458164065:"
1 PRT "ERG056.prt;-1508967178;210332543:"
1 PRT "1606-177.prt;80810929;-396904688:"
# end

#####
&?>#####
```

Sample AJT file

rdv_migrate_architecture

Effective from Teamcenter 2007.1 MP7, Platform Designer allows you to revise architecture breakdowns and carry forward all architecture breakdown elements (ABEs), variability, named variant expressions (NVEs) and part solutions to the next revision. Architectures created in previous versions of Teamcenter are not compatible with the current format and must be updated.

Use this utility to update all existing ABEs to the new format. After migration, each ABE is associated with two sets of appearance path nodes (APNs) and each APN is linked to an **AbsOccData** object. One **AbsOccData** object has its immediate parent as the context and the other **AbsOccData** object has the top-level architecture of the architecture breakdown as its context. The suppression flag on the first **AbsOccData** object is set to **true**, and the second suppression flag is set to **false**.

Note Use the [rdv_migrate_part_solutions](#) utility to migrate the associated part solutions.

SYNTAX

```
rdv_migrate_architecture [-u=user-id {-p=password |  
-pf=password-file} -g=group]  
{-arch_ItemId<TOP-ARCH-ID> |  
[-arch_key=[keyAttr1=keyVal1] [,keyAttr2=keyVal2]...[,keyAttrN=keyValN]]}  
-arch_RevId<TOP-ARCH-REVISION-ID>  
[-h]
```

ARGUMENTS

-u

Specifies the user ID.

This is generally **infodba** or another user with administration privileges. If this argument is used without a value, the operating system user name is used.

Note If Security Services single sign-on (SSO) is enabled for your server, the **-u** and **-p** arguments are authenticated externally through SSO rather than being authenticated against the Teamcenter database. If you do not supply these arguments, the utility attempts to join an existing SSO session. If no session is found, you are prompted to enter a user ID and password.

-p

Specifies the password.

If used without a value, the system assumes a null value. This argument is mutually exclusive with the **-pf** argument.

If this argument is not used, the system assumes the *user-ID* value to be the password.

-pf

Specifies the password file. If used without a value, the system assumes a null value. If this argument is not used, the system assumes the *user-ID* value to be the password.

For more information about managing password files, see [Manage password files](#).

This argument is mutually exclusive with the **-p** argument.

-g

Specifies the group associated with the user.

If used without a value, the user's default group is assumed.

-arch_ItemId

Specifies the item ID of the top architecture element.

-arch_key

Specifies the keys of the item to track. Use the following format:

```
[keyAttr1=keyVal1] [,keyAttr2=keyVal2]...[,keyAttrN=keyValN]
```

-h

Displays help for this utility.

ENVIRONMENT

As specified in [Manually configuring your environment for Teamcenter utilities](#) and the **RDV_debug** environment variable. If this variable is set, the **syslog** file contains additional debugging information.

FILES

As specified in [Log files produced by Teamcenter](#).

RESTRICTIONS

Run this utility with a user ID that has sufficient permissions to create tasks and to write to the specified architecture breakdown and ABE objects in the database.

EXAMPLES

- The following example migrates a top architecture element with an item ID of **architecture001**:

```
rdv_migrate_architecture -u=infodba -p=infodba -g=dba -arch_ItemId=architecture001
```

rdv_migrate_part_solutions

Effective from Teamcenter 2007.1 MP7, Platform Designer allows you to revise architecture breakdowns and carry forward all architecture breakdown elements (ABEs), variability, named variant expressions (NVEs) and part solutions to the next revision. Part solutions created in previous versions of Teamcenter are not compatible with the current format and must be updated.

Use this utility to migrate existing part solution data to the current format. The utility takes the top architecture element of an existing breakdown and fetches all the lines of usage for each of the ABEs in that breakdown. It then creates a **LOUHOLDER** holder under the top line, if one does not already exist. It adds all the part solutions under this holder and associates them with the corresponding ABEs with **Appearance Group** relations.

Note Use the [rdv_migrate_architecture](#) utility to migrate the associated architecture.

SYNTAX

```
rdv_migrate_part_solutions [-u=user-id {-p=password |  
-pf=password-file} -g=group]  
{-arch_ItemId<TOP-ARCH-ID> |  
[-arch_key=[keyAttr1=keyVal1] [,keyAttr2=keyVal2]...[,keyAttrN=keyValN]]}  
-arch_RevId=<TOP-ARCH-REVISION-ID>  
[-h]
```

ARGUMENTS

-u

Specifies the user ID.

This is generally **infodba** or another user with administration privileges. If this argument is used without a value, the operating system user name is used.

Note If Security Services single sign-on (SSO) is enabled for your server, the **-u** and **-p** arguments are authenticated externally through SSO rather than being authenticated against the Teamcenter database. If you do not supply these arguments, the utility attempts to join an existing SSO session. If no session is found, you are prompted to enter a user ID and password.

-p

Specifies the password.

If used without a value, the system assumes a null value. This argument is mutually exclusive with the **-pf** argument.

If this argument is not used, the system assumes the *user-ID* value to be the password.

-pf

Specifies the password file. If used without a value, the system assumes a null value. If this argument is not used, the system assumes the *user-ID* value to be the password.

For more information about managing password files, see [Manage password files](#).

This argument is mutually exclusive with the **-p** argument.

-g

Specifies the group associated with the user.

If used without a value, the user's default group is assumed.

-arch_ItemId

Specifies the item ID of the top architecture element.

-arch_key

Specifies the architecture breakdown keys of the item to track. Use the following format:

```
[keyAttr1=keyVal1] [,keyAttr2=keyVal2]...[,keyAttrN=keyValN]
```

-arch_RevId

Specifies the revision identifier of the same top architecture element.

-h

Displays help for this utility.

ENVIRONMENT

As specified in [Manually configuring your environment for Teamcenter utilities](#) and the **RDV_debug** environment variable. If this variable is set, the **syslog** file contains additional debugging information.

FILES

As specified in [Log files produced by Teamcenter](#).

RESTRICTIONS

Run this utility with a user ID that has sufficient permissions to create tasks and to write to the specified objects in the database.

EXAMPLES

- The following example migrates the part solutions in a top architecture element with an item ID of **architecture001** and revision identifier of **architecture001Rev1**:

```
rdv_migrate_part_solutions -u=infodba -p=infodba -g=dba  
-arch_ItemId=architecture001 -arch_RevId=architecture001Rev1
```

rdv_run_audit_report

Generates audit reports that report the information related to mismatches that exist between VAS CAD structure and usage data that is imported into Teamcenter.

Audit reports allow you to check that design solutions exist for each option combination (NVE) required on the architecture elements. This feature assumes you use both part and design solutions; if you use only part solutions, do not use this report.

Note You create audit reports for design solutions in Structure Manager. Use Platform Designer to create audit reports for part solutions.

To create an audit report for design solutions, Teamcenter first performs a consistency check, verifying that for each design solution in the selected installation assembly, there is a part solution that matches the NVE on the design solution. It also checks that the variant condition on the design solution is consistent with the variant condition on the part solution. If the NVE changes on the architecture or part solution, the variant condition on the design solution may be out-of-date and require refreshing. Alternatively, the design solution itself may be changed to meet the new NVE requirement.

Designers perform this type of audit on the installation assembly for which they are responsible. You cannot perform this audit on a level higher than the installation assembly.

You can still use the audit report if no part solutions exist yet, because the NVEs on the architecture are checked. You can create an audit report for a particular configuration by setting the variant and revision rules audit algorithm details. The audit checks the following:

- If the NVE is not referenced by any solution.
- If the NVE is referenced by one or more solutions with a matching architecture element ID, but the variant condition is out-of-date (assuming there are no split NVEs).
- If the usage quantity is not a positive integer.
- If the solutions are referencing an NVE, but the architecture element ID references a split NVE.
- If the NVE is referenced by one or more solutions, but none of them have an architecture element ID that matches that of the NVE. The audit report delivers a list of architecture breakdowns in rows, with colored indicators specifying whether the line is an exact, partial, or mismatch with respect to an NVE, part number (typically the part solution ID), and usage quantity.

The audit algorithms use data stored in occurrence notes on the design solution. It is therefore necessary to replace a design in the product using the Replace Design in Product wizard before the audit can be run to populate the appropriate occurrence notes. These occurrence notes are:

Usage_Product
Usage_PartNumber

Usage_Quantity

Configure these occurrence notes by setting the [RDV_copied_occurrence_notes](#) preference.

SYNTAX

```
rdv_run_audit_report [-u=user-id {-p=password | -pf=password-file} -g=group]  
[-product_id=item-id-of-top-level-product | -product_key=key-id-of-top-level-product]  
-product_rev_id=revision-id  
-v_rule=saved-variant-rule  
-rev_rule=revision-rule-name  
[-input_file=input-file-path | -keys_input_file=keys-input-file-path]  
-h
```

ARGUMENTS

-u

Specifies the user ID.

This is generally **infodba** or another privileged user. If this argument is used without a value, the operating system user name is used.

Note

If Security Services single sign-on (SSO) is enabled for your server, the **-u** and **-p** arguments are authenticated externally through SSO rather than being authenticated against the Teamcenter database. If you do not supply these arguments, the utility attempts to join an existing SSO session. If no session is found, you are prompted to enter a user ID and password.

-p

Specifies the password.

If used without a value, the system assumes a null value. This argument is mutually exclusive with the **-pf** argument.

If this argument is not used, the system assumes the *user-ID* value to be the password.

-pf

Specifies the password file. If used without a value, the system assumes a null value. If this argument is not used, the system assumes the *user-ID* value to be the password.

For more information about managing password files, see [Manage password files](#).

This argument is mutually exclusive with the **-p** argument.

-g

Specifies the group associated with the user.

If used without a value, the user's default group is assumed.

-product_id

Specifies the item ID of the top-level product.

-product_key

Specifies the key of the top-level product. This argument can be used to identify an item instead of the **-product_id** argument.

To find the key of an object, use the [get_key_string](#) utility.

For more information, see the [Business Modeler IDE Guide](#).

-product_rev_id

Specifies the item ID of the top-level product revision.

-v_rule

Specifies the saved variant rule.

-rev_rule

Specifies a named revision rule. Defaults to the site default, frequently the **latest working** revision.

-input_file

Specifies the path to the file used for input when you use the **-product_id** argument.

-keys_input_file

Specifies the path to the file used for input when you use the **-product_key** argument.

-h

Displays help for this utility.

ENVIRONMENT

As specified in [Manually configuring your environment for Teamcenter utilities](#).

FILES

As specified in [Log files produced by Teamcenter](#).

RESTRICTIONS

None.

rdv_set_default_variant_condition

Checks whether the input item has any qualified occurrences attached to it and sets a default variant condition on them so that the occurrence can participate in variant configuration. This utility configures the structure using the revision rule provided as input. If the revision rule is not specified, it uses the default revision rule.

This utility also sets the variant condition specified as input. If the variant condition is not specified then it uses the variant condition specified in the **RDV_default_variant_condition** preference.

SYNTAX

```
rdv_set_default_variant_condition [-u=user-id {-p=password | -pf=password-file}
-g=group]
[-item=item-ID | -key=key-ID]
[-revision_rule=revision-rule-name]
-variant_condition=condition-name
-delimiter=delimiter-string
-qualified_occu_scan_depth=depth
-h
```

ARGUMENTS

-u

Specifies the user ID.

This is generally **infodba** or another privileged user. If this argument is used without a value, the operating system user name is used.

Note

If Security Services single sign-on (SSO) is enabled for your server, the **-u** and **-p** arguments are authenticated externally through SSO rather than being authenticated against the Teamcenter database. If you do not supply these arguments, the utility attempts to join an existing SSO session. If no session is found, you are prompted to enter a user ID and password.

-p

Specifies the password.

If used without a value, the system assumes a null value. This argument is mutually exclusive with the **-pf** argument.

If this argument is not used, the system assumes the *user-ID* value to be the password.

-pf

Specifies the password file. If used without a value, the system assumes a null value. If this argument is not used, the system assumes the *user-ID* value to be the password.

For more information about managing password files, see [Manage password files](#).

This argument is mutually exclusive with the **-p** argument.

-g

Specifies the group associated with the user.

If used without a value, the user's default group is assumed.

-item

Specifies the ID of the item to be used for the variant condition.

-key

Specifies a string used to identify an item instead of specifying an item using the **-item** argument.

To find the key of an object, use the [get_key_string](#) utility.

For more information, see the [Business Modeler IDE Guide](#).

-revision_rule

Specifies a named revision rule. Defaults to the site default, frequently the **latest working** revision.

-variant_condition (Optional)

Specifies the variant condition.

-delimiter (Optional)

Specifies the delimiter to place between records.

-qualified_occu_scan_depth (Optional)

Specifies the level to scan occurrences.

-h

Displays help for this utility.

ENVIRONMENT

As specified in [Manually configuring your environment for Teamcenter utilities](#).

FILES

As specified in [Log files produced by Teamcenter](#).

RESTRICTIONS

None.

EXAMPLES

- The following example configures the structure using default revision rule, scans the item for null variant components, and sets the default variant condition specified in preference files:

```
rdv_set_default_variant_condition -item_id=TL109375
```

- The following example configures the structure using the latest working revision rule, scans the item for null variant components, and sets the default variant condition specified in preference files:

```
rdv_set_default_variant_condition -item_id=TL109375
-revision_rule=Latest Working
```

- The following example configures the structure using the latest working revision rule, scans the item for null variant components, and sets the variant data (**Color=Red**) on null variant components:

```
rdv_set_default_variant_condition -item_id=TL109375
-revision_rule=Latest Working -variant_condition=002763:Color:Red
```

start_sco_dispatcher

Starts the structure context object (SCO) dispatcher to initiate the transfer of SCOs created by the [rdv_context_download](#) utility to other sites. The [validate_and_replicate_assembly](#) utility subsequently consumes the PLM XML files created by the [rdv_context_download](#) utility and replicates them across sites.

SYNTAX

```
start_sco_dispatcher [-u=user-id {-p=password | -pf=password-file} -g=group]
-saved_query_name=Teamcenter-saved-query-name |
-sco_folder=Teamcenter-folder-name |
-sco_list=list-of-scOs | -sco_name=StructureContextObjectName
-site_name=Destination-site-name | -site_url=Destination-site-url
[-transfer_mode=transfer-mode-name] [-dataset_time_interval=time-interval]
[-user_attrs=user-attributes] [-h]
```

ARGUMENTS**-u**

Specifies the user ID.

This is generally **infodba** or another user with administration privileges. If this argument is used without a value, the operating system user name is used.

Note

If Security Services single sign-on (SSO) is enabled for your server, the **-u** and **-p** arguments are authenticated externally through SSO rather than being authenticated against the Teamcenter database. If you do not supply these arguments, the utility attempts to join an existing SSO session. If no session is found, you are prompted to enter a user ID and password.

-p

Specifies the password.

If used without a value, the system assumes a null value. This argument is mutually exclusive with the **-pf** argument.

If this argument is not used, the system assumes the *user-ID* value to be the password.

-pf

Specifies the password file. If used without a value, the system assumes a null value. If this argument is not used, the system assumes the *user-ID* value to be the password.

For more information about managing password files, see [Manage password files](#).

This argument is mutually exclusive with the **-p** argument.

-g

Specifies the group associated with the user.

If used without a value, the user's default group is assumed.

-sco_name

Specifies the name of the SCO from which a PLM XML file is generated. You can specify more than one SCO in a comma-separated list, for example:

SCO1,SCO2,SCO3

-sco_list

Specifies a list of SCOs to process. You can specify more than one SCO in a hash-separated list, for example:

SCO1#SCO2#SCO3

-saved_query_name

Specifies the name of a Teamcenter saved query that retrieves a set of SCOs to process for given search criteria.

-sco_folder

Specifies the name of a Teamcenter folder that contains a set of SCOs to process.

-transfer_mode

Specifies the name of a transfer mode to generate the PLM XML file. If no mode is specified, the SCOs are evaluated with the default transfer mode.

-site_name

Specifies the names of the sites with databases to which Teamcenter sends the SCOs. You can specify more than one site in a hash-separated list, for example:

Site1#Site2#Site3

-site_url

Specifies the URLs of the sites to which Teamcenter sends the SCOs. Use URLs, rather than site names, if the target sites do not have databases. You can specify more than one URL in a hash-separated list, for example:

Site1#Site2#Site3

-dataset_time_interval

Specifies the interval in minutes at which the **rdvcontextdownload** translator generates PLM XML files. If no interval is specified, the default value of 60 minutes is used.

-user_attribute

Specify user attributes to include in the PLM XML file. The specified attributes must be BOM line properties. You can specify more than one user attribute as a hash-separated list, for example:

Owningsitename#ItemID#RevisionID##last_mod_date

-h

Displays help for this utility.

ENVIRONMENT

As specified in [Manually configuring your environment for Teamcenter utilities](#).

FILES

As specified in [Log files produced by Teamcenter](#).

RESTRICTIONS

- You must have the necessary create or modify data privileges to successfully run this utility. If you do not have the appropriate privileges, data is not imported and an error message is written to the log file.

- If the utility evaluates an SCO and identifies that the target BOM lines are not valid, it exits and displays an error message to the user. It does not process any further SCOs.

EXAMPLES

- The following example runs a saved query called **SCO_SQR** to transfer SCOs to a single database site using Multi-Site Collaboration:

```
start_sco_dispatcher -saved_query_name=SCO_SQR -site_name=testsite  
-dataset_time_interval=120
```

- The following example transfers all SCOs in the **SCO_folder** folder to a single site without a database. The transfer is achieved via FMS by specifying the target site URL:

```
start_sco_dispatcher -folder_name=SCO_folder -transfer_mode=PIEPLMXMLDEFAULT  
-site-url=http://testsite:7001
```

sync_product_apns

Enables a site to synchronize product appearance path nodes (APNs) with other sites. The utility also synchronizes the absolute occurrence data that is associated with these APNs. In addition, it also synchronizes item revision attachments with the **RDV_appgrp_toplevel_relation** relationship type. In RDV, this attachment type associates part usage occurrence groups with an architecture revision. Part usage occurrence groups are workspace objects that collect one architecture element (APN) and all corresponding part usage occurrences (APNs). A part usage references its architecture element by means of this part usage occurrence group. Because **RDV_appgrp_toplevel_relation** attachments are handled with this utility, Siemens PLM Software recommends you exclude this relationship when replicating the architecture revision with the **data_sync** or **data_share** utilities.

This utility is similar to the **sync_product_variant_data** utility in that it synchronizes occurrence data, occurrence roots, APNs etc. while the **sync_product_variant_data** utility synchronizes variant data, such as variant objects, variant revision objects, item revision expressions, and NVEs.

This utility operates in the following modes and each mode has different mandatory and optional arguments:

- **Export to disk**

Use of the **-item** argument indicates that the utility runs in **export** mode. Note that either **-dir=directory-name** or **-count** are required as indicated with the { | } notation.

- **Send to remote IDSM**

Use of the **-dir** and **-site** arguments indicate that the utility runs in **send** mode.

- **Read/import from disk**

Use of the **-dir** argument without the **-site** argument indicates that the utility runs in **read/import** mode. Note that the utility runs in **read-only** mode if the **-browse** argument is specified, otherwise data is imported to the local site.

SYNTAX

- **Export to disk**

```
sync_product_apns[-u=user-id {-p=password |
-pf=password-file} -g=group]
[-item=item-id | -key=key-id] [-arch] [-bypass]
-site=site-name [-reason=text]
{-dir=directory-name [-tag_list=file-name [-from=number] [-to=number] ]
| -count[-tag_list=file-name] }
[-modified_only] [-debug]
[-exclude_variants]
[-silent] [-rollback_on_failure]
```

- **Send to remote IDSM**

```
sync_product_apns[-u=user-id {-p=password |
-pf=password-file} -g=group]
-dir=directory-name -site=site-name
[-from=number] [-to=number]
[-silent] [-rollback_on_failure]
```

- **Read / import from disk**

```
sync_product_apns[-u=user-id {-p=password |
-pf=password-file} -g=group]
-dir=directory-name [-browse] [-bypass]
[-from=number] [-to=number]
[-silent] [-rollback_on_failure]
```

ARGUMENTS

-dir

Specifies a local directory name. The utility writes exported data into this directory. If the directory exists, it must not contain export data.

For **export** mode, the directory may or may not exist. If the directory exists, it must not contain exported data, for example, it must not have been used as an export directory in a previous run. For **send** or **read/import** modes, the directory must exist and must contain export data.

-item

Item ID of the item of which appearance and absolute occurrence data is to be exported.

-key

Specifies the key of the object. The **-key** argument can be used instead of the **-item** argument.

To find the key of an object, use the [get_key_string](#) utility.

For more information, see the [Business Modeler IDE Guide](#).

-arch

Exports appearance and absolute occurrence data of the architecture items that are associated with the item specified with the **-item** argument.

-site

This argument can only be used with the **export** or **send** modes; it cannot be used in **read/browse** mode.

- **Export mode**

Specifies the sites to which objects are to be replicated. Upon exporting an object, the utility creates one export record for each site if this object type supports export records. Export records contain the time of export to support incremental replication. If the user requests a roll back at the end of the utility, all export records that were generated during export are rolled back.

- **Send mode**

Specifies the sites to which objects are to be sent. This mode requires an IDSM service to be available at the remote site. If multiple **-site** switches are used to specify multiple sites, the export directory is sent to each site sequentially. For better throughput, run separate processes for each site instead of running one process for multiple sites.

-count

Reports the number of objects to export. If used with the **-modified_only** argument, the count indicates the number of changed objects that need to be replicated to at least one of the sites specified with the **-site** argument. If used with the **-tag_list** argument, a tag list file is written.

-exclude_variants

Causes the export to exclude objects of either the **Variant** or **VariantRevision** class that are attached to the item specified with the **-item** argument. This argument is useful when exporting in multiple batches. If this option is not set, each batch contains all variant options referenced from its expressions causing multiple batches to contain an overlapping set of objects. Overlapping sets of objects cannot be imported in parallel.

-apn_roots_only

Specifies to only export appearance path node roots and absolute occurrence data qualifiers. If this argument is not specified, these objects are excluded.

-silent

Disables rollback capability and rollback confirmation dialog and executes in silent mode.

-rollback_on_failure

Automatically rolls back on any error code other than **ITK_ok**.

-debug

Turns on verbose mode that generates a detailed export and modified dates.

-bypass

Disables access control. You must have DBA privileges to use this argument.

-modified_only

Specifies to only export objects that have changed since they were exported to the sites specified by the **-site** argument.

Note If more than one site is specified, the object is exported according to the most out-of-date export record. This can cause some objects to be exported to some sites even though these sites appear to have an up-to-date export record. After completing the export, all sites specified with the **-site** argument have the same export date for each exported object.

-reason

Specifies the reason for export. The maximum number of characters is 240. If no reason is specified, the command line of running the export utility is used as the *reason* text.

-from

Specifies the index of the first object to be exported. The default value is zero.

Use this argument for **export**, **send**, and **read/import** modes. In **export** and **send** mode, the index refers to the tag list file specified with the **-tag_list** argument. Otherwise, it refers to the object list in the export data located in the directory specified with the **-dir** argument.

-to

Specifies the index of the object *following* the last object to process. For example, if you specify **-to=10**, the object at index 9 is the last object to be processed.

Use this argument for **export**, **send**, and **read/import** modes. In **export** and **send** mode, the index refers to the tag list file specified with the **-tag_list** argument. Otherwise, it refers to the object list in the export data located in the directory specified with the **-dir** argument.

-tag_list

Specifies a file name containing a list of object UIDs to export. This file is written if the **-count** argument is specified. If the **-count** argument is not specified, the file is read.

-browse

Specifies to browse objects in the export area without exporting or importing.

-u

Specifies the user ID.

This is a required argument unless the **TC_auto_login** site preference is set.

This is generally **infodba** or another user with administration privileges. If this argument is used without a value, the operating system user name is used.

Note If Security Services single sign-on (SSO) is enabled for your server, the **-u** and **-p** arguments are authenticated externally through SSO rather than being authenticated against the Teamcenter database. If you do not supply these arguments, the utility attempts to join an existing SSO session. If no session is found, you are prompted to enter a user ID and password.

-p

Specifies the password.

This is a required argument unless the **TC_auto_login** site preference is set.

If used without a value, the system assumes a null value. This argument is mutually exclusive with the **-pf** argument.

If this argument is not used, the system assumes the *user-ID* value to be the password.

-pf

Specifies the password file. If used without a value, the system assumes a null value. If this argument is not used, the system assumes the *user-ID* value to be the password.

For more information about managing password files, see [Manage password files](#).

This argument is mutually exclusive with the **-p** argument.

-g

Specifies the group associated with the user.

This is a required argument unless the **TC_auto_login** site preference is set.

If used without a value, the user's default group is assumed.

-h

Displays help for this utility.

ENVIRONMENT

As specified in [Manually configuring your environment for Teamcenter utilities](#).

To bypass the export of named variant expressions, set the **RDV_export_nve** environment variable to **FALSE**.

When using the **data_share** and **data_sync** utilities, Siemens PLM Software recommends you exclude APN and NVE synchronization by setting the following environment variables:

- **TC_EXCLUDE_APN=TRUE**
- **RDV_export_nve=FALSE**

FILES

As specified in [Log files produced by Teamcenter](#).

RESTRICTIONS

None.

EXAMPLES

1. Export to disk all appearance path nodes, absolute occurrences, and absolute occurrence data on item **RDV00190**, excluding the item itself:

```
sync_product_apns -u=infodba -p=*** -g=dba
-bypass -dir=C:\Temp\RDV00190_apns -item=RDV00190
-site=cologne -site=turin
```

2. Export to disk all appearance path nodes, absolute occurrences, and absolute occurrence data on item **RDV00190**, along with data of associated architecture breakdowns:

```
sync_product_apns -u=infodba -p=*** -g=dba
```

```
-bypass -dir=C:\Temp\RDV00190_apns -item=RDV00190 -arch
-site=cologne -site=turin
```

3. Export to disk all modified appearance path nodes, absolute occurrences, and absolute occurrence data on item **RDV00190**, and its architecture breakdowns:

```
sync_product_apns -u=infodba -p=*** -g=dba
-bypass -dir=C:\Temp\RDV00190_apns -item=RDV00190 -arch
-modified_only -site=cologne -site=turin
```

4. Determine objects for an incremental update of **RDV00190/D** to sites Cologne and Turin. This includes appearance path nodes, absolute occurrences, and absolute occurrence data for all revisions of **RDV00190** and all revisions of associated architecture breakdown items. Object UIDs are written to the **uid.txt** text file for later export.

```
sync_product_apns -u=infodba -p=*** -g=dba
-bypass -dir=C:\Temp\RDV00190_apns -item=RDV00190 -arch
-count -modified_only -site=cologne -site=turin -tag_list=uid.txt
```

5. Export to disk object index numbers 1,000 through 1,999 of all appearance path nodes, absolute occurrences, and absolute occurrence data of item **RDV00190**, along with appearance path nodes, absolute occurrences, and absolute occurrence data of associated architecture breakdowns. Objects are exported as determined in example 4.

The **-exclude_variants** argument causes variant and variant revision objects associated with item **RDV00190** to be excluded. This allows exporting variant data with the **sync_product_variant_data** utility in parallel.

```
sync_product_apns -u=infodba -p=*** -g=dba
-bypass -dir=C:\Temp\RDV00190_apns -item=RDV00190 -arch
-site=cologne -site=turin -from=1000 -to=2000
-exclude_variants -tag_list=uid.txt
```

6. Export to disk object index numbers 1,000 through the end of all appearance path nodes, absolute occurrences, and absolute occurrence data of item **RDV00190**, along with appearance path nodes, absolute occurrences, and absolute occurrence data of associated architecture breakdowns:

```
sync_product_apns -u=infodba -p=*** -g=dba
-bypass -dir=C:\Temp\RDV00190_apns -item=RDV00190 -arch
-site=cologne -site=turin -from=1000 -exclude_variants
```

7. Browse export area:

```
sync_product_apns -u=infodba -p=*** -g=dba
-dir=C:\Temp\RDV00190_apns -browse
```

8. Send export area to Turin's IDSM:

```
sync_product_apns -u=infodba -p=*** -g=dba
-dir=C:\Temp\RDV00190_apns -site=turin
```

9. Send object index numbers 1,000 through 1,999 in export area to Turin:

```
sync_product_apns -u=infodba -p=*** -g=dba
-dir=C:\Temp\RDV00190_apns -site=turin -from=1000 -to=2000
```

10. Browse import area:

```
sync_product_variant_data -u=infodba -p=*** -g=dba
-dir=C:\Temp\RDV00190_apns -browse
```

11. Import objects in import area:

```
sync_product_variant_data -u=infodba -p=*** -g=dba  
-dir=C:\Temp\RDV00190_variant_data
```

12. Import object index numbers 1,000 through 1,999 in import area:

```
sync_product_variant_data -u=infodba -p=*** -g=dba  
-dir=C:\Temp\RDV00190_apns-from=1000 -to=2000
```

sync_product_variant_data

Enables sites to synchronize product variant data with other sites.

This utility is similar to the **sync_product_apns** utility in that it synchronizes variant data, such as variant objects, variant revision objects, item revision expressions, and NVEs while the **sync_product_apns** utility synchronizes occurrence data, occurrence roots, APNs, and so forth.

This utility operates in the following modes and each mode has different mandatory and optional arguments:

- **Export to disk**

Using the **-item** argument indicates that the utility runs in **export** mode. Note that either **-dir=directory-name** or **-count** are required as indicated with the {} notation.

- **Send to remote IDSM**

Using the **-dir** and **-site** arguments indicate that the utility runs in **send** mode.

- **Read / import from disk**

Using the **-dir** argument without the **-site** argument indicates that the utility runs in **read/import** mode. The utility runs in **read-only** mode if the **-browse** argument is specified; otherwise, data is imported to the local site

SYNTAX

- **Export to disk**

```
sync_product_variant_data
[-u=user-id {-p=password | -pf=password-file} -g=group]
[-item=item-id | -key=key-id] [-rev=revision-id] [-arch] [-bypass]
-site=site-name [-reason=text]
{-dir=directory-name [-tag_list=file-name [-from=number] [-to=number] ]
| -count[-tag_list=file-name] }
[-modified_only] [-debug]
[-exclude_variants]
[-silent] [-rollback_on_failure]
```

- **Send to remote IDSM**

```
sync_product_variant_data
[-u=user-id -p=password | -pf=password-file -g=group]-dir=directory-name
-site=site-name
[-from=number] [-to=number]
[-silent] [-rollback_on_failure]
```

- **Read / import from disk**

```
sync_product_variant_data
[-u=user-id {-p=password | -pf=password-file} -g=group]
-dir=directory-name [-browse] [-bypass]
[-from=number] [-to=number]
```


[-silent] [-rollback_on_failure]

ARGUMENTS

-u

Specifies the user ID.

This is generally **infodba** or another user with administration privileges. If this argument is used without a value, the operating system user name is used.

Note

If Security Services single sign-on (SSO) is enabled for your server, the **-u** and **-p** arguments are authenticated externally through SSO rather than being authenticated against the Teamcenter database. If you do not supply these arguments, the utility attempts to join an existing SSO session. If no session is found, you are prompted to enter a user ID and password.

-p

Specifies the password.

If used without a value, the system assumes a null value. This argument is mutually exclusive with the **-pf** argument.

If this argument is not used, the system assumes the *user-ID* value to be the password.

-pf

Specifies the password file. If used without a value, the system assumes a null value. If this argument is not used, the system assumes the *user-ID* value to be the password.

For more information about managing password files, see [Manage password files](#).

This argument is mutually exclusive with the **-p** argument.

-g

Specifies the group associated with the user.

If used without a value, the user's default group is assumed.

-dir

Specifies a local directory name. The utility writes exported area into this directory. If the directory exists, it must not contain export data.

For **export** mode, the directory may or may not exist. If the directory exists, it must not contain exported data, for example, it must not have been used as an export directory in a previous run. For **send** or **read/import** modes, the directory must exist and must contain export data.

-item

Specifies the item ID of the item of which variant data is to be exported.

-key

Specifies the key of the object. The **-key** argument can be used instead of the **-item** argument.

To find the key of an object, use the [get_key_string](#) utility.

For more information, see the [Business Modeler IDE Guide](#).

-rev

Specifies the revision ID of the item revision of which variant data is to be exported in addition to the variant data of its item.

-arch

Exports variant data of the architecture items and architecture revisions that are associated with the item specified with the **-item** argument and the revision specified with the **-rev** argument.

-site

This argument can only be used with the **export** or **send** modes; it cannot be used in **read/browse** mode.

- **export mode**

Specifies the sites to which objects are to be replicated. Upon exporting an object, the utility creates one export record for each site if this object type supports export records. Export records contain the time of export to support incremental replication. If the user requests a rollback at the end of the utility, all export records that were generated during export are rolled back.

- **send mode**

Specifies the sites to which objects are to be sent. This mode requires an IDSM service to be available at the remote site. If multiple **-site** switches are used to specify multiple sites, the export directory is sent to each site sequentially. For better throughput, run separate processes for each site instead of running one process for multiple sites.

-count

Reports the number of objects to export. If used with the **-modified_only** argument, the count indicates the number of changed objects that need to be replicated to at least one of the sites specified with the **-site** argument. If used with the **-tag_list** argument, a tag list file is written.

-exclude_variants

Causes the export to exclude objects of either the **Variant** or **VariantRevision** class that are attached to the item specified with the **-item** argument. This argument is useful when exporting in multiple batches. If this option is not set, each batch contains all variant options referenced from its expressions causing multiple batches to contain an overlapping set of objects. Overlapping sets of objects cannot be imported in parallel.

-silent

Disables rollback capability and rollback confirmation dialog and executes in silent mode.

-rollback_on_failure

Automatically rolls back on any error code other than **ITK_ok**.

-debug

Turns on verbose mode, which generates a detailed report of export and modified dates.

-bypass

Disables access control. You must have DBA privileges to use this argument.

-modified_only

Specifies to only export objects that have changed since they were exported to the sites specified by the **-site** argument.

Note If more than one site is specified, the object is exported according to the most out-of-date export record. This can cause some objects to be exported to some sites even though these sites appear to have an up-to-date export record. After completing the export, all sites specified with the **-site** argument have the same export date for each exported object.

-reason

Specifies the reason for export. The maximum number of characters is 240. If no reason is specified, the command line of running the export utility is used as the *reason* text.

-from

Specifies the index of the first object to be exported. The default value is zero.

Use this argument for **export**, **send**, and **read/import** modes. In **export** and **send** mode, the index refers to the tag list file specified with the **-tag_list** argument. Otherwise, it refers to the object list in the export data located in the directory specified with the **-dir** argument.

-to

Specifies the index of the object *following* the last object to process. For example if you specify **-to=10**, the object at index 9 is the last object to be processed.

Use this argument for **export**, **send**, and **read/import** modes. In **export** and **send** mode, the index refers to the tag list file specified with the **-tag_list** argument. Otherwise, it refers to the object list in the export data located in the directory specified with the **-dir** argument.

-tag_list

Specifies a file name containing a list of object UIDs to export. This file is written if the **-count** argument is specified. If the **-count** argument is not specified, the file is read.

-count

Reports the number of objects to export. If used with the **-modified_only** argument, the count indicates the number of changed objects that need to be replicated to at least one of the sites specified with the **-site** argument. If used with the **-tag_list** argument, a tag list file is written.

-browse

Specifies to browse objects in the export area without exporting or importing.

-u

Specifies the user ID.

This is generally **infodba** or another user with administration privileges. If this argument is used without a value, the operating system user name is used.

Note If Security Services single sign-on (SSO) is enabled for your server, the **-u** and **-p** arguments are authenticated externally through SSO rather than being authenticated against the Teamcenter database. If you do not supply these arguments, the utility attempts to join an existing SSO session. If no session is found, you are prompted to enter a user ID and password.

-p

Specifies the password.

If used without a value, the system assumes a null value. This argument is mutually exclusive with the **-pf** argument.

If this argument is not used, the system assumes the *user-ID* value to be the password.

-pf

Specifies the password file. If used without a value, the system assumes a null value. If this argument is not used, the system assumes the *user-ID* value to be the password.

For more information about managing password files, see [Manage password files](#).

This argument is mutually exclusive with the **-p** argument.

-g

Specifies the group associated with the user.

If used without a value, the user's default group is assumed.

-h

Displays help for this utility.

ENVIRONMENT

As specified in [Manually configuring your environment for Teamcenter utilities](#).

In addition, to export named variant expressions (NVEs), you must set the **RDV_export_nve** preference to **TRUE**. To bypass the export of NVEs, set the **RDV_export_nve** preference to **FALSE**.

FILES

As specified in [Log files produced by Teamcenter](#).

RESTRICTIONS

None.

EXAMPLES

1. Export to disk all variant options and variant revisions of item **RDV00190**, excluding the item itself:

```
sync_product_variant_data -u=infodba -p=*** -g=dba
-bypass -dir=C:\Temp\RDV00190_variant_data -item=RDV00190
-site=cologne -site=turin
```

2. Export to disk all variant options and variant revisions of item **RDV00190** with variant data of associated architecture breakdowns (mostly NVEs):

```
sync_product_variant_data -u=infodba -p=*** -g=dba
-bypass -dir=C:\Temp\RDV00190_variant_data -item=RDV00190
-arch -site=cologne -site=turin
```

3. Export to disk all variant options and variant revisions of item **RDV00190** and all modified NVEs on its architecture breakdowns:

```
sync_product_variant_data -u=infodba -p=*** -g=dba
-bypass -dir=C:\Temp\RDV00190_variant_data -item=RDV00190
-arch -modified_only -site=cologne -site=turin
```

4. Determine objects for an incremental update of **RDV00190/D** to sites Cologne and Turin. This includes variant expressions of **RDV00190/D** and its associated architecture breakdown revisions (mostly **DECLARE** and **DEFAULT** statements) and the NVEs of associated architecture breakdown items. Object UIDs are written to the **uid.txt** text file for later export.

```
sync_product_variant_data -u=infodba -p=*** -g=dba
-bypass -dir=C:\Temp\RDV00190_variant_data -item=RDV00190 -rev=D
-arch -count -modified_only -site=cologne -site=turin
-exclude_variants -tag_list=uid.txt
```

5. Export to disk all variant options of item **RDV00190** and all variant expressions (mostly **DECLARE** and **DEFAULT** statements) on revision D:

```
sync_product_variant_data -u=infodba -p=*** -g=dba
-bypass -dir=C:\Temp\RDV00190_variant_data -item=RDV00190
-rev=D -site=cologne -site=turin
```

6. Export to disk all variant options of item **RDV00190** and all variant expressions (mostly **DECLARE** and **DEFAULT** statements) on revision D, along with variant data of associated architecture breakdowns (mostly NVEs), and associated architecture breakdown revisions (mostly **DECLARE** and **DEFAULT** statements):

```
sync_product_variant_data -u=infodba -p=*** -g=dba
-bypass -dir=C:\Temp\RDV00190_variant_data -item=RDV00190
-rev=D -arch -site=cologne -site=turin
```

7. Export to disk object index numbers 1,000 through 1,999 of all variant expressions (mostly **DECLARE** and **DEFAULT** statements) of revision D of item **RDV00190**, along with variant data of associated architecture breakdowns (mostly NVEs), associated architecture breakdown revisions (mostly **DECLARE** and **DEFAULT** statements), and their attached saved variant rules. Objects are exported as determined in example 4.

To allow exporting and importing multiple batches in parallel it is recommended to export and import data as described in example 1, before batches as described in this example are exported and imported. The **-exclude_variants** command line option causes the objects exported in example 1 to be excluded. Otherwise, these variants would be exported into each export batch and upon importing a conflict may arise when importing the same variant object via parallel IDSM processes at the same time.

```
sync_product_variant_data -u=infodba -p=*** -g=dba
-bypass -dir=C:\Temp\RDV00190_variant_data -item=RDV00190 -rev=D
-arch -site=cologne -site=turin -from=1000 -to=2000
-exclude_variants -tag_list=uid.txt
```

8. Export to disk object index numbers 1,000 through the end of all variant expressions (mostly **DECLARE** and **DEFAULT** statements) on revision D of item **RDV00190**, along with variant data of associated architecture breakdowns (mostly NVEs), associated architecture breakdown revisions (mostly **DECLARE** and **DEFAULT** statements), and their attached saved variant rules:

```
sync_product_variant_data -u=infodba -p=*** -g=dba
-bypass -dir=C:\Temp\RDV00190_variant_data -item=RDV00190 -rev=D
-arch -site=cologne -site=turin -from=1000 -exclude_variants
```

9. Browse export area:

```
sync_product_variant_data -u=infodba -p=*** -g=dba
-dir=C:\Temp\RDV00190_variant_data -browse
```

10. Send export area to Turin's IDSM:

```
sync_product_variant_data -u=infodba -p=*** -g=dba
-dir=C:\Temp\RDV00190_variant_data -site=turin
```

11. Send object index numbers 1,000 through 1,999 in export area to Turin:

```
sync_product_variant_data -u=infodba -p=*** -g=dba
-dir=C:\Temp\RDV00190_variant_data -site=turin -from=1000 -to=2000
```

12. Browse import area:

```
sync_product_variant_data -u=infodba -p=*** -g=dba
-dir=C:\Temp\RDV00190_variant_data -browse
```

13. Import objects in import area:

```
sync_product_variant_data -u=infodba -p=*** -g=dba
-dir=C:\Temp\RDV00190_variant_data
```

14. Import object index numbers 1,000 through 1,999 in import area:

```
sync_product_variant_data -u=infodba -p=*** -g=dba
-dir=C:\Temp\RDV00190_variant_data -from=1000 -to=2000
```

RDV cache maintenance

get_qpl_harvester_assemblies

Generates an XML file with the list of installation assemblies or special assemblies from the BOM structure of the assembly. This utility is used in the RDV cache maintenance process and its main purpose is to extract information about the assemblies that the QPL harvester must process.

OVERVIEW

The utility also finds new, modified, or deleted installation assemblies or special assemblies by comparing the newly generated XML file with a previously generated XML file. The following information about each node is recorded in the XML file:

- **Item_id**
- **Item_name**
- **item_rev**
- **released_date**
- **abs_transformation_matrix**
- **occurrence_path**
- **unique_id**
- **owning_site**

This information is used for two purposes:

- Comparing to the previously generated XML file to find installation assemblies or special assemblies that have been changed, released, deleted from, or added to the structure.
- Providing installation assembly information required by the **ug_spacemap** utility, such as the transformation matrix (absolute xform matrix), the occurrence path (**subfile_id** path that must be appended to the occurrence path of each part occurrence being processed in the installation assembly), and the **unique_id** that is used in conjunction with the **subfile_id** of the **part_occ** to arrive at a uniquely identified number for this **part_occ** in the context of the top-level assembly.

The following input is required for the **get_qpl_harvester_assemblies** utility:

- Item ID, revision, revision rule, or a bookmark file (optional)
- Level at which the special assemblies can be identified
- Branches in which the special assemblies must be identified
- Names of the modules

The cache administrator must specify the level in such a way that the assemblies selected at that level belong to the product assembly structure site. The administrator can specify different levels for different branches, for example:

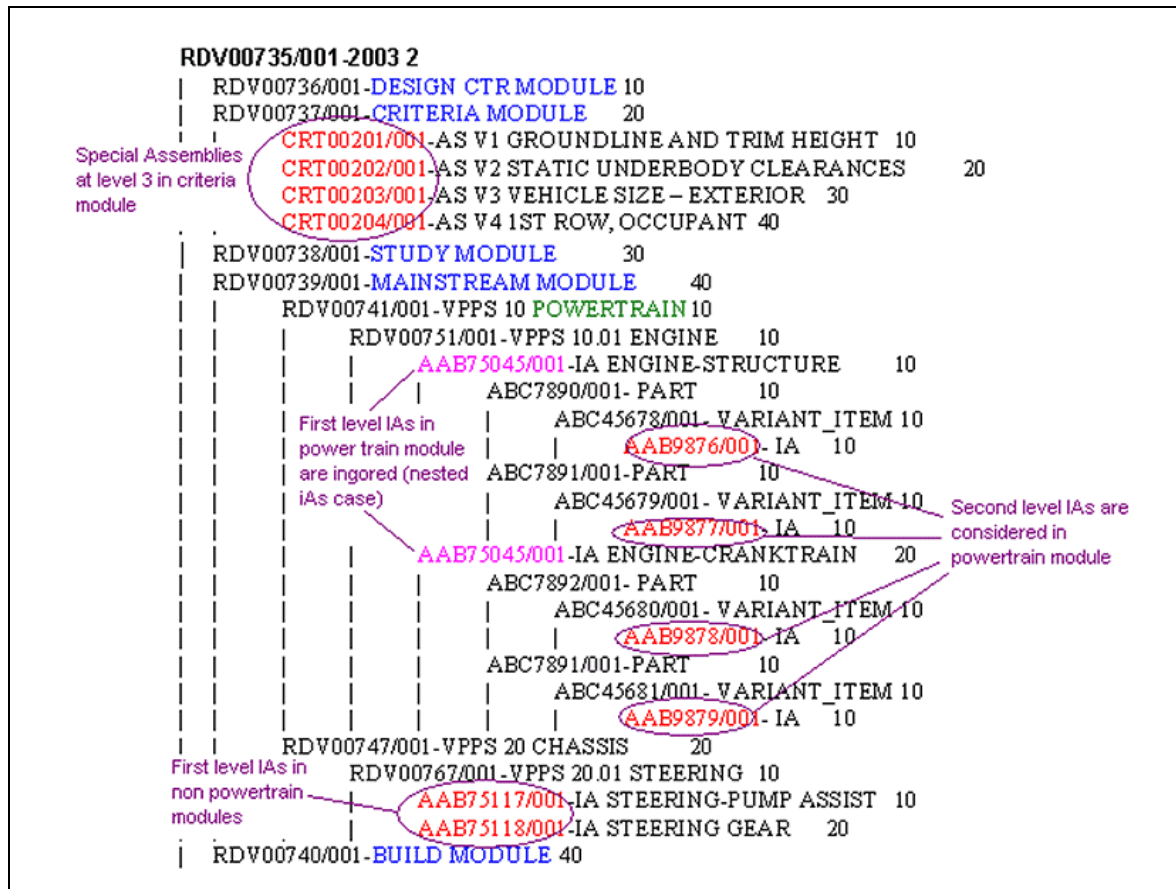
```
"-select_by_level=4:CRITERIA MODULE"
"-select_by_level=3:STUDY MODULE"
```

The **get_qpl_harvester_assemblies** utility traverses the assembly structure of the assembly by walking down from the top node. Because installation assemblies do not exist in all the branches of the assemblies, it must identify the special

assemblies in those branches that do not have installation assemblies. The **get_qpl_harvester_assemblies** utility identifies the branch by its item name. The second-level assemblies in the vehicle structure represent the branch or modules and they are of type **CORP_Proc_Plan**. This utility is provided with the list of branches or modules, which do not have the installation assemblies. Therefore, as the program walks down the structure, it checks whether the branch name matches with the ones specified in special branches. If yes, it walks further down the level specified in the input argument for this branch and identifies the assemblies. It also checks whether or not these assemblies belong to the product assembly structure site. However, if the branch name does not match the special branches specified as input, then the program walks further down to look for the installation assemblies (with item type **CORP_install**). If this branch is integrated with modules, it requires special handling, as these branches may contain nested installation assemblies. Therefore, the program walks further down from the installation assembly level to find lower level installation assemblies for these modules.

To determine whether the branch is integrated with the module, it must check whether the names of the items of type **CORP_Proc_Plan** matches the module names specified as input to the program.

The following code sample illustrates a typical vehicle assembly structure. Red nodes in the structure are those identified as installation assemblies or special assemblies by the **get_qpl_harvester_assemblies** program.



Vehicle assembly structure

SYNTAX

```
get_qpl_harvester_assemblies [-u=user-id {-p=password |
-pf=password-file} -g=group]
[-item_id=item-id] [-item_rev=rev-id] [-rev_rule=revision-rule]
[-bookmark=bkm-file]
-select_by_level=level:branch-name
-select_bottom_up=module-name
[-process_assy_list=file-with-items]
-out_modified_list=file-to-be-created
-out_special_assy_list=file-to-be-created
```

The following options can be specified multiple times:

```
-select_by_level
-select_bottom_up
```

The following arguments are optional:

```
-process_assy_list
-bookmark (If -item_id, -item_rev, and -rev_rule arguments are specified.)
-item_id, -item_rev, and -rev_rule (If -bookmark argument is specified.)
```

ARGUMENTS

-u

Specifies the user ID.

This is generally **infodba** or another user with administration privileges. If this argument is used without a value, the operating system user name is used.

Note

If Security Services single sign-on (SSO) is enabled for your server, the **-u** and **-p** arguments are authenticated externally through SSO rather than being authenticated against the Teamcenter database. If you do not supply these arguments, the utility attempts to join an existing SSO session. If no session is found, you are prompted to enter a user ID and password.

-p

Specifies the password.

If used without a value, the system assumes a null value. This argument is mutually exclusive with the **-pf** argument.

If this argument is not used, the system assumes the *user-ID* value to be the password.

-pf

Specifies the password file. If used without a value, the system assumes a null value. If this argument is not used, the system assumes the *user-ID* value to be the password.

For more information about managing password files, see [Manage password files](#).

This argument is mutually exclusive with the **-p** argument.

-g

Specifies the group associated with the user.

If used without a value, the user's default group is assumed.

-item_id

Specifies the item ID of the top-level vehicle assembly part.

-item_rev

Specifies the item revision of the top-level item.

-rev_rule

Specifies the revision rule with which the top-level vehicle assembly is configured.

-bookmark

Indicates the bookmark representing the top-level vehicle assembly structure.

Note

Use of the **-item_id**, **-item_rev**, and **-rev_rule** arguments and the **-bookmark** argument are mutually exclusive.

-process_assy_list

Specifies the process assembly list, which is a flat file containing a list of item numbers. The assemblies listed in the file are added to the modified lists (either to the installation assembly list or the special list) even though they are not newly released.

Note

If the assemblies do not satisfy one of the following requirements, they are not added to the installation assembly or special list:

1. Assembly items must belong to a specific item type, such as the installation assembly item type.
2. The assembly must be a special assembly that was selected based on the input options for the **-select_by_level** argument.

-select_by_level

Specifies module (branch) names in which the **get_qpl_harvester_assemblies** should identify the special assemblies based on the level specified in the option, for example:

```
-select_by_level=3:DESIGN CTR MODULE  
-select_by_level=4:CRITERIA MODULE
```

-select_bottom_up

Specifies the names of the items below which eligible assemblies in lower levels have precedence over eligible higher level assemblies. This option can be used to select lower-level assemblies of embedded products, for example subassemblies of an embedded power train within a vehicle product assembly structure. For example:

```
-select_bottom_up=POWERTRAIN  
-select_bottom_up=POWERTRAIN INTEGRATION
```

-out_xml_file

Specifies the name of the output structure XML file, as follows:

```
-out_xml_file=bkm_struct.xml
```

-out_modified_list

Specifies the name of the file in which the utility writes the modified or new installation assembly to be processed, as follows:

```
-out_modified_list=bkm_modified_date.txt
```

-out_special_assy_list

Specifies the name of the file in which the utility writes the special assemblies to be processed, as follows:

```
-out_special_assy_list=bkm_special_date.txt
```

-h

Displays help for this utility.

ENVIRONMENT

The Teamcenter environment must be set. Normally, this utility is run from the RDV cache maintenance scripts. This is one of the utilities in the RDV cache maintenance utility suite.

FILES

None.

RESTRICTIONS

None.

EXAMPLES

•

```
$TC_ROOT/bin/get_qpl_harvester_assemblies -u=username -p=passwd -g=group
-item_id=12344 -rev=A -rev_rule=Alpha
"-select_by_level=3:DESIGN CTR MODULE"
"-select_by_level=4:CRITERIA MODULE"
"-select_bottom_up=POWERTRAIN"
"-select_bottom_up=POWERTRAIN INTEGRATION"
-out_xml_file=/tmp/12344_A_struct.xml
```

•

```
$TC_ROOT/bin/get_qpl_harvester_assemblies -u=username -p=passwd -g=group
-bookmark=/tmp/12344_A.bkm
"-select_by_level=3:DESIGN CTR MODULE"
"-select_by_level=4:CRITERIA MODULE"
"-select_bottom_up=POWERTRAIN"
"-select_bottom_up=POWERTRAIN INTEGRATION"
-out_xml_file=/tmp/12344_A_struct.xml
```

Chapter

10 Manufacturing utilities

cc_writer

Creates a PLM XML cache file for a process structure contained in a specified collaboration context object. The PLM XML cache file can be then loaded to Teamcenter lifecycle visualization mockup from within Teamcenter.

SYNTAX

```
cc_writer [-u=user-id {-p=password | -pf=password-file} -g=group]  
-cc_name=cc-name  
| -cc_uid=uid -sc_name=structure-context-name  
| -sc_type=structure-context-type -output_file=file  
[-ua=pref:preferenceName,target:targetName1,key:keyName1,prop:propName1  
...,target:targetNameN,key:keyNameN,prop:propNameN] [-h]
```

ARGUMENTS**-u**

Specifies the user ID.

This is generally **infodba** or another user with administration privileges. If this argument is used without a value, the operating system user name is used.

Note

If Security Services single sign-on (SSO) is enabled for your server, the **-u** and **-p** arguments are authenticated externally through SSO rather than being authenticated against the Teamcenter database. If you do not supply these arguments, the utility attempts to join an existing SSO session. If no session is found, you are prompted to enter a user ID and password.

-p

Specifies the password.

If used without a value, the system assumes a null value.

If this argument is not used, the system assumes the *user-ID* value to be the password.

This argument is mutually exclusive with the **-pf** argument.

-pf

Specifies the password file. If used without a value, the system assumes a null value. If this argument is not used, the system assumes the *user-ID* value to be the password.

For more information about managing password files, see [Manage password files](#).

This argument is mutually exclusive with the **-p** argument.

-g

Specifies the group associated with the user.

If used without a value, the user's default group is assumed.

-cc_name

Specifies the name of the collaboration context object. Either this argument or the **-cc_uid** argument must be specified.

Note

The collaboration context name is not assumed to be unique. If more than one object is found, the utility returns an error message.

-cc_uid

Specifies the UID of the collaboration context object in Teamcenter. This argument is used in place of the **-cc_name** argument in situations where identifying the collaboration context object by name only is not sufficient because the collaboration context name is not unique in Teamcenter. A query must be defined to obtain the UID of a given collaboration context object.

-sc_name

Specifies the name of the structure context to export. This argument is optional. By default, the composition in the collaboration context object is exported. This argument is used only to export non-composition structure context.

-sc_type

Specifies the type of structure context to export, for example, **MEProcessContext**. This argument is optional.

-output_file

Specifies the name of the file to be created and associated with the dataset. The file is created in the current folder if a path is not specified or in the target directory if a path is specified.

-ua

Defines user attributes. The syntax is similar to the **bomwriter -ua** option.

- **pref:***preference-name-used-for-this-utility*
For example, **pref: CC_ExtraPLMXMLInstanceAttributes** picks the value of “**CC_ExtraPLMXMLInstanceAttributes**” as the user attribute.
- **target:***the-element-in-the-PLM XML-under-which-the- property-is-added*
Valid values are **Part** and **Occurrence**. If **Part**, the property is added to **ProductRevisionView**. If **Occurrence**, the property is added to **Occurrence**.
- **key:***name-of-property-in-PLM XML-file*
- **prop:***name-of-the-property-in-Teamcenter*

-h

Displays help for this utility.

RESTRICTIONS

None.

EXAMPLES

- To create the **cc_process_cache.plmxml** PLM XML file for the process structure found in the structure context of the **cc_process_cache** collaboration context object, enter the following command on a single line:

```
cc_writer -u=infodba -p=infodba -g=dba -cc_name=cc_process_cache
-sc_type= MEProcessContext -output_file=cc_process_cache.plmxml
-ua=target:Part, key:Description,prop:bl_item_object_desc, target:Occurrence,
key:FNA, prop:Usage_FNA
```

The PLM XML file contains additional attributes exported as user data. The **Description** attribute is added to the product revision element and the **FNA** attribute is added to the occurrence element. Those attributes can then be presented with their values in Teamcenter lifecycle visualization mockup.

- To create the **va_test.xml** PLM XML file for the process structure found in the structure context *process* of the collaboration context object **va_test**, enter the following command on a single line:

```
cc_writer -u=infodba -p=infodba -g=dba -cc_name=va_test  
-output_file=va_test.xml -ua=pref:CC_ExtraPLMXMLInstanceAttributes,  
target:Occurrence, key:test_name, prop:CompoundName
```

The PLM XML file contains additional attributes exported as user data. The user attribute is the combination of what is specified in the **CC_ExtraPLMXMLInstanceAttributes** preference and the **target:Occurrence,key:test_name,prop:CompoundName** attributes. Those attributes can then be presented with their values in Teamcenter lifecycle visualization mockup.

tcexcel_import

Creates process structures based on an input file, for example, generated from an Excel spreadsheet.

For more information about using this utility, see the [Manufacturing Process Planner Guide](#).

SYNTAX

```
tcexcel_import [-u=user-id {-p=password | -pf=password-file} -g=group]
-i=input-file1[:input-file2...] [-d=debug-level] [-o={on | off}]
[-t=item-type]
[-s=dummy-status] [-pf={precise | imprecise}]
[-m=marker-file-name] [-f=file-format-help] [-psfile] [-parser_only]
[-delimiter=delimiter-character] [-h]
```

ARGUMENTS

-u

Specifies the user ID.

This is generally **infodba** or another user with administration privileges. If this argument is used without a value, the operating system user name is used.

Note If Security Services single sign-on (SSO) is enabled for your server, the **-u** and **-p** arguments are authenticated externally through SSO rather than being authenticated against the Teamcenter database. If you do not supply these arguments, the utility attempts to join an existing SSO session. If no session is found, you are prompted to enter a user ID and password.

-p

Specifies the password.

If used without a value, the system assumes a null value.

If this argument is not used, the system assumes the *user-ID* value to be the password.

This argument is mutually exclusive with the **-pf** argument.

-pf

Specifies the password file. If used without a value, the system assumes a null value. If this argument is not used, the system assumes the *user-ID* value to be the password.

For more information about managing password files, see [Manage password files](#).

This argument is mutually exclusive with the **-p** argument.

-g

Specifies the group associated with the user.

If used without a value, the user's default group is assumed.

-i

Specifies input files. At least one file name is required. Additional file names are separated with a semicolon (;).

-d

Sets debug level. Default value is 0, no debug messages.

-o

Toggles overwrite mode on or off. Default is to overwrite.

-t

Specifies default item type. Default is **item**.

-s

Specifies dummy release status for working revisions. Default is **DUMMY_STATUS**.

-pf

Specifies precise mode. Default is **imprecise**.

-m

Specifies marker file name. Default value is **marker.txt**.

-f

Displays a help message describing file format.

-psfile

Specifies if the file specified by the **-i** option is a PIM file.

-parser_only

Specifies to stop after preparsing of Excel files with Header4.

-delimiter

Specifies delimiter character. Default is #.

-h

Displays help for this utility.

**INPUT FILE
FORMAT**

The input file has the following format:

- One item per line, fields separated by value of ##.
- Comments and directives start a line with #.

#DELIMITER <i>x</i>	(default:)
#ALT_DELIMITER <i>x</i>	(default: ;)
#CONSUMED_INFO <i>c1,c2 ...</i>	(optional)
#COL <i>field ...</i>	(default:item type rev attributes option loadif level seq occs uom alt matrix status link_root plant_root consumed resource required workarea occ_note abs_occ act_name act_desc activities duration predecessor owner group process_link)

Alternates to be separated by the value of **#ALT_DELIMITER**.

Valid fields are:

Field	Description
item	Item ID
rev	Revision ID (default=A)
name	Item name
revname	Item revision name
type	Item type
descr	Item description
attributes	Attributes
option	Option;Value;Value...
loadif	ItemID;Option ==/!= Value
level	Determines the structural hierarchy (top = 0)
seq	Sequence number for item in the BOM view
occs	Number of occurrences in this item
qty	Quantity of an item in the structure
uom	Unit of measure (symbol, not the name)
alt	Alternates (default delimiter=;)
matrix	Graphics position
status	Revision status
link_root	Link root as target
plant_root	Plant_root
consumed	Assign consumed items
resource	Link resources to operation
required	Assign required items
workarea	Link workareas to operation
occ_note	Attach occurrence notes to process/operation
abs_occ	Attach absolute occurrence
act_name	Activity name
act_desc	Activity description
activities	Create activities and attach forms
duration	Duration for operation activity
predecessor	Predecessor indicator (for process or operations)
owner	Change ownership to specified owner
group	Change group to specified group
process_link	Link to already existing process

import_nxcam_post_files

Imports CAM post and CSE driver data from an operating system-based file system into the Teamcenter database. For example, you can use this utility to import the files currently being used in the NX environment to the Teamcenter database. In this situation, run this utility only once to initially load existing supporting files into the Teamcenter database. Once the correct NX customer default is set, the files are stored correctly during the sessions.

SYNTAX

import_nxcam_post_files [-u=*user-id* {-p=*password* | -pf=*password-file*} -g=*group*] -action=ignore | overwrite | new_revision [-import_dir=*import-directory-name*] [-h]

ARGUMENTS**-u**

Specifies the user ID.

This is generally **infodba** or another user with administration privileges. If this argument is used without a value, the operating system user name is used.

Note

If Security Services single sign-on (SSO) is enabled for your server, the **-u** and **-p** arguments are authenticated externally through SSO rather than being authenticated against the Teamcenter database. If you do not supply these arguments, the utility attempts to join an existing SSO session. If no session is found, you are prompted to enter a user ID and password.

-p

Specifies the password.

If used without a value, the system assumes a null value.

If this argument is not used, the system assumes the *user-ID* value to be the password.

This argument is mutually exclusive with the **-pf** argument.

-pf

Specifies the password file. If used without a value, the system assumes a null value. If this argument is not used, the system assumes the *user-ID* value to be the password.

For more information about managing password files, see [Manage password files](#).

This argument is mutually exclusive with the **-p** argument.

-g

Specifies the group associated with the user.

If used without a value, the user's default group is assumed.

-action

Specifies the actions performed during the import.

- **ignore**

Does not import objects that already exist in the database.

- **overwrite**

Overwrites objects that already exist in the database.

- **new_revision**

Creates a new revisions of the objects that already exists in the database. This is the default setting.

-import_dir

Specifies the directory containing the CAM post and CSE driver data to be imported. If this argument is not used, one of the environment variables listed in the *Environment* section must be set.

-h

Displays help for this utility.

ENVIRONMENT

As specified in [Manually configuring your environment for Teamcenter utilities](#).

If the **-import_dir** argument is not specified, one of the following environment variables must be set before running this utility:

- **UGII_CAM_POST_DIR**

Imports global postprocessor files and/or posts and ISV drivers for generic machines.

- **UGII_CAM_USER_DEF_EVENT_DIR**

Imports user-defined event files.

- **UGII_CAM_SHOP_DOC_DIR**

Imports shop documentation files.

- **UGII_CAM_LIBRARY_INSTALLED_MACHINES_DIR**

Imports the post/CSE files of installed machines.

For additional information on these environment variables, see the NX documentation.

FILES

As specified in [Log files produced by Teamcenter](#).

RESTRICTIONS

None.

EXAMPLES

To import machine post and CSE data from **d:\import_dir** directory using the default action, enter the following command on a single line:

```
%TC_ROOT%\bin\import_nxcam_post_files -u=infodba -p=infodba-password -g=dba
-import_dir=d:\import_dir
```

To import machine post and CSE data from the **d:\mach\resource** directory using the **-overwrite** action, enter the following command on a single line:

```
%TC_ROOT%\bin\import_nxcam_post_files -u=infodba -p=infodba-password -g=dba
-action=overwrite -import_dir=d:\mach\resource
```

To import only global postprocessor files from the **d:\mach\resource\postprocessor** directory to the database using the default action, enter the following two commands.

```
set UGII_CAM_POST_DIR= d:\mach\resource\postprocessor
%TC_ROOT%\bin\import_nxcam_post_files -u=infodba -p=infodba-password -g=dba
-import_dir=d:\import_dir
```

upgrade_nx_cam_templates

Replaces NX/CAM setup templates in the Teamcenter database.

Note The templates with a component are imported from the directory specified by the value of **TC_DATA**; templates without a component are imported from the directory specified by the value of **UGII_BASE_DIR**.

SYNTAX

```
upgrade_nx_cam_templates [-u=user-id {-p=password |  
-pf=password-file} -g=group]  
[-included_list=included-file-name] [-excluded_list=excluded-file-name] [-h]
```

ARGUMENTS

-u

Specifies the user ID.

This is generally **infodba** or another user with administration privileges. If this argument is used without a value, the operating system user name is used.

Note If Security Services single sign-on (SSO) is enabled for your server, the **-u** and **-p** arguments are authenticated externally through SSO rather than being authenticated against the Teamcenter database. If you do not supply these arguments, the utility attempts to join an existing SSO session. If no session is found, you are prompted to enter a user ID and password.

-p

Specifies the password.

If used without a value, the system assumes a null value.

If this argument is not used, the system assumes the *user-ID* value to be the password.

This argument is mutually exclusive with the **-pf** argument.

-pf

Specifies the password file. If used without a value, the system assumes a null value. If this argument is not used, the system assumes the *user-ID* value to be the password.

For more information about managing password files, see [Manage password files](#).

This argument is mutually exclusive with the **-p** argument.

-g

Specifies the group associated with the user.

If used without a value, the user's default group is assumed.

-included_list

Specifies a text file with a template ID list that is replaced in the Teamcenter database. The complete file specification, that is, the full path and file name, must be supplied. This is an optional argument. If no file is provided, all CAM templates in the Teamcenter **CAM Setup Templates** folder are replaced.

-excluded_list

Specifies a text file with template ID list that is not replaced in the Teamcenter database. The complete file specification, that is, full path and file name, must be supplied. This is an optional argument. If no file is provided, then all CAM templates in the Teamcenter **CAM Setup Templates** folder are replaced.

-h

Displays help for this utility.

ENVIRONMENT

The **UGII_BASE_DIR** and **TC_DATA** variables must be set to use this utility.

FILES

None.

RESTRICTIONS

None.

EXAMPLES

- To replace all templates in the **CAM Setup Templates** folder:

```
upgrade_nx_cam_templates -u=infodba -p=infodba -g=dba
```

- To replace only the templates listed in the **in_file.txt** file in the **d:\workdir** folder:

```
upgrade_nx_cam_templates -u=infodba -p=infodba -g=dba  
-included_list=d:\workdir\in_file.txt
```

The content of the **in_file.txt** file is:

```
drill_inch  
drill_metric  
turning_inch  
turning_metric
```

Only four templates, **drill_inch**, **drill_metric**, **turning_inch**, and **turning_metric**, in the **in_file.txt** file are replaced.

- To replace all templates in the **CAM Setup Templates** folder *except* the templates listed in the **ex_file.txt** file in the **d:\workdir** folder.

```
upgrade_nx_cam_templates -u=infodba -p=infodba -g=dba  
-excluded_list=d:\workdir\ex_file.txt
```

The content of the **ex_file.txt** file is:

```
drill_inch  
drill_metric  
turning_inch  
turning_metric
```

Four templates, **drill_inch**, **drill_metric**, **turning_inch**, and **turning_metric**, in the **ex_file.txt** file are *not* replaced.

mrm_export_resources

Exports resources from Teamcenter classification. This is useful if you manage your resource and classification data in Teamcenter but run NX in its native mode. In this situation, you can use this utility to export tooling classification and data from Resource Manager so it can be used by NX CAM in native mode.

For more information about using Resource Manager tooling data with native NX CAM, see [Getting Started with Manufacturing](#).

SYNTAX

```
mrm_export_resources [-u=user-id {-p=password | -pf=password-file} -g=group]
  -class=root-class-ID
  -def_file=definition-file [-class_graphics_dir=directory
  [-class_graphics_option=all | changed]]
  -dat_file=database-file [-dat_graphics_dir=directory
  [-dat_graphics_option=all | changed]] [-h]
```

ARGUMENTS

-u

Specifies the user ID.

This is generally **infodba** or another user with administration privileges. If this argument is used without a value, the operating system user name is used.

Note

If Security Services single sign-on (SSO) is enabled for your server, the **-u** and **-p** arguments are authenticated externally through SSO rather than being authenticated against the Teamcenter database. If you do not supply these arguments, the utility attempts to join an existing SSO session. If no session is found, you are prompted to enter a user ID and password.

-p

Specifies the password.

If used without a value, the system assumes a null value.

If this argument is not used, the system assumes the *user-ID* value to be the password.

This argument is mutually exclusive with the **-pf** argument.

-pf

Specifies the password file. If used without a value, the system assumes a null value. If this argument is not used, the system assumes the *user-ID* value to be the password.

For more information about managing password files, see [Manage password files](#).

This argument is mutually exclusive with the **-p** argument.

-g

Specifies the group associated with the user.

If used without a value, the user's default group is assumed.

-class

Exports the classification hierarchy and/or instance data from the specified root class ID. Use the **-def_file** and **-dat_file** arguments to specify whether classification hierarchy, and/or instance data, is exported, respectively.

-def_file

Exports the specified definition file. The file is based on the class hierarchy structure.

Note

The definition file includes the list of required attributes and key-LOV definitions. It also includes the hierarchical structure of the nested classes.

For each class, the list of available attributes for the **Search** dialog box and the resulting table must be defined. If a definition for the **RSET** attribute set is defined, the attributes specified for the **Search** dialog box is used for the **RSET** attribute set.

For an example of the definition file, see the *Examples* section.

-class_graphics_dir

Exports classification graphics used for the **Search** and the **RSET** dialog boxes to the specified directory.

-class_graphics_option

Specifies whether to export all classification graphics or only those graphics that are modified. Valid values are **all**, which exports all graphics, and **changed**, which exports only graphics that are modified.

-dat_file

Exports the specified ASCII database file. The file includes parameter values for all instances. Each data line also specifies the tool classification.

-dat_graphics_dir

Exports resource graphics to the local file system.

-dat_graphics_option

Specifies whether to export all resource graphics or only those graphics that are modified. Valid values are **all**, which exports all graphics, and **changed**, which exports only graphics that are modified.

-h

Displays help for this utility.

ENVIRONMENT

As specified in [Manually configuring your environment for Teamcenter utilities](#).

FILES

As specified in [Log files produced by Teamcenter](#).

RESTRICTIONS

None.

EXAMPLES

To export the definition file for the Manufacturing Tooling Library:

```
mrm_export_resources -u=infodba -p=infodba -g=dba -class=TLCUA
-def_file=dbc_mfg_toollib_customer_tlas.def
```

To export the ASCII tool database file for all tool assemblies, including all updated graphic files:

```
mrm_export_resources -u=infodba -p=infodba -g=dba -class=TOOL02
-dat_file=tool\metric\tool_database.dat -dat_graphics_dir= tool\metric\graphics\ -dat_gra
```

A sample definition file:

```
DB_ALIAS DCII
{
  DB_ID      -500118
  DB_ID_TYPE s
  DIALOG_NAME "Cutting diameter Dc"
  RSET_NAME   "Cutting diameter Dc"
}
DB_ALIAS PartType
{
  DB_ID      -3651
  DB_ID_TYPE s
  OPTIONS    "Right" "Left"
  OPTIONS_IDS "R" "L" DIALOG_NAME "Type"
  RSET_NAME  "Type"
}
CLASS TOOL
{
  TYPE QRY
  QUERY "[DB(Type)] == [TOOL01] && [DB(Type)] == [TLCUA]"
  DIALOG libref
  RSET libref
  UI_NAME "Tool"
}
CLASS TOOL_MRM
{
  TYPE QRY
  QUERY "[DB(Type)] == [TOOL01]"
  DIALOG libref Holder
  RSET libref Descr MaterialDes Holder
  UI_NAME "MRM Tooling"
}
CLASS MILLS
{
  TYPE QRY
  QUERY "[DB(Type)] == [TAM02]"
  DIALOG libref Diameter Holder
  RSET libref Descr Diameter MaterialDes Holder
  UI_NAME "Milling"
}
```

gcs_import

Imports GCS connection types and connection point definitions from an XML file.

For more information about using the guided component search, see the [Resource Manager Guide](#).

SYNTAX

```
gcs_import [-u=user-id {-p=password | -pf=password-file} -g=group]  
-xml_file=xml_file_name -import_mode=ignore | overwrite [-h]
```

ARGUMENTS**-u**

Specifies the user ID.

This is generally **infodba** or another user with administration privileges. If this argument is used without a value, the operating system user name is used.

Note

If Security Services single sign-on (SSO) is enabled for your server, the user and password arguments are authenticated externally through SSO rather than being authenticated against the Teamcenter database. If you do not supply these arguments, the utility attempts to join an existing SSO session. If no session is found, you are prompted to enter a user ID and password.

-p

Specifies the user's password.

If used without a value, the system assumes a null value. If this argument is not used, the system assumes the *user-ID* value to be the password.

This argument is mutually exclusive with the **-pf** argument.

-pf

Specifies the password file. If used without a value, the system assumes a null value. If this argument is not used, the system assumes the *user-ID* value to be the password.

For more information about managing password files, see [Manage password files](#).

This argument is mutually exclusive with the **-p** argument.

-g

Specifies the group associated with the user.

If used without a value, the user's default group is assumed.

-xml_file

Specifies the name of an XML file containing connection types and connection point definitions. The syntax of this file is explained in the *EXAMPLES* section.

-import_mode

Specifies how existing data is handled by the import. Valid values are:

ignore The existing data is skipped by the import.

overwrite The existing data is replaced by the contents of the XML file.

-h

Displays help for this utility.

ENVIRONMENT

As specified in [Manually configuring your environment for Teamcenter utilities](#).

FILES

As specified in [Log files produced by Teamcenter](#).

RESTRICTIONS

None.

EXAMPLES

- To import the connection type and connection point definition found in an XML file named **GCS_data.xml**, enter the following command on a single line:

```
GCS_import -u=user-id -p=password -g=group -xml_file=GCS_data.xml
-import_mode=ignore
```

The **GCS_data.xml** file must have the following format:

```
<GCS_import>
<ConnectionType>
  <Name>CT_Cylinder</Name>
  <Attribute>
    <ID>-40032</ID>
    <ComparisonCriterion>&lt;=</ComparisonCriterion>
  </Attribute>
  <Attribute>
    <ID>-40033</ID>
    <ComparisonCriterion>&gt;=</ComparisonCriterion>
  </Attribute>
</ConnectionType>
<ConnectionType>
  <Name>CT_Square</Name>
  <Attribute>
    <ID>-40127</ID>
    <ComparisonCriterion>=</ComparisonCriterion>
  </Attribute>
</ConnectionType>
<ConnectionType>
  <Name>CT_Insert</Name>
  <Attribute>
    <ID>-40920</ID>
    <ComparisonCriterion>=</ComparisonCriterion>
  </Attribute>
</ConnectionType>
<ConnectionPointDefinition>
  <ClassID>TC_MILL_10_10_100</ClassID>
  <ConnectionTypeName>CT_Cylinder</ConnectionTypeName>
  <Index>1</Index>
  <Quantity>1</Quantity>
  <Direction>Upwards</Direction>
  <Shape>Plug</Shape>
  <Attribute>
    <ID>-40032</ID>
    <Mapping>#-40235</Mapping>
  </Attribute>
  <Attribute>
    <ID>-40033</ID>
    <Mapping>#-40235</Mapping>
  </Attribute>
</ConnectionPointDefinition>
<ConnectionPointDefinition>
  <ClassID>TC_DRILL_12_10_100</ClassID>
  <ConnectionTypeName>CT_Cylinder</ConnectionTypeName>
```

```

        <Index>1</Index>
        <Quantity>1</Quantity>
        <Direction>Upwards</Direction>
        <Shape>Plug</Shape>
        <Attribute>
            <ID>-40032</ID>
            <Mapping>#-40235</Mapping>
        </Attribute>
        <Attribute>
            <ID>-40033</ID>
            <Mapping>#-40235</Mapping>
        </Attribute>
    </ConnectionPointDefinition>
    <ConnectionPointDefinition>
        <ClassID>TC_HOLDER_20_00_190</ClassID>
        <ConnectionTypeName>CT_Cylinder</ConnectionTypeName>
        <Index>1</Index>
        <Quantity>1</Quantity>
        <Direction>Upwards</Direction>
        <Shape>Socket</Shape>
        <Attribute>
            <ID>-40032</ID>
            <Mapping>#-40032</Mapping>
        </Attribute>
        <Attribute>
            <ID>-40033</ID>
            <Mapping>#-40033</Mapping>
        </Attribute>
    </ConnectionPointDefinition>
    <ConnectionPointDefinition>
        <ClassID>TC_HOLDER_10_00_110</ClassID>
        <ConnectionTypeName>CT_Square</ConnectionTypeName>
        <Index>1</Index>
        <Quantity>1</Quantity>
        <Direction>Upwards</Direction>
        <Shape>Socket</Shape>
        <Attribute>
            <ID>-40127</ID>
            <Mapping>#-40127</Mapping>
        </Attribute>
    </ConnectionPointDefinition>
    <ConnectionPointDefinition>
        <ClassID>TC_TURN_10_10_100</ClassID>
        <ConnectionTypeName>CT_Square</ConnectionTypeName>
        <Index>1</Index>
        <Quantity>1</Quantity>
        <Direction>Upwards</Direction>
        <Shape>Plug</Shape>
        <Attribute>
            <ID>-40127</ID>
            <Mapping>#-40238</Mapping>
        </Attribute>
    </ConnectionPointDefinition>
    <ConnectionPointDefinition>
        <ClassID>TC_TURN_10_10_100</ClassID>
        <ConnectionTypeName>CT_Insert</ConnectionTypeName>
        <Index>2</Index>
        <Quantity>1</Quantity>
        <Direction>Downwards</Direction>
        <Shape>Socket</Shape>
        <Attribute>
            <ID>-40920</ID>
            <Mapping>#-40920</Mapping>
        </Attribute>
    </ConnectionPointDefinition>
    <ConnectionPointDefinition>

```

```
<ClassID>TC_INSERT_10_00_110</ClassID>
<ConnectionTypeName>CT_Insert</ConnectionTypeName>
<Index>1</Index>
<Quantity>1</Quantity>
<Direction>Upwards</Direction>
<Shape>Plug</Shape>
<Attribute>
  <ID>-40920</ID>
  <Mapping>#-40920</Mapping>
</Attribute>
</ConnectionPointDefinition>
</GCS_import>
```

assy_jt_creator

Creates a JT representation of a given Teamcenter assembly. The resulting file is stored in the file system and, optionally, in the database.

You can use the utility to collect the JT files on each node in a configured structure and combine all JT files into a single monolithic assembly JT. You can use a monolithic JT file to view assemblies in Resource Browser.

The utility can also create an assembly JT file that contains pointers to other JT files, each representing a component in an assembly. Each node can have a JT file and a transformation of that node.

Additionally, the utility can create a JT file for use with Tecnomatix eMServer only.

SYNTAX

```
assy_jt_creator -u=user-id -p={password | -pf=password-file} [-g=group ]
{-sc=structure-context |
```

```
{ -item=item-id | -key=[keyAttr1=keyVal1][,keyAttr2=keyVal2]...[,keyAttrN=keyValN] }
```

```
-tar_loc=target-location
```

```
[-i=batch-file]
```

```
[-log=logfile-name]
```

```
[-rev=revision]
```

```
-rev_rule=revision-rule
```

```
[-var_rule=variant-rule]
```

```
[-tmode=transfer-mode-name]
```

```
[-jt_type={1 | 2 | 3} ]
```

```
[-pin=0 | 1]
```

```
[-cof=0 | 1]
```

```
[-dt_type=dataset-type] [-dt_rel=dataset-relation]
```

```
[-dt_ref=named-reference]
```

```
[-de=n | e | a | r]
```

```
[-assy_update={create | replace} ]
```

```
[-skip_root_dset_with_name=dataset-name]
```

```
[-skip_remote_lines]
```

```
[-debug]
```

```
[-h]
```

ARGUMENTS**-u**

Specifies the user ID.

This is generally **infodba** or another user with administration privileges. If this argument is used without a value, the operating system user name is used.

Note

If Security Services single sign-on (SSO) is enabled for your server, the **-u** and **-p** arguments are authenticated externally through SSO rather than being authenticated against the Teamcenter database. If you do not supply these arguments, the utility attempts to join an existing SSO session. If no session is found, you are prompted to enter a user ID and password.

-p

Specifies the password.

If used without a value, the system assumes a null value.

If this argument is not used, the system assumes the *user-ID* value to be the password.

This argument is mutually exclusive with the **-pf** argument.

-pf

Specifies the password file. If used without a value, the system assumes a null value. If this argument is not used, the system assumes the *user-ID* value to be the password.

For more information about managing password files, see [Manage password files](#).

This argument is mutually exclusive with the **-p** argument.

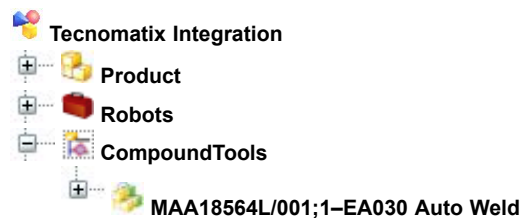
-g

Specifies the group associated with the user.

If used without a value, the user's default group is assumed.

-sc

Specifies the name of the structure context (see **CompoundTools** in the following figure). If you include this argument, item ID, revision ID, revision rule, variant rule, and transfer mode arguments are not required.



For more information about structure contexts, see the [Multi-Structure Manager Guide](#).

-item

Specifies the item ID of the top line.

If you do not include the **-sc** argument, either this argument or the **-key** argument is required.

-key

Specifies the key of the top line when multiple attributes are used to form the unique item ID. Use the following format:

```
[keyAttr1=keyVal1] [,keyAttr2=keyVal2]...[,keyAttrN=keyValN]
```

To find the key of an object, use the [get_key_string](#) utility.

For more information, see the [Business Modeler IDE Guide](#).

-tar_loc

Specifies the full operating system path to an existing location where Teamcenter stores the JT assembly file.

This argument is required.

-i

Specifies an input batch file. This XML file uses elements that are equivalent to the command line arguments and contains information about the assembly JT files

you want included in the monolithic JT assembly. Use this method to simplify the input of parameters. This allows you save a set of arguments you can use each time you run the utility.

This file must contain the following elements as a minimum (all defaults are used in this file):

Note For more information, see the equivalent command line argument description (identified in comments in the file listings).

```
<ParamBatch> <!-- Required root element -->
  <ParamSet id=""> <!-- Element holding JT file parameters -->
    <!--If a <StructureContext> element is not included, the <ItemId>
      or <KeyId> element is required. See next file listing. -->
    <StructureContext> </StructureContext> <!-- See -sc argument -->
    <TargetLoc> </TargetLoc> <!-- See -tar_loc argument -->
    <DataSetType> </DataSetType> <!-- See -dt_type argument -->
  </ParamSet>
</ParamBatch>
```

The **ParamBatch** element contains all **ParamSet** elements. A separate **ParamSet** element with a unique **id** attribute is required for each JT file you include in the assembly.

This file shows all possible elements that you can specify with empty elements for informational purposes. No defaults are used. Unless an element contains only attributes, which is not the case for an input batch file, well-formed XML requires that you provide a value between the element opening and closing tag. Therefore, do not include empty elements if you want to use the default value.

```
<ParamBatch>
  <ParamSet id="">
    <ItemId> </ItemId> <!-- See -item argument -->
  <!-- If multiple key attributes (multifield key) are used, the <KeyId>
    element replaces the <ItemId> element. See -key
  -->
    <RevId> </RevId> <!-- See -rev argument -->

    <RevisionRule> </RevisionRule> <!-- See -rev_rule argument -->
    <!-- If the revision rule is not specified, the utility uses the default
      "Latest Working" revision rule.
    -->

    <VariantRule> </VariantRule> <!-- See -var_rule argument -->

    <TransferMode> </TransferMode> <!-- See -tmode argument -->
    <!-- If TransferMode is not specified, the utility uses the default
      "JTDataExportDefault" transfer mode -->

    <JtAssyType> </JtAssyType> <!-- See -jt_type argument-->
    <!-- The type of assembly to be created can be specified as:
      1 - Monolithic JT
      2 - Assembly JT
      3 - CoJT.
      The utility uses "Monolithic" if this is not specified
    -->

    <ProcessIntermediateNode> </ProcessIntermediateNode> <!-- See -pin argument -->
    <!-- ProcessIntermediateNode indicates if the intermediate nodes need to be
      processed. This is a boolean value (1/0). If this is not specified,
      The utility uses 1 - intermediate nodes gets processed.
    -->

    <ContinueOnFail> </ContinueOnFail> <!-- See -cof argument -->
```

```

<!-- ContinueOnFail specifies whether the creation of the assembly should
      continue if any errors are encountered. This is a boolean value (1/0)
      If this is not specified, the utility uses 1 - continue with creation
      of the assembly.
-->

<DatasetType> </DatasetType> <!-- See -dt_type argument -->
<!-- The dataset type used for attaching the resulting assembly to the Item
      revision. If this is not specified or the type is not found, the
      utility uses a default value based on the JtAssytType value.
-->

<DatasetRelation> </DatasetRelation> <!-- See -dt_rel argument -->
<!-- The dataset relation used for attaching the dataset to the Item
      revision.
-->

<LogFile> </LogFile> <!-- See -log argument -->
<!-- LogFile identifies the complete path and the file name for the log
      file the utility generates.
-->

<UpdateType> </UpdateType> <!-- See -assy_update argument -->
<!-- UpdateType determines how to attach the dataset to the Item revision.
      If this is not specified, the utility replaces the named reference,
      if one exists, with the dataset.
-->

<TargetLoc> </TargetLoc>
<!-- TargetLoc specifies the path to the location on your operating system
      where the utility saves the new JT assembly file.-->

<CreateDataset> </CreateDataset> <!-- See -de argument -->
<!-- The valid values for CreateDataset are:
      n - creates a new dataset every time.
      e - if the dataset already exists it does not revise it. If the dataset
          does not exist it is create.
      a - if the dataset already exists it revises it. If the dataset does not
          exist it is created.
      r - revises the dataset if it already exists. If the dataset does not
          exist an error is thrown.
      If this is not specified, the utility does not create a dataset.
-->

<NamedRef> </NamedRef> <!-- See -dt_ref argument -->
<!-- The named reference used for uploading the assembly file to the dataset.
      If this is not specified, the utility uses the default named reference
      attached to the dataset type.
-->

</ParamSet> <!-- End of JT file parameters -->
: <!-- Additional ParamSet elements for each set of JT file parameters are here in -->
</ParamBatch> <!-- End required root element -->

```

See [Examples](#) for a sample XML file listing.

Note If an input XML file is specified, all other command line arguments (with the exception of the log on (**-u**, **-p** or **-pf**, and **-g** arguments) are ignored.

-log

Specifies the full path file name of the log file that stores PLM XML and load library errors.

-rev

Specifies the revision ID of the top line.

If you do not include the **-sc** argument, this argument is required.

-rev_rule

Specifies the revision rule for configuring the assembly.

-var_rule

Specifies the variant rule for configuring the assembly.

-tmode

Specifies the name of the transfer mode. The transfer mode is not stored in the system but is created at run time.

If you do not include this argument, the utility uses the default **JTDataExportDefault** transfer mode.

-jt_type

Specifies the type of assembly to be created. Valid values are:

1

A monolithic JT file is a single file containing all the graphics for a specific assembly.

2

An assembly JT file is an assembly file that contains pointers to other JT files, each representing a component in an assembly.

3

A CoJT file is for use with Tecnomatix eMServer only.

If you do not specify this argument, the utility uses **1** (Monolithic JT).

-pin

Indicates whether to process intermediate nodes. Intermediate nodes are any nodes that are not leaf nodes. Valid values are:

0

Do not process intermediate nodes.

1

Process intermediate nodes.

If you do not specify this argument, the utility uses **1** (process intermediate nodes).

-cof

Indicates whether to terminate assembly creation upon encountering errors. Valid values are:

0

Stop process when encountering an error.

1

Continue processing when encountering an error.

If you do not specify this argument, the utility uses **1** (continue processing).

-dt_type

Specifies dataset type used for attaching the resulting assembly to the item revision.

-dt_rel

Specifies the dataset relation used for attaching the dataset to the item revision.

-dt_ref

Specifies the named reference used for uploading the assembly file to the dataset. If no named reference is given, the default named reference attached to the dataset type is used.

-de

Specifies the behavior of dataset creation.

If you do not enter a value for this parameter, Teamcenter creates a JT file without an associated dataset. Valid value are:

n

Creates a new dataset.

e

Does not revise an existing dataset. If the dataset does not exist, it creates the dataset.

a

Revises an existing dataset. If the dataset does not exist, it creates the dataset.

r

Revises an existing dataset. If the dataset does not exist, it generates an error.

-assy_update

Specifies how the assembly JT is added to the dataset.

create

Assembly file is uploaded if no named reference is found in the dataset.

replace

Assembly file is always created and replaces the current named reference, if it exists.

-skip_root_dset_with_name

Prevents the utility from exporting the dataset of this name that is associated to the root node by an **IMAN_Rendering** relation.

-skip_remote_lines

Prevents the utility from exporting BOM lines that are not loadable.

-Debug

Turns on the debug mode that creates additional debug information files in the log file location.

-h

Displays help for this utility.

ENVIRONMENT

As specified in [Manually configuring your environment for Teamcenter utilities](#).

FILES

As specified in *Log files produced by Teamcenter*.

RESTRICTIONS

None.

EXAMPLES

- The following example is for an input XML file (**robots.xml**) that uses the **StructureContext** element and therefore does not require the **ItemId**, **RevId**, **RevisionRule**, **VariantRule** or **Transfermode** elements. The default **Latest Working** revision rule and **JTDataExportDefault** transfer mode are used. The default dataset type for a monolithic JT assembly is used to attach the dataset to the JT assembly because the **DatasetType** element is not specified:

```
<!-- assy_jt_creator utility robots.xml input file -->
<ParamBatch>
  <ParamSet id="001">
    <StructureContext>CompoundTools</StructureContext>
    <ItemId />
    <RevId />
    <RevisionRule />
    <VariantRule />
    <TransferMode />
    <JtAssyType>1</JtAssyType>
    <ProcessIntermediateNode>1</ProcessIntermediateNode>
    <ContinueOnFail>1</ContinueOnFail>
    <DatasetType />
    <DatasetRelation>IMAN_reference</DatasetRelation>
    <LogFile>c:\temp\monolithicJTUtility\logfile.log</LogFile>
    <UpdateType>create</UpdateType><CreateDataset></CreateDataset>
    <TargetLoc>c:\temp\monolithicJTUtility</TargetLoc>
    <CreateDataSet>n</CreateDataSet>
    <NamedRef>JTPart</NamedRef>
  </ParamSet>
  <!-- uncomment this section to add additional JT file parameters
  <ParamSet id="002">
    :
  </ParamSet>
-->
</ParamBatch>
```

Use the following command to create the JT assembly using the **robots.xml** file:

```
assy_jt_creator -u=infodba -p=infodba -g=dba -i=robots.xml
```

- The following example shows the same command line version creating the same monolithic JT assembly shown in the previous XML file example. However, the command line version does not continue to create the assembly if an error is encountered during the process, does not revise the dataset if one exists, and uses a custom transfer mode. A debug file is also created in the log directory. The entire command must be entered on a single line.

```
assy_jt_creator -u=infodba -p=infodba -g=dba -sc=Robots -jt_type=1
-tar_loc=c:\temp\monolithicJTUtility -tmode=JTDataExportCustomTM
-pin=1 -cof=0 -dt_rel=IMAN_reference -assy_update=create -de= -dt_ref=JTPart
-debug
```

ipa_b_executer

Generates or updates in-process assemblies and updates filtered in-process assemblies in a manufacturing process structure. The input for this utility is a parameter file in which you specify whether to update either the IPA, the FIPA, or both simultaneously. You cannot create FIPAs with this utility. You can only create FIPAs from within the Manufacturing Process Planner application.

Note Depending on the size of the structure, running this utility can be time-consuming.

For more information about in-process assemblies, see the [Manufacturing Process Planner Guide](#).

SYNTAX

```
ipa_b_executer -u=user-id {-p=encrypted-password | -pf=password-file} [-g=group
-f=path-to-parameter-file]
[-h]
```

ARGUMENTS

-u

Specifies the user ID.

This is generally **infodba** or another user with administration privileges. If this argument is used without a value, the operating system user name is used.

Note If Security Services single sign-on (SSO) is enabled for your server, the user and password arguments are authenticated externally through SSO rather than being authenticated against the Teamcenter database. If you do not supply these arguments, the utility attempts to join an existing SSO session. If no session is found, you are prompted to enter a user ID and password.

-p

Specifies the user's password. This password must be encrypted

You can obtain an encrypted password by typing:

```
ipa_b_executer -p=plain-text-password -encrypt
```

Teamcenter returns the encrypted password as output.

This argument is mutually exclusive with the **-pf** argument.

-pf

Specifies the password file. If used without a value, the system assumes a null value. If this argument is not used, the system assumes the *user-ID* value to be the password.

For more information about managing password files, see [Manage password files](#).

This argument is mutually exclusive with the **-p** argument.

-g

Specifies the group associated with the user.

If used without a value, the user's default group is assumed.

-f

Specifies the path to the parameter file (**.txt** file) that is used as input for this utility.
For more information, see [Parameter file syntax](#).

-h

Displays help for this utility.

ENVIRONMENT

As specified in [Manually configuring your environment for Teamcenter utilities](#).

FILES

As specified in [Log files produced by Teamcenter](#).

RESTRICTIONS

None.

**PARAMETER
FILE SYNTAX**

The first entry in each line in the parameter file indicates whether an IPA or FIPA is to be updated. Therefore, there are two types of lines in this file:

- The parameter line for an IPA starts with **#Item**
- The parameter line for a FIPA starts with **#Item_FIPA**

Given these two types of lines, three cases can exist:

- If only an **#Item** line is present in the file, Teamcenter updates the IPA only.
- If only an **#Item_FIPA** line is present in the file, Teamcenter updates the FIPA only.
- If both **#Item** and **#Item_FIPA** lines are present in the file, Teamcenter updates both the IPA and the FIPA.

In the line, each parameter must be separated from its value by the **%** symbol. If there is more than one value for a parameter, each of these values must be separated by a **%** symbol.

You can use the following parameters to create or update an IPA:

Item

Specifies the top process ID.

Key

Specifies the top process key in the form of **attr=value,attr2=value2...**

Rev

Specifies the top process revision.

Rule

Specifies the configuration rule.

OG

Specifies the type of IPA occurrence group.

Name

Specifies the name of the IPA.

Occ

Specifies the types of the product occurrences that are included.

Proc

Specifies the types of the processes for which an occurrence group is created.

Use the following parameters to update a filtered IPA:

Item_FIPA

Specifies the process ID of the top process of the BOP structure in which FIPAs are to be updated. This parameter can have only one value.

Rev

Specifies the top process revision. This parameter can have only one value.

Rule

Specifies the revision rule with which the FIPAs are to be updated. This parameter can have only one value.

Parent_FIPA

Specifies the process IDs of the processes for which FIPAs are to be updated. This is mandatory parameter that can have multiple values. In the case of multiple process IDs, the IDs must be separated by a % symbol.

Name

Specifies that only this FIPA in the **Parent_FIPA** process is updated. This is an optional parameter. If no name is specified, then all the FIPAs under the **Parent** process are updated. This parameter can have multiple values.

Prod_ID

Specifies the ID of the top product of the product structure from which items are consumed into the process structure. This parameter can have only one value.

Prod_Rev

Specifies the revision of the top product line. This parameter can have only one value.

Prod_Rule

Specifies the revision rule applied on the product structure. This parameter can have only one value.

The following is a sample line for updating FIPAs:

```
#Item_FIPA%Top-process-id##Rev%current-revision-level##Rule%current-revision-rule
#Parent%process-id-1[process-id-2..
process-id-n]#Name%fipa-name-1[%fipa-name-2..%fipa-name-n]
```

EXAMPLES

These examples reference the following structure.

Process Structure	Item Type	Occurrence Type	Bounding Boxes
000029/A;1-demoNested (View)	MEProcess		
Proc6_fipa	MEProximityResult	MEFilteredIPA	
Proc6_assy	OccurrenceGroup	MEWorkpiece	
Proc5_assy	OccurrenceGroup		
000019/A;1-D6	Item		-0.004762500000000...
000030/A;1-Proc1 (View)	MEProcess		
000014/A;1-D1	Item	MEConsumed	-0.012220138907432...
000031/A;1-Proc2 (View)	MEProcess		
Proc1_fipa	MEProximityResult	MEFilteredIPA	
Proc1_assy	OccurrenceGroup	MEWorkpiece	
000015/A;1-D2	Item	MEConsumed	0.000000000000000, ...
000032/A;1-Proc3 (View)	MEProcess		
Proc2_assy	OccurrenceGroup	MEWorkpiece	
000016/A;1-D3	Item	MEConsumed	0.000000000000000, ...
000033/A;1-Proc4 (View)	MEProcess		
Proc3_assy	OccurrenceGroup	MEWorkpiece	
000017/A;1-D4	Item	MEConsumed	-0.000793750000000...
000034/A;1-Proc5 (View)	MEProcess		
Proc4_fipa	MEProximityResult	MEFilteredIPA	
Proc4_assy	OccurrenceGroup	MEWorkpiece	
000018/A;1-D5	Item	MEConsumed	-0.004165600115060...
000035/A;1-Proc6 (View)	MEProcess		
Proc5_fipa	MEProximityResult	MEFilteredIPA	
Proc5_assy	OccurrenceGroup	MEWorkpiece	
000019/A;1-D6	Item	MEConsumed	-0.004762500000000...

- To update only the IPA:

```
#Item%000029##Rev%A##Rule%Latest Working##OG%OccurrenceGroup#
#Name%IPA -Jun 30, 2011 12:01:21 PM##Occ%MEConsumed#
#Proc%MEProcess#
```

The file does not contain a line for the FIPA.

- To update only the FIPA for process **Proc2**:

```
#Item_FIPA%000029##Rev%A##Rule%Latest Working##Parent_FIPA%000031#
#Name%Proc2_fipa##Prod_ID%000013##Prod_Rev%A##Prod_Rule#
#Latest Working#
```

The file does not contain a line for the IPA.

- To update the FIPAs for multiple processes:

```
#Item_FIPA%000029##Rev%A##Rule%Latest Working#
#Parent_FIPA %000031%000034%000035#
#Name%Proc2_fipa%Proc5_fipa%Proc6_fipa#
#Prod_ID%000013##Prod_Rev%A##Prod_Rule##Latest Working#
```

The file does not contain a line for the IPAs.

- To update the FIPA for the top process:

```
#Item_FIPA%000029##Rev%A##Rule%Latest Working##Parent_FIPA%000029#
#Name%demoNested_fipa##Prod_ID%000013##Prod_Rev%A##Prod_Rule#
#Latest Working#
```

The file does not contain a line for the IPA.

- To update all the FIPAs present in the process structure:

```
#Item_FIPA%000029##Rev%A##Rule%Latest Working##Parent_FIPA%000029#
#Name ##Prod_ID%000013##Prod_Rev%A##Prod_Rule##Latest Working#
```

The file does not contain a line for the IPAs.

Note

If a value for the **#Name#** parameter is not present, then all the FIPAs under the **Parent_FIPA** process are updated. You must always include the **#Name#** parameter even if it has no value.

- To update the IPA and multiple FIPAs in the process structure:

```
#Item%000029##Rev%A##Rule%Latest Working##OG%OccurrenceGroup#
#Name%IPA -Jun 30, 2011 12:01:21 PM##Occ%MEConsumed##Proc%MEProcess#
#Item_FIPA%000029##Rev%A##Rule%Latest Working##Parent_FIPA
%000031%000034%000035#
#Name%Proc2_fipa%Proc5_fipa%Proc6_fipa##Prod_ID%000013##Prod_Rev%A#
#Prod_Rule##Latest Working#
```

Chapter

11 Classification utilities

icsutility

Imports the following types of Classification data:

- Class definitions.
This can optionally include the attributes, groups, parent classes, subclasses, or the class hierarchy from the root.
- Any individual attribute or Key-LOV object.
- Resource Manager resource assemblies.
This can optionally include root-level components, intermediate components, bottom-level components, and propagation start points. The hierarchical component positions are maintained.

icsutility also allows you to import class-specific and instance-specific files.

SYNTAX

icsutility[-u=*user-id* {-p=*password* | -pf=*password-file*} -g=*group*]

ARGUMENTS**-u**

Specifies the user ID.

This is generally **infodba** or another user with administration privileges. If this argument is used without a value, the operating system user name is used.

Note

If Security Services single sign-on (SSO) is enabled for your server, the **-u** and **-p** arguments are authenticated externally through SSO rather than being authenticated against the Teamcenter database. If you do not supply these arguments, the utility attempts to join an existing SSO session. If no session is found, you are prompted to enter a user ID and password.

-p

Specifies the password.

If used without a value, the system assumes a null value.

If this argument is not used, the system assumes the *user-ID* value to be the password.

This argument is mutually exclusive with the **-pf** argument.

-pf

Specifies the password file. If used without a value, the system assumes a null value. If this argument is not used, the system assumes the *user-ID* value to be the password.

For more information about managing password files, see [Manage password files](#).

This argument is mutually exclusive with the **-p** argument.

-g

Specifies the group associated with the user.

If used without a value, the user's default group is assumed.

-mod

Specifies the type of modification: **insert**, **update**, or **revise**.

insert

Creates a new ICO, item, and item revision. If an ICO or item with the corresponding name already exists, nothing is imported.

Attribute values are imported in the new ICO. Class-specific and instance-specific files (for example, **HPGL** files, **GIF** files, **JT** files, and **PRT** files) are attached to the new item revision.

-update

Updates the values in the ICO attached to the latest item revision (or to the item) and the assembly structure in the latest item revision (or item), as well as the class-specific and instance-specific files.

-revise

Creates a new item revision and ICO and imports the attribute values in the new item revision and ICO that classifies the new item revision. Class-specific and instance-specific files are attached to the new item revision.

You can specify an additional argument, **-forceConversion=1** (default value **0**). In this case, the relationship to the item is converted to a relationship from the ICO to the latest item revision.

-dbg

Specifies the debug level. Any positive integer number up to level **7**. **0** turns off debugging mode. The higher the debug level, the more detailed is the trace being output.

-del

Specifies the list of characters used to delimit multiple search directories and file extensions in the following file path and extension options:

-fid

Specifies the name of a folder into which imported items and datasets are placed. This folder is placed in the **Newstuff** folder.

-sfn

Specifies the name of the SML file to be processed. Do not include the **.sml** file extension.

-sfp

Specifies the path of the directory containing the **.sml** file. Paths must be terminated using the platform-specific path delimiter, as follows:

\ Windows delimiter

/ UNIX delimiter

-cfp

Specifies the directory path (or paths) containing the class-related files to be imported, such as class image files.

-cfe

Specifies the file extension of the class-related files to be imported. Accepts multiple file extensions separated by the delimiter set using the **-del** argument.

-ofp

Specifies the directory path (or paths) of the object-related files to be imported.

-ofe

Specifies the file extension of the object-related files to be imported. Accepts multiple file extensions separated by the delimiter set using the **-del** argument.

-ffp

Specifies the directory path containing the file types configuration file to be used.

-ffn

Specifies the name of the file types configuration file to be used, including the file extension.

-clr

If used, specifies that item revisions are classified when importing data. If not used, items are classified upon import.

When you specify the **-clr** argument and try to import an ICO that is already in the database and the item, but not the item revision, was classified, an error message is output when **insert** mode is used. When **update** mode is used, the relationship to the item is converted to a relationship from the ICO to the latest item revision.

-cit

Creates items of the specified item type, rather than the standard items.

ENVIRONMENT

This utility must be run in the Teamcenter shell environment.

FILES

As specified in *Log files produced by Teamcenter* and the **filetypeDefaults.txt** configuration file.

The entries in the **filetypeDefaults.txt** configuration file map the extensions of all imported files to specific Teamcenter data structures: GRM relationship type, dataset type, named reference, optional default tool, and optional subdirectory name. The mapping of file extensions to specific Teamcenter data structures can be configured per import directory/subdirectory combination.

RESTRICTIONS

You must have the corresponding privileges (create/modify) to start the import process. If you do not have the privileges, the data is not imported and a message is written to the log file.

Only the initial population of an empty Resource Manager database is fully supported.

EXAMPLES

To import the **C:\import\sml\test.sml** file using the file type configuration file **C:\import\config\filetypeDefaults.txt** together with **GIF** class image files residing in the **C:\import\class** directory and object-related **JT**, **GIF**, and Word files residing in the **C:\import\object** directory, enter the following command on a single line:

```
icsutility -u=smith -p=secret -g=admin
```



```
-mod=update -dbg=1 -del=, -fid=import01 -ffp=C:\import\config\  
-ffn=filetypeDefaults.txt -sfp=C:\import\sml -sfn=test  
-cfp=C:\import\class\ -cfe=gif -ofp=C:\import\object\ -ofe=jt,gif,doc
```

icsxml

Exports the following types of Classification data through XML files:

- Class definitions.

This can optionally include the attributes, subclasses, parent classes, or the class hierarchy from the root.

- Any individual attribute or Key-LOV object.

**INPUT FILE
FORMAT**

The format of the input and output file for the **icsxml** utility must comply with the XML standard. This standard defines five special characters used to structure the content. If these characters are included in the body of the XML file, they must be replaced, as shown in the following table.

Symbol	Replacement characters
&	&amp;
<	&lt;
>	&gt;
,	&apos;
“	&quote

The following example illustrates the use of the special XML characters:

```
<KeyLOV keyLOVId="-123451">
  <Name>Space & test</Name>
  <Values>
    <Key>01 22</Key>
    <Value>Value & & 01</Value>
  </Values>
</KeyLOV>
```

SYNTAX

```
[-u=user-id {-p=password | -pf=password-file} -g=group]
icsxml -import import-flags
icsxml -export export-flags
```

ARGUMENTS**-u**

Specifies the user ID.

This is generally **infodba** or another user with administration privileges. If this argument is used without a value, the operating system user name is used.

Note

If Security Services single sign-on (SSO) is enabled for your server, the **-u** and **-p** arguments are authenticated externally through SSO rather than being authenticated against the Teamcenter database. If you do not supply these arguments, the utility attempts to join an existing SSO session. If no session is found, you are prompted to enter a user ID and password.

-p

Specifies the password.

If used without a value, the system assumes a null value.

If this argument is not used, the system assumes the *user-ID* value to be the password.

This argument is mutually exclusive with the **-pf** argument.

-pf

Specifies the password file. If used without a value, the system assumes a null value. If this argument is not used, the system assumes the *user-ID* value to be the password.

For more information about managing password files, see [Manage password files](#).

This argument is mutually exclusive with the **-p** argument.

-g

Specifies the group associated with the user.

If used without a value, the user's default group is assumed.

-export

Exports the given subset of Classification data into an XML file.

-import

Imports Classification data from a given XML file into the Teamcenter database.

-update

Updates existing objects while importing the Classification data. If this argument is not used, the utility does not import data for existing objects. This argument is only valid in conjunction with the **-import** argument.

-file= *xml-file-path*

Specifies the physical path of the XML file.

For the **-export** option, the utility generates the given XML file.

For the **-import** option, the utility reads the given XML file.

-filter= *A, K, C, V, I, all*

Specifies the object types to be considered for the import/export operation. One or more of the following characters can be specified:

A Attributes

K Key-LOV

C Classes

V Views

I Instances

all All types of objects are accepted for import or export

If filter options are not specified, **all** is used as the default.

-class= *class-id*

Specifies the ID of the class definition object to be exported. This argument is valid only in conjunction with the **-export** argument.

-objtype= *A | K | C | I*

Specifies the type of object for which an ID is given using the **-objid** argument.

One the following values can be specified for the **-objtype** argument:

A	Attributes
K	Key-LOV
C	Classes
I	Instances

-objid= *object-id*

Specifies the ID of the object. Used in conjunction with the **-objtype** argument.

Note The **-objtype** and **-objid** arguments are valid only in conjunction with the **-export** argument.

-parent -parentviews -subclass -subclassviews -hierarchy

These arguments are valid only in conjunction with the **-export** and **-class** arguments. These options specify the associated objects to be exported with the specified class object.

-parent

Enables the export of parent classes.

-parentviews

Enables the export of the parent classes and associated views.

-subclass

Enables the export of associated subclasses.

-subclassviews

Enables the export of associated subclasses and views.

-hierarchy

Enables the export of the Classification hierarchy, from the root to the given class.

ENVIRONMENT

This utility should be run from a shell where the Teamcenter environment is set.

FILES

As specified in [Log files produced by Teamcenter](#).

RESTRICTIONS

None.

EXAMPLES

- To import Classification data from the **ics_data1.xml** file with the **update** option and to import objects of only the types **attribute**, **class**, and **view**, enter the following command on a single line:

```
icsxml -file=ics_data.xml -filter=ACV -import -update
-u=infodba -p=pwd -g=dba
```

- To import Classification data from the **ics_data1.xml** file without the update option, and to import all types of objects, enter the following command on a single line:

```
icsxml -file=ics_data.xml -filter=all -import  
-u=infodba -p=pwd -g=dba
```

- To export Classification data for the **ugc101** class into the **ugc101.xml** file with the option to export parent classes with associated views and subclasses, enter the following command on a single line:

```
icsxml -file=ugc101.xml -filter=all -export -class=ugc101  
-parentviews -subclass -u=infodba -p=pwd -g=dba
```

- To export Classification data for the **-2005** attribute into the **-2005.xml** file with the option to export the associated Key-LOV objects, enter the following command on a single line:

```
icsxml -file=-2005.xml -filter=AK -export -objtype=A  
-objid=-2005 -u=infodba -p=pwd -g=dba
```

ics_connect

Associates classification objects (ICOs) with workspace objects, based on item ID. You can list multiple workspace objects, or you can create a text file containing a list of item IDs.

For example, when you use the [smlutility](#) utility to import classification data into Teamcenter, the ICOs are created as specified in the input file, but they are not attached to the items defined in the input file via their workspace object **uid**. Use the **ics_connect** utility to associate all your ICOs (not already connected to any item) to the specified item of the same ID.

Caution If you use multifield keys to define item IDs, you cannot use **ics_connect** to connect a standalone ICO to an existing item because multiple items with the same ID exist.

SYNTAX

```
ics_connect [-u=user-id {-p=password | -pf=password-file} -g=group]
[-names=item1-ID,item2-ID,... | -file=file-name] [-h]
```

ARGUMENTS**-u**

Specifies the user ID.

This is generally **infodba** or another user with administration privileges. If this argument is used without a value, the operating system user name is used.

Note

If Security Services single sign-on (SSO) is enabled for your server, the **-u** and **-p** arguments are authenticated externally through SSO rather than being authenticated against the Teamcenter database. If you do not supply these arguments, the utility attempts to join an existing SSO session. If no session is found, you are prompted to enter a user ID and password.

-p

Specifies the user's password.

If used without a value, the system assumes a null value. If this argument is not used, the system assumes the *user-ID* value to be the password.

This argument is mutually exclusive with the **-pf** argument.

-pf

Specifies the password file. If used without a value, the system assumes a null value. If this argument is not used, the system assumes the *user-ID* value to be the password.

For more information about managing password files, see [Manage password files](#).

This argument is mutually exclusive with the **-p** argument.

-g

Specifies the group associated with the user.

If used without a value, the user's default group is assumed.

-names

Specifies the item IDs of the items to be associated with the ICOs of the same ID.

-file

Specifies the name of the file containing the list of item IDs to be associated with the ICOs. Place one item or item revision ID per line.

-h

Displays help for this utility.

ENVIRONMENT

As specified in [Manually configuring your environment for Teamcenter utilities](#).

FILES

As specified in [Log files produced by Teamcenter](#).

RESTRICTIONS

None.

EXAMPLES

- To associate the ICOs with IDs of **item2** and **item3** with the items with IDs of **item2** and **item3**:

```
ics_connect -u=infodba -p=infodba -g=dba -names=item2,item3
```

The system searches for items with the ID **item2** and **item3**, connecting them to ICOs with the same ID, as long as the ICOs are not attached to any other workspace object.

- To associate the ICOs with the **item4/A** and **item5/C** item revisions:

```
ics_connect -u=infodba -p=infodba -g=dba -names=item4/A,item5/C
```

The system searches for item revisions with the ID **item4/A** and **item5/C**, connecting them to ICOs with the same ID, as long as the ICOs are not attached to any other workspace object.

- To associate the ICOs with the item IDs contained in the **wso_names.txt** file:

```
ics_connect -u=infodba -p=infodba -g=dba -file=wso_names.txt
```

The file must contain item and item revision IDs, one per line.

smlutility

Updates shared Classification hierarchy definitions to all sites with which they are shared.

Use this command if you do not want to run the **subscriptionmgrd** daemon in the background. This utility can also be used to share specific definitions immediately, for example, if you modify a definition and want to share it with a colleague at a different site. In such cases, you can execute the command by specifying the definitions you want to share and the sites to which you want to update these definitions.

When you import classification data into Teamcenter using the **smlutility** utility, the ICOs that are created are not attached to the items defined in the input file using a workspace object UID. Use the **ics_connect** utility to associate all the ICOs not already connected to any item to the specified item of the same ID.

Caution The **smlutility** program overwrites all attributes when run in **-update** mode. This program does not support the update of individual attributes.

SYNTAX

```
smlutility [-u=user-id {-p=password | -pf=password-file} -g=group]
{[-import | -export | -delete | -list_hierarchy | -sync |
-reassign_to_rev | -ask_shared_sites | -add_shared_sites | -update_shared_sites |
-list_local_icos | -migrate | -update_unit_system | -lost_icos |
-list_invalid_class_ids | -repair_default_value | -h]}
```

ARGUMENTS**-u**

Specifies the user ID.

This is generally **infodba** or another user with administration privileges. If this argument is used without a value, the operating system user name is used.

Note

If Security Services single sign-on (SSO) is enabled for your server, the **-u** and **-p** arguments are authenticated externally through SSO rather than being authenticated against the Teamcenter database. If you do not supply these arguments, the utility attempts to join an existing SSO session. If no session is found, you are prompted to enter a user ID and password.

-p

Specifies the user's password.

If used without a value, the system assumes a null value. If this argument is not used, the system assumes the *user-ID* value to be the password.

This argument is mutually exclusive with the **-pf** argument.

-pf

Specifies the password file. If used without a value, the system assumes a null value. If this argument is not used, the system assumes the *user-ID* value to be the password.

For more information about managing password files, see [Manage password files](#).

This argument is mutually exclusive with the **-p** argument.

-g

Specifies the group associated with the user.

If used without a value, the user's default group is assumed.

-import

Imports definitions from the specified file into the database.

-insert

Appends the definitions to the existing definitions.

-update

Overwrites the existing definitions with those specified in the file.

-export

Exports definition data to a specified file. For example:

```
smlutility -export user password group file-name...[-select:object-selection]
[-objects:object-details]
```

object-selection

dictionary | keylov | class=cid[... | ::sid] | all

object-details

dictionary | keylov | class | data | hierarchy | all

-delete

Deletes definitions based on class, view, ICO ID, or attribute. For example:

```
smlutility -delete SA user SA password SA group
-class -id=class-ID [-icos] [-recurse]|
-view -id=class-ID [-icos]|
-ico (-cid=class-ID|-id=object-ID)|
-attribute -id=ID-number
-keylov
```

-class

Removes the identified class.

-icos

Removes all instances of the specified class.

-recurse

Removes all children of the specified class.

-view

Removes the specified view.

-icos

Removes all instances of the specified view if the view is a subclass.

-ico

Removes the identified instances (ICOs).

-attribute

Removes the specified attribute.

-keylov

Removes the specified key-LOV.

-list_hierarchy

Displays the classification hierarchy starting from the specified class.

For example:

```
smlutility -list_hierarchy [-u=SA user -p=SA password -g=SA group]
-id=class|groupID [-icos [-wso]] [-views]
[-attr] [-norecurse] [-nodescribe]
```

-icos

Displays basic information about the ICOs of each class.

-wso

Displays information about the classified workspace object.

-norecurse

Lists direct children only.

-nodescribe

Displays less information.

-sync

Synchronizes Classification definition data with remote sites, for example:

```
smlutility -sync SA-user SA-password SA-group [-definitions]
[-sites] [-verbose]
```

-definitions=

VIEW | CLASS | GROUP | DICTIONARY):ID [,ID[,...]]

VIEW:

Class-ID::View-ID

DICTIONARY:

Attribute-ID

CLASS:

Class-ID

KEYLOV:

KeyLOV-ID

GROUP:

Group ID

Note

You can specify multiple definitions. If no definition is specified, all objects are synchronized.

Examples

```
-definition=VIEW:myClass::myView,myClass::mySecondView
-definition=CLASS:myClass,mySecondClass,MyThirdClass
-definition=KEYLOV:-20000
```

-sites

Specifies the sites to which the data is updated. If you do not specify sites, synchronization takes place between all sites with which definitions are shared. For example:

```
-sites=site-name[,site-name[,...]]
```

Example

```
-sites=IMC-12345,IMC-56789
```

-reassign_to_rev

Modifies ICOs that classify an item so that they classify the latest item revision. For example:

```
smlutility -reassign_to_rev -u=SA-user -p=SA-password -g=SA-group class  
-id=class ID [-recurse]|view -id=view-ID|ico -id=object-ID
```

class

Identifies the class for which all ICOs will be reassigned. For example, using **class -id=ugc010101** indicates that all ICOs in the **ugc010101** class that classify an item will be changed so that they classify the latest revision of the item. You can use the **-recurse** option to reassign all ICOs of the class and all subclasses of the identified class. For example, using **class -id=ugc010101 -recurse** indicates that all ICOs in the **ugc010101** class that classify an item will be changed so that they classify the latest revision of the item. In addition to changing the ICOs of the class, the ICOs in all subclasses of the given class are changed as well.

view

Reassigns all ICOs of the identified view. For example, using **class -id=ugc010101 view -id=DefaultView** reassigns all ICOs in the **DefaultView** of class **ugc010101** so that the latest revision of the item is classified.

ico

Reassigns the identified Classification instances (ICOs). For example, using **ico -id=ugc010101_001** changes the specific ICO (**ugc010101_001**) so that it classifies the latest revision of the item.

General rules

If an ICO to be reassigned has no associated item or already classifies an item revision, the ICO is not reassigned.

When an ICO is reassigned, the ICO object ID is changed automatically. For example, ICO **ugc010101_001** changes to **ugc010101_001/A** if **A** is the latest revision of the item.

-ask_shared_sites

Returns the names of the sites to which the specified object is shared.

```
smlutility -add_shared_sites [-u=SA-user] [-p=SA-password] [-g=SA-group]  
-objectType={class | view | attribute | key-LOV} -objectID=object-ID
```

-objectType

Specifies the type of object whose site you want to know. You can request the site of a class, view, attribute or key-LOV.

-objectID

Specifies the ID of the object whose site you request. If you request a view's site, the ID has the following format:

class-ID::view-ID

-add_shared_sites

Shares a class, and optionally including its descendants and parents, to one or more sites.

```
smlutility -add_shared_sites  
-u=user -p=password -g=group -classes=class-ID -sites=site-name -options=options
```

-classes

classid[,classid[,...]]

-sites

sitename[sitename[,...]]

-options

option[,option[,...]]

shareChildClasses

Specifies to share child classes.

shareDefaultViews

Specifies to share default views.

shareSubclasses

Specifies to share subclasses.

shareSpecificViews

Specifies to share specific views. The **ICS_share_viewtypes** preference, which can be set to a combination of **user**, **group**, or **role**, is evaluated.

shareParents

Specifies that the parents of the classes named in the argument **classes** are also shared. If this option is not set, the share operation fails for classes whose parents are not shared to the selected site.

shareViews

Use to share default views, subclasses, and specific views, that is, specifying **shareViews** includes the **shareDefaultViews**, **shareSubclasses**, and **shareSpecificViews** options.

shareAll

Use to include the **shareViews**, **shareChildClasses**, and **shareParents** options.

For example:

```
smlutility -add_shared_sites -classes=myClass1,myClass2 -sites=Vienna  
-options=shareChildClasses,shareDefaultViews
```

-update_shared_sites

Changes the site name in all groups, classes, views, attributes, and key-LOVs when updating a site name in Multi-Site.

```
smlutility -update_shared_sites [-u=SA-user] [-p=SA-password] [-g=SA-group]
-oldSite=site-name -newSite=site-name
```

-oldSite

Specifies the original site name that is to be replaced by a new name.

-newSite

Specifies the new site name to replace the old one in all groups, classes, views, attributes, and key-LOVs.

-list_local_icos

Lists all ICOs that are owned locally.

```
smlutility -list_local_icos [-u=SA-user] [-p=SA-password] [-g=SA-group]
-cid=Class-ID [-verbose]
```

-cid

Specifies the unique ID of a classification class or group whose local ICOs should be listed.

-verbose

Displays additional information.

-migrate

Assigns the measurement system to each individual ICO within a class.

If an ICO previously exists within a solely metric or nonmetric class, the measurement system is not directly assigned to the ICO but is contained in the class definition. When you move from a metric or nonmetric class to one that contains both, Teamcenter assigns the measurement system to each individual ICO within the class.

```
smlutility -migrate [-u=SA-user] [-p=SA-password|-pf=password-file]
[-g=SA group] -cid=class-ID
[-dryrun] [-verbose={0|1|2}] [-remote]
```

-dryrun

Displays results of running the utility without making any changes and tests if all ICOs of the class can be accessed and changed.

-verbose

Displays additional information. The value for this argument must be one of the following:

- 0** No output
- 1** Output on error
- 2** Information and error output

-remote

Updates remote ICOs as well.

-update_unit_system

Changes a class or a class hierarchy that is previously metric or nonmetric to support both unit systems simultaneously.

```
smlutility -update_unit_system [-u=SA-user] [-p=SA-password] [-g=SA-group]
-cid=class-ID -unit=both [-recurse]
```

-cid

Specifies the class ID of the class required to support both unit systems.

-unit=both

Specifies that the class is changed to support both unit systems.

-recurse

Changes all child classes beneath the class specified by the **-cid** parameter to support both unit systems.

-lost_icos

Reports and recovers ICOs that do not have a class ID set. If, during creation, Classification crashes, an ICO can already be created, but the attribute specifying in which class it is to be classified not yet saved. This utility finds these unsaved ICOs and can, optionally, classify them in the specified class.

```
smlutility -lost_icos [-u=user-id {-p=password | -pf=password-file} -g=group]
[-cid=class-ID]
[-createClass]
[-recover]
[-reportfile=name-of-external-file-containing-results]
```

-cid

Specifies the ID of the class in which the recovered ICOs are to be classified. The ICOs are only classified in this class if **-recover** is also specified.

-createClass

Creates the specified class if it does not already exist.

-recover

Moves the found ICOs to the specified class. If this argument is not set, Teamcenter lists the ICOs in the command window or in a file, if you set the **-reportfile** argument.

-reportfile

Specifies the name of a file in which Teamcenter writes the names of all the ICOs found by the lost ICO search. If you do not use this argument, Teamcenter lists the results in the command window.

-list_invalid_class_ids

Lists all the group and class IDs in the database that contain invalid characters. By default, the following characters are invalid:

```
|%*:0{}[] \
```

You can specify a subset of these characters to be valid using the **ICS_allowed_chars_for_class_id** preference.

```
smlutility -list_invalid_class_ids [-u=user-id {-p=password |
-pf=password-file} -g=group]
[-reportfile=file-name]
```

-reportfile

Specifies the name of a file in which Teamcenter lists all the invalid group and class IDs. If you do not use this argument, Teamcenter lists the results in the command window.

-repair_default_value

Deletes default value for a view attribute if the corresponding class attribute's value is set to fixed.

-h

Displays help for this utility.

ENVIRONMENT

As specified in [Manually configuring your environment for Teamcenter utilities](#).

FILES

As specified in [Log files produced by Teamcenter](#).

RESTRICTIONS

This utility synchronizes shared definitions but does not change to which sites definitions are shared. If you specify a definition and site for synchronization, but the site is not one for which sharing has been designated, the definition is not sent to the site. For example, if the **myClass** class is only shared to the sites **Rome** and **London**, nothing is synchronized if you execute the following command:

```
smlutility -sync SA user SA password SA group -definition=CLASS:myClass
-sites=Paris
```

EXAMPLES

- To import Classification data from a file, perform the following steps:

- Enter the following command at a system command prompt:

```
smlutility -import user password group file -insert
```

For example:

```
smlutility -import infodba infodba dba in-CLASSexample.sml -insert
```

- Press the Enter key.

- To update existing Classification data, perform the following steps:

- Enter the following command at a system command prompt:

```
smlutility -import user password group file -update
```

For example:

```
smlutility -import infodba infodba dba in-CLASSexample.sml -update
```

- Press the Enter key.

- Perform the following steps to remove a class definition:

- Enter the following command at a system command prompt:

```
smlutility -delete user password group [class|view|instance] [id]
```

For example:

```
smlutility -delete infodba infodba dba class screws
```

2. Press the Enter key.

- Perform the following steps to remove a specific attribute with ID 1643:

1. Enter the following command at a system command prompt:

```
smlutility -delete user password group attribute -id=ID number
```

For example:

```
smlutility -delete infodba infodba dba attribute -id=1643
```

2. Press the Enter key.

- Perform the following steps to remove all unreferenced attributes:

1. Enter the following command at a system command prompt:

```
smlutility -delete infodba infodba dba attribute -id=*
```

2. Press the Enter key.

Sample files for smlutility and icsutility

To help you become familiar with the **sml** file format and the process of importing and exporting the data necessary to create your Classification hierarchy, a directory of sample files, along with a brief instructional page entitled **Readme.txt**, is included in your Teamcenter installation and can be found in the following location:

```
tcddata\sample\in-CLASS
```

You must install this directory using the Teamcenter Environment Manager.

Understanding the smlutility and SML file format

The **smlutility** program provides a variety of utilities related to Classification, including the capability to import and export the Classification hierarchy and related instances to and from an ASCII text format. These import/export features are useful for the bulk loading of Classification data and for transferring a Classification hierarchy tree from one database to another.

Although import/export data can be contained in one file or distributed over multiple files, the order in which the elements are imported is important. For example, attribute definitions can reference lists. Therefore, the lists must already be defined. Additionally, classes and subclasses make reference to attributes that must first be defined in the attribute (UNCT) dictionary. To avoid dependency problems, Siemens PLM Software recommends that the following elements be imported in the following order:

- List
- Attribute
- Class
- Subclass

- Data

Each line of the import/export file starts with a three-character keyword followed by the appropriate arguments. Each field of the command line is separated by the vertical bar (|) character.

For example:

```
Keyword | argument1| argument2| argument3|
```

The following table lists reserved characters, their purpose, and a description:

Character	Purpose	Description
! (exclamation mark)	Comment	All characters after the ! character are comments. You can place the ! symbol anywhere within the command line.
& (ampersand)	Line continuation	Continues the command on the following line. Place the & symbol at the end of a line.
(vertical bar)	Field separation	Separator for the arguments.

Elements of an import/export file

An import/export file contains the following elements:

File header	<p>A file header is not required for import files, however, it is recommended. The file header includes information that helps you identify the data contained within the file. The file header is information-only.</p> <p>The export function automatically generates a file header. The export file header contains information, such as the Teamcenter version number, user, node, database server, and creation date.</p>
List definitions	<p>A list of values is associated with an attribute, such that when a user clicks the down-arrow in the Attribute field, the system displays a list of valid values for that attribute. Teamcenter uses lists when an attribute has a finite set of legal values. The list is more efficient than manual entry because it eliminates the need for the user to memorize the valid values. Lists also enforce consistency.</p> <p>Lists are defined prior to the attribute definition and are associated with the attribute at the time of its definition.</p> <p>A list definition is composed of the menu definition (STV keyword) and one or more menu item definitions (STD keyword).</p>

Attribute definitions	<p>Attributes are placeholders for values that distinguish one instance of a class/subclass from another. For example, within the sheet metal screws subclass, the length, diameter, and thread attributes distinguish one screw from another. Attributes are defined prior to the class definition and are associated with the class at the time of its definition.</p> <p>Once defined, attributes, like lists, are stored in a dictionary. They are only defined once and can be used as many times as needed.</p> <p>Attribute definitions use the SMV keyword.</p>
Group definitions	<p>Groups are defined in the import/export file to organize a large set of classes. Groups are essentially classes with no attributes.</p> <p>Group definitions use the SML keyword.</p>
Class definitions	<p>The class section of the import/export file defines the class, associates the class with a group, and associates a list of attributes with the class.</p> <p>A class definition is composed of the class (SML keyword) and the association of one or more attributes using the STD keyword.</p>
Subclass definitions	<p>The subclass section of the file defines subclasses, associates each subclass with a class, and associates a subset of the class attributes with each subclass.</p> <p>A subclass definition is composed of the subclass (BLD keyword) and the association of one or more attributes using the BSM keyword.</p>
Instances	<p>Instances of Classification objects are normally created by classifying a Teamcenter object, but can also be imported or exported. This is useful when loading third-party part or tool libraries, copying a Classification scheme from one database to another, or performing bulk loads of existing data.</p> <p>Instances of Classification objects are created using the DAT keyword.</p>

SML import file and BOM line syntax

The following figure illustrates the SML file format required to support import of assembly structures.

```

DAT |mc0101_001|mc01|01|0| | |2005: 01
BOM |mc0101_001|mc01|01| &
  NO;          1;mc0101_002;mc01;01; 1;1; 0.0;0.0;0.0;90.0;-0.0;-0.0;| &
  NO.0;        2;mc0101_003;mc01;01; 1;1; -54.0;-84.0;0.0;-0.0;-0.0;90.0;| &
  NO.0.0;      6;mc0101_004;mc01;01; 1;1; -47.227;-77.227;0.0;-0.0;-0.0;-175.0;| &
  NO.0.0.0;    7;C1;;; 1;1; +0.0;+0.0;+0.0;+0.0;+0.0;+0.0;| &
  NO.0.1;      2;mc0101_005;mc01;01; 1;1; 10.0;-104.0;0.0;-0.0;-0.0;90.0;| &
  NO.0.1.0;    6;mc0101_006;mc01;01; 1;1; 3.121;-97.121;1.590;-0.0;-0.0;95.0;| &
  NO.0.1.0.0; 7;C2;;; 1;1; +0.0;+0.0;+0.0;+0.0;+0.0;+0.0;|

```

Assembly structure BOM line syntax example

As shown in this example, each line containing an assembly structure definition begins with a BOM tag. The three columns that follow the BOM tag contain the

record, class, and subclass identifiers of the database record that defines a resource assembly. The remaining columns contain information pertaining to different components of the assembly.

The record identifier is equivalent to the ICO-ID and the item ID.

The first entry encodes the position of the component within the hierarchical assembly structure.

The second entry defines the node type for the component. Supported node types are:

Node type value	Node type
1	Root Component
2	Intermediate Component
6	Bottom level Component
7	Propagation Start Point

For component nodes, the third, fourth and fifth entries contain the record, class, and subclass identifiers of the database record representing the component. For propagation start points (PSPs), the third entry contains the number of the propagation start point prefixed by the letter **C**. In this case, the fourth and fifth entries remain empty.

The sixth entry contains a component quantity field. Multiple components of the same type, for example **mc0101_002**, must be listed on subsequent lines, each indicating a quantity of one, which allows each line to define its own component transformation.

The seventh entry indicates whether the component graphics are to be displayed in the context of the Genius4000 XTM assembly. This value is not yet mapped to any attribute of the ICS data model.

The remaining entries, including and beyond the eighth entry, define the position of a component in terms of coordinates **x**, **y** and **z** as well as the orientation of the component using the Euler rotation angles **a**, **b** and **c**. This information generally applies only to a single component, thus diluting the interpretation of the component quantity field.

Import/export file example

```
#####
!
! Copyright (C) Siemens PLM Software
#####
! SML-IMPORT-EXPORT-HEADER
!-----
! File : in-class_example.sml
!-----
! Version : V1.0
! User : andre
! Node : kwnc16
! DB Server : iman
! Date : 06/07/2000 20:48:59
#####
!
! in-CLASS Classification Example
!
! This file includes the definition of
```

```

! - attributes
! - menus
! - groups
! - classes/subclasses
! - and some sample data records
!
#####
!
! ICM -> ICM Classification Root
! myhl -> My Highest Level
! myg1 --> Subgroup 1
! 2001 2002 2003 2004 2005 2006 2007
! mc01 --> My Class 1 Diam Thk Descr Angl Dir
! mc01:00 --> All Data X X X
! mc01:01 --> Subclass 1-01 X X X X
! mc01:02 --> Subclass 1-02 X X X X
!
! mc02 --> My Class 2 Diam Thk #Hls Descr Mat AnglDir
! mc02:00 --> All Data X X X
! mc02:01 --> C2 with material X X X X X
! mc02:02 --> C2 including all X X X X X X X
! mc02:03 --> C2 limited X X X
!
! myg2 --> Subgroup 2
! ...
!
#####
! ---- Attributes ----
!SMV| UNCT| Format F.2| Text-Description| Short-Text|Ref|Unit U.2|Fl|EX1|EX2|
SMV | 2001| 20307 20509| Diameter | Diam | |mm inch| 0| | |
SMV | 2002| 20408 20509| Thickness | Thickn. | |mm inch| 0| | |
SMV | 2003| 10002 0| Number Of Holes | Num Holes | | | 0| | |
SMV | 2004| 00080 0| Description | Descr. | | | 0| | |
SMV | 2005| -2005 0| Material | Mat | | | 0| | |
SMV | 2006| 20307 0| Angle | Angle | |degree | 0| | |
SMV | 2007| -2007 0| Direction | Dir | | | 0| | |
! ---- Popups ----
! Popup for Material
STV |-2005| Material Popup |
STD | 01 | Lexan |
STD | 02 | Alloy Steel |
STD | 03 | HS Steel |
STD | 04 | Aluminium |
STD | 05 | Copper |
! Popup for Direction
STV |-2007| Direction Popup |
STD | N | Neutral |
STD | L | Left |
STD | R | Right |
! ---- Groups ----
!SML| SMLID | Typ | Group |Description |ShortDesc|Graphic|Flags|EX1|EX2|
SML | myhl | SAM | ICM |My highest level | | | 2 | | |
SML | myg1 | SAM | myhl |Subgroup 1 | | | 2 | | |
SML | myg2 | SAM | myhl |Subgroup 2 | | | 2 | | |
! --- Classes ----
! Class mc01 : My Class 1 (UNIT: english) -----
!SML| SMLID | Typ | Group |Description |ShortDesc|Graphic|Flags|EX1|EX2|
SML | mc01 | ICM | myg1 |My Class 1 | | | 1 | | |
! Attributes for "My Class 1"
!SMD| UNCT | ID | min | max | Flags | EX1 | EX2 |
SMD | 2001 | D1 | | | 0 | | |
SMD | 2002 | TH | | | 0 | | |
SMD | 2004 | REM| | | 0 | | |
SMD | 2006 | A1 | | | 0 | | |
SMD | 2007 | DIR| | | 0 | | |
! Subclass 00 : All Data for "My Class 1"

```

```

!
!BLD| BLDID| Description |ShortDesc|Graphic|Flags|EX1|EX2|
BLD | 00 | All Data | | | 0 | | |
!
!BSM|SML-Id| min| max| Flag| TXT | Flag1| TXT1| Flag2| TXT2| EX1| EX2|
BSM | 2001 | | | 0 | | 0 | | 0 | | | |
BSM | 2004 | | | 0 | | 0 | | 0 | | | |
BSM | 2007 | | | 0 | | 0 | | 0 | | | |
! Subclass 01 : "My Class 1" Subclasses 01
!BLD| BLDID| Description |ShortDesc|Graphic|Flags|EX1|EX2|
BLD | 01 | My Subclass 1-01 | | | 0 | | |
!
!BSM|SML-Id| min| max| Flag| TXT | Flag1| TXT1| Flag2| TXT2| EX1| EX2|
BSM | 2001 | | | 0 | | 0 | | 0 | | | |
BSM | 2006 | | | 0 | | 0 | | 0 | | | |
BSM | 2004 | | | 0 | | 0 | | 0 | | | |
BSM | 2007 | | | 0 | | 0 | | 0 | | | |
! Subclass 02 : "My Class 1" Subclasses 02
!BLD| BLDID| Description |ShortDesc|Graphic|Flags|EX1|EX2|
BLD | 02 | My Subclass 1-02 | | | 0 | | |
!
!BSM|SML-Id| min| max| Flag| TXT | Flag1| TXT1| Flag2| TXT2| EX1| EX2|
BSM | 2001 | | | 0 | | 0 | | 0 | | | |
BSM | 2002 | | | 0 | | 0 | | 0 | | | |
BSM | 2004 | | | 0 | | 0 | | 0 | | | |
BSM | 2007 | | | 0 | | 0 | | 0 | | | |
! Class mc02 : My Class 2 (UNIT: metric) -----
!SML| SMLID | Typ | Group |Description |ShortDesc|Graphic|Flags|EX1|EX2|
SML | mc02 | ICM | myg1 |My Class 2 | | | 0 | | |
! Attributes for "My Class 2"
!SMD| UNCT | ID | min | max | Flags | EX1 | EX2 |
SMD | 2001 | D1 | | | 0 | | |
SMD | 2002 | TH | | | 0 | | |
SMD | 2003 | NH | | | 0 | | |
SMD | 2004 | REM| | | 0 | | |
SMD | 2005 | MAT| | | 0 | | |
SMD | 2006 | A1 | | | 0 | | |
SMD | 2007 | D2 | | | 0 | | |
! Subclass 00 : All Data for "My Class 2"
!
!BLD| BLDID| Description |ShortDesc|Graphic|Flags|EX1|EX2|
BLD | 00 | All Data | | | 0 | | |
!
!BSM|SML-Id| min| max| Flag| TXT | Flag1| TXT1| Flag2| TXT2| EX1| EX2|
BSM | 2003 | | | 0 | | 0 | | 0 | | | |
BSM | 2004 | | | 0 | | 0 | | 0 | | | |
BSM | 2007 | | | 0 | | 0 | | 0 | | | |
! Subclass 01 : "My Class 2" C2 with material
!BLD| BLDID| Description |ShortDesc|Graphic|Flags|EX1|EX2|
BLD | 01 | C2 with material | | | 0 | | |
!
!BSM|SML-Id| min| max| Flag| TXT | Flag1| TXT1| Flag2| TXT2| EX1| EX2|
BSM | 2001 | | | 0 | | 0 | | 0 | | | |
BSM | 2003 | | | 0 | | 0 | | 0 | | | |
BSM | 2004 | | | 0 | | 0 | | 0 | | | |
BSM | 2005 | | | 0 | | 0 | | 0 | | | |
BSM | 2007 | | | 0 | | 0 | | 0 | | | |
! Subclass 02 : "My Class 2" C2 including all
! (Attributes displayed in different order)
!BLD| BLDID| Description |ShortDesc|Graphic|Flags|EX1|EX2|
BLD | 02 | C2 including all | | | 0 | | |
!
!BSM|SML-Id| min| max| Flag| TXT | Flag1| TXT1| Flag2| TXT2| EX1| EX2|
BSM | 2004 | | | 0 | | 0 | | 0 | | | |
BSM | 2005 | | | 0 | | 0 | | 0 | | | |
BSM | 2001 | | | 0 | | 0 | | 0 | | | |

```

```

BSM | 2002 | | | 0 | | 0 | | 0 | | | |
BSM | 2006 | | | 0 | | 0 | | 0 | | | |
BSM | 2007 | | | 0 | | 0 | | 0 | | | |
BSM | 2003 | | | 0 | | 0 | | 0 | | | |
! Subclass 03 : "My Class 2" C2 limited
!BLD| BLDID| Description |ShortDesc|Graphic|Flags|EX1|EX2|
BLD | 03 | C2 limited | | | 0 | | | |
!
!BSM|SML-Id| min| max| Flag| TXT | Flag1| TXT1| Flag2| TXT2| EX1| EX2|
BSM | 2003 | | | 0 | | 0 | | 0 | | | |
BSM | 2004 | | | 0 | | 0 | | 0 | | | |
BSM | 2007 | | | 0 | | 0 | | 0 | | | |
! ---- Data Records ----
! Class 1
DAT |mc0101_001|mc01|01|0| | |2001: 3.75 |2004:My sample record |2007:L
DAT |mc0101_002|mc01|01|0| | |2001:15.5 |2004:Please remind me
DAT |mc0101_003|mc01|01|0| | |2001:85 |2004:Right oriented |2007:R
DAT |mc0101_004|mc01|01|0| | |2001: .123|2006: 90.0 |2007:L
DAT |mc0101_005|mc01|01|0| | |2006:180 |2007:N
DAT |mc0101_006|mc01|01|0| | |2001:98.76 |2006:270 |2007:L
DAT |mc0102_001|mc01|02|0| | |2001: .01 |2004:Small data |2002:2.5
DAT |mc0102_001|mc01|02|0| | |2001: 0.333|2004:Negative Diameter|2002:4
DAT |mc0102_001|mc01|02|0| | |2001:20 |2002:8.8
DAT |mc0102_001|mc01|02|0| | |2007:L |2004:Big Part |2002:13.73
! Class 2
DAT |lexan_01 |mc02|01|0| | |2001: 4.0 |2003: 4 |2005:01 |2007:L
DAT |lexan_02 |mc02|01|0| | |2001: 4.0 |2003: 6 |2005:01 |2007:L
DAT |lexan_03 |mc02|01|0| | |2001: 4.0 |2005:01 |2007:R
DAT |lexan_04 |mc02|01|0| | |2001: 4.0 |2003: 4 |2005:01 |2007:R
DAT |lexan_05 |mc02|01|0| | |2001: 4.0 |2003:18 |2005:01 |2007:L
DAT |copper_01 |mc02|01|0| | |2001: 4.0 |2003: 4 |2005:05 |2007:L
DAT |copper_02 |mc02|01|0| | |2001: 4.0 |2003: 6 |2005:05 |2007:L
DAT |copper_03 |mc02|01|0| | |2001: 2.0 |2005:05 |2007:L
DAT |copper_04 |mc02|01|0| | |2001: 2.0 |2003: 4 |2005:05
DAT |copper_06 |mc02|01|0| | |2001: 1.0 |2003:18 |2005:05 |2007:N
DAT |alu_01 |mc02|01|0| | |2001:18.1 |2003: 7 |2005:04 |2007:R
DAT |alu_02 |mc02|01|0| | |2001:20.2 |2003: 6 |2005:04 |2007:L
DAT |alu_03 |mc02|01|0| | |2001:21.3 |2005:04 |2007:N
DAT |mc0202_001|mc02|02|0| | |2001:123.456 &
|2002:78.905 &
|2003: 13 &
|2005: 02 &
|2006: 45 &
|2007: N &
|2004: Keep in stock !
DAT |mc0203_001|mc02|03|0| | |2003: 2 |2004:Example 001 |2007:L
DAT |mc0203_002|mc02|03|0| | |2003: 4 |2004:Example 002 |2007:R
DAT |mc0203_003|mc02|03|0| | |2003: 8 |2004:Example 003 |2007:R
DAT |mc0203_004|mc02|03|0| | |2003: 16 |2004:Example 004 |2007:N
DAT |mc0203_005|mc02|03|0| | |2003: 31 |2004:Example 005 |2007:N
DAT |mc0203_006|mc02|03|0| | |2003: 62 |2004:Example 006 |2007:N
DAT |mc0203_007|mc02|03|0| | |2003: 88 |2004:
DAT |mc0203_008|mc02|03|0| | |2003: 56 |2004:Example 008 |2007:L
DAT |mc0203_009|mc02|03|0| | |2003: 39 |2004:Example 009 |2007:N
DAT |mc0203_010|mc02|03|0| | |2003: 27 |2004:Example 010 |2007:L

```

smlutility keyword syntax and description

STV and STD keywords (lists)

The **STV** keyword defines a list. Lists associated with an attribute provide a method by which the Classification administrator defines a list of values for a given attribute.

One or more **STD** statements defining the list of values for the list follow the **STV** statement defining the list.

Element name	Definition	Description
STV	Keyword	
STXT-ID	List ID number	Always negative.
Name	List title	

Element name	Definition	Description
STD	Keyword	
Key		Key for list.
Value	Text for list item	

The following example creates a list to be used with a **Cut Direction** attribute. The resulting list contains the **Left Hand Climb** and **Right Hand Climb** options.

Keyword	STXT-ID	Name
STV	-9110	Cut Direction

Keyword	Key	Value
STD	LHC	Left Hand Climb
STD	RHC	Right Hand Climb

SMV keyword (attributes)

The **SMV** keyword defines an attribute and adds the attribute definition to the attribute (UNCT) dictionary. Once an attribute is defined in the dictionary, it can be used and reused as needed.

Element name	Definition	Description
SMV	Keyword	
UNCT	Attribute identifier	Uniquely identifies an attribute and is the key field for associating the attribute with a class or subclass. UNCT numbers can be positive or negative.

Note Positive numbers from 0 to 999 are reserved for Siemens PLM Software.

Element name	Definition	Description
Format	Value format	<p>Defines the type of value that is stored for the attribute. The major types of values are:</p> <ul style="list-style-type: none"> Integer Real Date Time List <p>Zero, one, or two formats can be entered in the Format field. For more information about units and formats, see Units and formats.</p>
Name	Attribute name	Describes the attribute that is being defined. The name can be a maximum of 63 alphanumeric characters in length and is case sensitive.
Short Name	Short name	Short name for an attribute can consist of up to 10 alphanumeric characters. When there is limited space available, Teamcenter uses the short name for creating reports.
Ref		Not used.
Units	Units	<p>Units are entered as a string that appear after the attribute value in the Classification pane located on the Properties tab.</p> <p>Zero, one or two unit descriptions can be entered in the Units field. For more information about units and formats, see Units and formats.</p>
Flags		Not used, always 0 .
EX1		Not used.
EX2		Not used.

Units and formats

To identify the units and control the format of a numeric attribute, use the **Units** and **Format** parameters of the **SMV** command line.

For numeric attributes that change between measuring systems dependent on the class to which they are assigned, you can enter two formats and two units.

For example, if the numeric attribute **diameter** is used in the **inch** class and **metric** class, the following definition can be used:

```
SMV |-2491 |21309 21308| Diameter | Dia | | mm inch | 0 | |
```

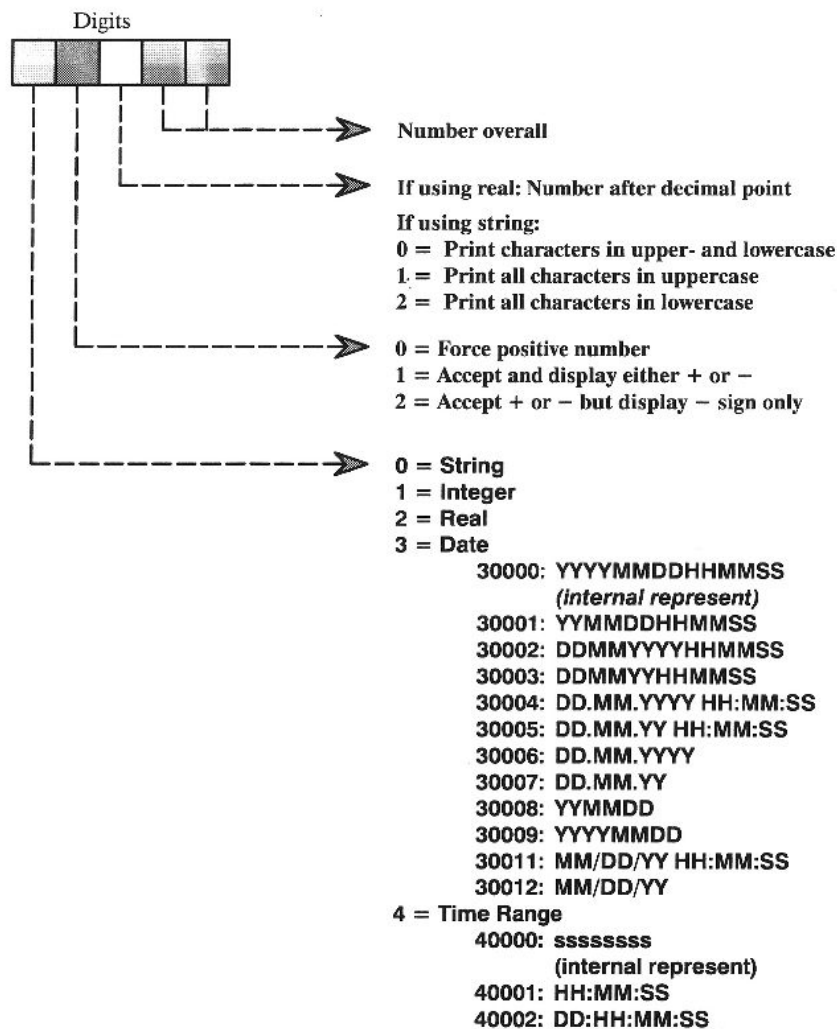

This definition describes a **+/-6.3** format with units of **mm** for the **metric** class and a **+/-3.3** format with units of **inch** for the **inch** class. For example:

For + or - REAL(6.3) use 21311. Where 11 is the sum of: (6 plus 3=9) plus .=10 plus - which gives 11.
 For a forced pos. REAL(6.3) use 20310. Where (6 plus 3 plus .)=10.
 For a Real(3.3) that forces pos. numbers, use the code 20307.
 For a Real(6.3) that forces pos. numbers, use the code 20310.

The value in the **Flags** parameter of the **SML** (class) command determines which of the two definitions are used with that class.

Note Negative format numbers are the STXT-ID of a list.

The first digit of a positive format number defines what types of values are expected in the field and controls the format of the value. The format number can be up to five digits in length. The following figure explains the meaning of the digit in the format number:



Digit in format number

In the following tables, the system creates a list with two entries and then defines two **Cut Direction** attributes. The first **Cut Direction** attribute (UNCT number

3001) represents a string attribute that allows the user to enter a 40-character text string representing the cut direction. In this case, the user is responsible for entering valid strings for the attribute value within the Classification interface.

In the second attribute definition (UNCT number 3002) creates a **Cut Direction** attribute that allows the user to choose predefined values from a list.

Note The negative format number matches the STXT-ID of the list definition and associates the attribute to the previously defined list.

List definition

Keyword	STXT-ID	Name
STV	-3002	Direction
Keyword	Key	Value
STD	L	Left
STD	R	Right

Cut direction attribute definition

Keyword	UNCT	Format	Name	Short Name	Ref	Units	Flags
SMV	3001	00040 0	Cut Direction	Cut Dir			0
SMV	3002	-3002 0	Cut Direction	Cut Dir			0

SML and SMD keywords (class/group)

The **SML** keyword defines a class or group. A class definition consists of an **SML** definition statement followed by up to 60 **SMD** lines to associate attributes with the class. A group is essentially a class with no attributes; the group definition consists of an **SML** statement only with the **Flags** parameter set to **2**.

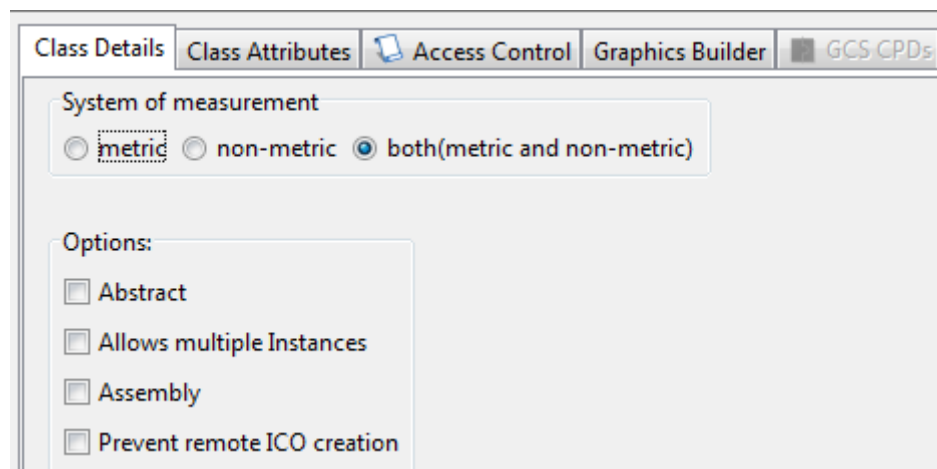
Note The attributes are associated with the class because they immediately follow the **SML** statement. There is no explicit reference to the class in the **SMD** statements.

Element name	Definition	Description
SML	Keyword	
SML-ID	Class or group ID	Internal unique class/group identifier.
Type	Module name	
Group	Parent class or group	For class definitions, the Group parameter defines the ID of the class or group that the current class or group belongs to in the hierarchy. You can only nest a group underneath another group. You can add a class to a group or to another class.

Element name	Definition	Description
Description	Class name	Name displayed in the Classification user interface.
Short Description	Short name	
Graphics File	Associated image file name	
Flags	Special case flags	Flags denote the options that you set when creating a class or group. For more information, see Understanding SML flags .
EX1		Not used.
EX2		Not used.

Understanding SML flags

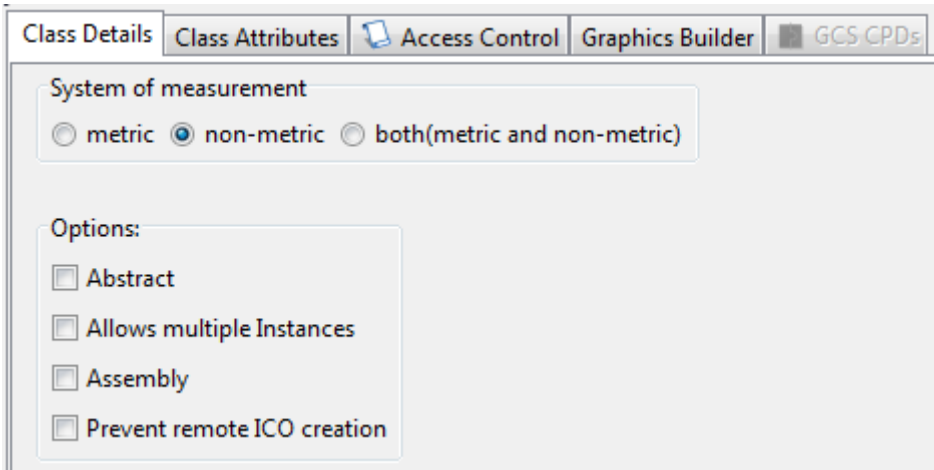
Flags denote the following options that you set when creating a class or group using the **smlutility** utility.



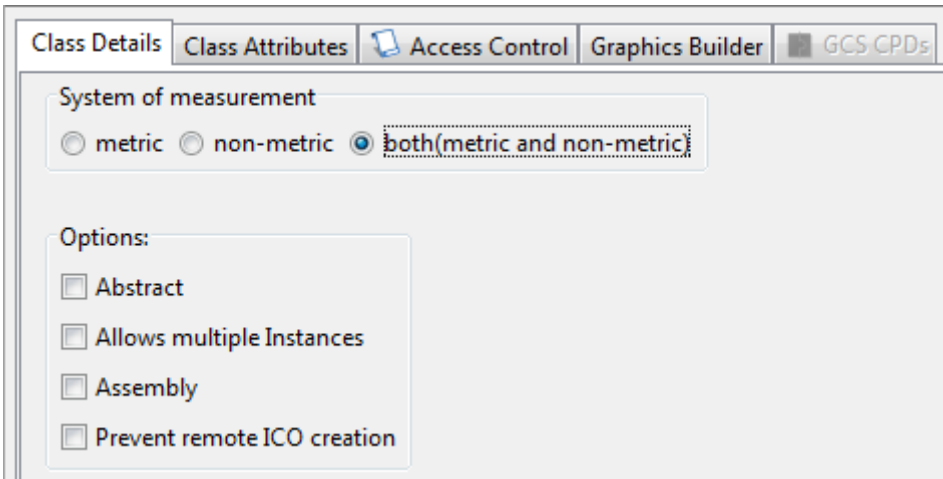
You can set these flags using specific integers in the import files.

Flag	Description
0	Class with metric unit system
1	Class with nonmetric unit system
2	Group definition
4	Class with both unit systems (metric and nonmetric)
16	Storage class (Abstract option not selected)
32	Assembly class
512	Class that prevents remote classified object creation (Prevent remote ICO creation selected)

You can also specify any meaningful combination of these options by adding the flags. For example, **17** denotes a nonmetric storage class (**16** + **1**). This is equivalent to setting the following options.



Specifying a flag of **20** denotes a storage class containing both systems of measure (**16** + **4**) and is equivalent to setting the following options.



SMD keyword (class attributes)

The **SMD** keyword associates attributes with a class. The class definition must associate all of the attributes that will be used by any subordinate subclass. A subclass can only associate attributes that have been previously associated with the parent class.

Element name	Definition	Description
SMD	Keyword	
UNCT	Unique attribute ID	Specifies the ID of an attribute defined in the UNCT dictionary. It associates the attribute with the class. For more information, see SMV keyword (attributes) .

Element name	Definition	Description
Standard Designation	Standard dimensional designation DIN or ASCII standard designation for attribute	The Standard Designation and Attribute Name (from the attribute definition) appear next to the respective attribute value field in the Classification pane located on the Properties tab.
Min	Minimum numeric value	
Max	Maximum numeric value	
Flags		Not used, always 0 .
EX1		Not used.
EX2		Not used.

For example, the following lines define the **Fasteners** group (essentially a class without any attributes), add the **Nuts01** class, and associate the **ID**, **OD**, and **THRD** predefined attributes with the class:

```
SML | Fasteners | FSTNRS | | Fasteners | FSTNRS | | 2 | | | !Fasteners Group
SML | Nuts01 | NT | Fasteners | Nuts | Nts | nuts.gif | 0 | | | !Nuts Class
SMD | -25011 | ID | | | 0 | | | !ID attribute
SMD | -25011 | OD | | | 0 | | | !OD attribute
SMD | -25011 | Thread | | | 0 | | | !Thread attribute
```

BLD and BSM keywords (subclass)

The **BLD** keyword defines a subclass. A subclass definition consists of a **BLD** definition statement followed by up to 60 **BSM** lines that associate attributes with the subclass. The selected attributes must be a subset of the class attributes.

Note A subclass is similar to a view of a class. It is comprised of a subset of the attributes used to define the class. The order of the attributes does not have to be the same as the order of the attributes at the class level. Classification displays the attributes in the order defined by the subclass. The attributes are associated with the class because they immediately follow the **BLD** statement. There is no explicit reference to the class in the **BLD** statements.

Element name	Definition	Description
BLD	Keyword	
BLD-ID	Subclass ID	Internal unique subclass identifier.
Description	Description	Subclass name displayed in the Classification user interface.
Short Description	Short description	
Graphics File	Graphics file	
Flags		Not used, always 0 .
EX1		Not used.
EX2		Not used.

Element name	Definition	Description
BSM	Keyword	
UNCT	Attribute ID	Specifies the ID of an attribute defined in the UNCT dictionary. It associates the attribute with the subclass. The list of attributes assigned to a subclass must be a subset of the attribute list defined for the class.
Min	Minimum numeric value	
Max	Maximum numeric value	
Flags		Not used, always 0.
TXT		Not used.
Flags1		Not used, always 0.
TXT1		Not used.
Flags2		Not used, always 0.
TXT2		Not used.
EX1		Not used.
EX2		Not used.

For example, the following lines define the **HexNuts01** and **SquareNuts01** subclasses and associate a predefined attribute named ID:

```
BLD|00      |Nuts   |Fasteners|nuts.gif|Nts|nuts.gif| 0 | | |!Subclass
BLD|01      |HXNuts|Fasteners|nuts.gif|Nts|nuts.gif| 0 | | |!Subclass
BSM|-25011|ID      |         |         |0 |         | | |!ID attribute
```

DAT keyword (instances)

The **DAT** keyword represents a Classification instance. Each **DAT** statement represents one instance.

An instance is a set of attribute values corresponding to an attribute list that defines a particular subclass. Any number of instances can exist for a specific subclass. Along with a relation to a Teamcenter object (for example, item, item revision, and dataset), the system creates a complete classification. The import/export function of the **smlutility** provides the ability to import or export Classification instances.

Element name	Definition	Description
DAT	Keyword	
DATA-ID	Instance ID	
SML ID	Class ID	
BLD ID	Subclass ID	
Flags		Not used, always 0.
POM-TAG	E Refs & I Refs	

Element name	Definition	Description
UNCT : Value	UNCT code : Value pair	Repeated for each attribute.

For example, the following statement defines the **ucg010101_001** Classification instance that belongs to the **Subclass 01** subclass of the **ugc0101** class. The 13 attribute values are entered for this subclass.

Attribute values are stated in terms of pairs of attribute ID and values separated by a colon (:) character as follows:

```
DAT | ugc010101_001 | ugc0101 | 01 | 0 | | &
| -2605: 001.500 | -2603: 04 | -2619: R | -2503: TMc0_00006 &
| -18032: HSS-Co5-TiN | -2653: 000.000 | -2637: 050_000 &
| -2618: 006.000 | -4110: 006.000 | -4110: 006.000 | -4100: B | -2690: 0 | -2691: 1 &
| -1200: Parallel shank cutter
```

When creating a classification instance in a class that contains both metric and nonmetric units of measure, you must specify the unit of measure for the instance. Do this by adding **-630:0** for a metric instance and **-630:1** for a nonmetric instance to the attribute/value list.

Chapter

12 Query utilities

build_fts_index

Builds keyword indexes for the Autonomy search engine on an object-by-object basis. These indexes enable the Teamcenter full-text keyword search and can index both the properties of Teamcenter dataset objects and the contents of dataset files.

- If a dataset file is not of a document type supported by Autonomy, the utility invokes a user-specified filter program to convert the file to a supported format.
- You do not need to specify languages to index. The Autonomy software automatically detects the language used in a file, and Teamcenter determines the metadata language.
- A file attached to a dataset must contain enough words for the Autonomy software to detect the language.

Note Before running this utility, add the **TC_fts_indexed_types** preference to the database. This preference defines a list of the dataset types that you want to index. For more information about managing options and preferences, see the *Rich Client Interface Guide*.

For more information about all the preferences required to index the Autonomy search engine, see the *Application Administration Guide*.

Autonomy supports a wide range of file types for word processing, spreadsheet, and presentation graphics applications.

**WORD
PROCESSING**

The following word processing file types are supported:

- HTML
- SGML
- XML
- TEXT
- RTF
- WML
- Adobe PDF
- ASCII text
- ANSI text
- Unicode V2.x
- Microsoft RTF
- Microsoft Word for Windows V3.x and later

- Microsoft Word Mac V4.x to V6.x
- Microsoft Word PC V2.0 to V5.5
- Microsoft Word 2007 (MSWordX)
- Quark QXD

SPREADSHEET

The following spreadsheet file types are supported:

- Microsoft Works V3.x and later
- Microsoft Excel V3.x and later
- Microsoft Excel 2007 (MSEExcelX)

PRESENTATION
GRAPHICS
FORMATS

The following presentation graphics file formats are supported:

- Shockwave Flash (with Autonomy Flashslave)
- Microsoft PowerPoint V4 and later
- Microsoft PowerPoint 2007 (MSPowerpointX)

SYNTAX

build_fts_index.exe [-u=*user-id* | -p=*password* | -pf=*password-file*] -g=*group*]
 | -type=*variable_name* | -ext=*extension* | -filter=*/path/scriptfile* |
 -workdir=*work_directory* | -db=*Autonomy_database* |
 -filenumber=*number_of_files* | -log=*logfile* | -f=[**index** | **delete** |
update] | -maxresults=*number_of_queries* | -query=*query_name* |
 -entry=*entry_name* | -value=*entry_name_value* | -report

ARGUMENTS

-u

Specifies the user ID.

This is generally **infodba** or another user with administration privileges. If this argument is used without a value, the operating system user name is used.

Note

If Security Services single sign-on (SSO) is enabled for your server, the **-u** and **-p** arguments are authenticated externally through SSO rather than being authenticated against the Teamcenter database. If you do not supply these arguments, the utility attempts to join an existing SSO session. If no session is found, you are prompted to enter a user ID and password.

-p

Specifies the password.

If used without a value, the system assumes a null value.

If this argument is not used, the system assumes the *user-ID* value to be the password.

This argument is mutually exclusive with the **-pf** argument.

-pf

Specifies the password file. If used without a value, the system assumes a null value. If this argument is not used, the system assumes the *user-ID* value to be the password.

For more information about managing password files, see [Manage password files](#).

This argument is mutually exclusive with the **-p** argument.

-g

Specifies the group associated with the user.

If used without a value, the user's default group is assumed.

-type

Specifies the dataset type to index. Specified dataset types must be defined by the [TC_fts_indexed_types](#) preference.

If this argument is not specified, all dataset types defined by the [TC_fts_indexed_types](#) preference are indexed.

This argument can be specified multiple times to index multiple dataset types in a single utility session, for example:

-type=Text -type=HTML

Note

If the [TC_fts_indexed_types](#) preference is used, the **WSOM_find_list_separator** value must be set in the **tc_env** file.

-ext

Specifies the extension pattern of the files to be indexed. This argument can be specified multiple times to index multiple file types in a single session. This argument can contain wildcard characters.

If not specified, the default value is an asterisk (*), and all files associated with datasets are indexed.

-filter

Specifies a filter program requiring that two arguments, input file and output file, be specified. The input file is read and its contents are output in a file format that is supported by Autonomy. By default, no filter is applied.

Note

The **-filter** argument must be entered immediately following the **-ext** argument to which it applies.

-workdir

Specifies the full path to the directory under which an **autonomy** subdirectory is created to store all exported dataset files and any immediate files to be indexed by Autonomy.

- This directory must have large scratch spaces to support indexing of a large number of datasets in a single run.
- All exported files and any immediate files are removed from the directory after the utility is run.
- The user must have the write privilege into the work directory.

If this argument is not specified, the default is the current directory.

-db

Defines the Autonomy database where index data is stored.

If this argument is not specified, the data is stored in the database defined by the **TC_fts_database_name** preference.

-filenumber

Defines the number of files that can be exported and stored in the directory before being indexed in to Autonomy. Use this argument to allow the utility program to index datasets at intervals without consuming large amounts of disk space.

If this argument is not specified, the default file number is **100**.

-log

Specifies the name of the file into which the import statistics are written. The file is created and stored in the **TC_TMP_DIR/fts** directory, with the Teamcenter process ID appended to it.

The log file provides the following information:

- Number of calls made to Autonomy during the utility run
- Number of datasets found
- Number of datasets indexed
- Number of datasets that failed to be indexed and the corresponding failure messages

If this argument is not specified, the default log file is **tc_index_processId.log**.

-f

Specifies one of the following operations:

- **index**
Indexes datasets for all languages.
- **delete**
Deletes invalid index entries from the Autonomy database.
This option does not support the **-query** argument.
- **update**
Deletes invalid index entries, if applicable, and regenerates new entries if datasets have not yet been indexed. This option supports the cases in which datasets have either been modified or created after the last index or were not indexed. Siemens PLM Software recommends using the **-update** option once the entire database has been indexed using the **-index** option.

If this argument is not specified, the default operation is **f=index**.

-maxresults

Defines the number of query results that can be returned by the search engine.

This option is useful if you want to control how often to place query calls to the Autonomy server and allow the utility to process invalid index entries at intervals without consuming large amounts of memory. For example, use **-maxresults** to control how often to make query calls to Autonomy server during the **-f=delete** operation.

Note This option works only in conjunction with the **-f=delete** argument.

If this argument is not specified, the default maximum number is **5000**.

-query

Defines the name of the Teamcenter saved query that can be run to find all dataset objects that need to be indexed.

This argument is useful when you select a few individual datasets to be indexed with special options, such as a different language type.

- Datasets returned by the query are indexed only if their dataset types are defined by the **TC_fts_indexed_types** preference.
- Invalid dataset types are ignored.

Note When used with the **-f** argument, **-query** works only with the **-f=index** and **-f=update** options.

-entry

Specifies the user entry name for the saved query for which the search value is being defined.

-value

Specifies the value corresponding to the entry name.

Note The **-entry** and **-value** arguments must be supplied in pairs.

-report

Prints all the datasets that are reindexed when the utility is run using the **-update** option. The **-report** argument can be used to report datasets that were not indexed during the last indexing run.

ENVIRONMENT

As specified in *Manually configuring your environment for Teamcenter utilities*.

FILES

As specified in *Log files produced by Teamcenter*.

RESTRICTIONS

None.

EXAMPLES

- The following example indexes all dataset types defined by the **TC_fts_indexed_types** preference:

```
$TC_BIN/build_fts_index  
-u=user-name -p=password -g=group-name\  
-filenumber=500
```

Note If a query is not included, each use of the **build_fts_index** utility indexes all datasets.

- The following example indexes **Text** type datasets. Only the dataset files with **txt** extensions are indexed.

```
$TC_BIN/build_fts_index
-u=user-name -p=password -g=group-name\
-type=Text
-filenumber=500
```

- The following creates index entries in the Autonomy database for dataset objects modified after the specified date (in this case, *11-Aug-2011*):

```
$TC_BIN/build_fts_index
-u=user-name -p=password -g=group-name\
-query="Dataset..."
-entry="Modified After"
-value="11-Aug-2011"
```

Note The query value must exactly match a name value as shown in Query Builder.

For information about using Query Builder, see the [Query Builder Guide](#).

- The following example deletes invalid index entries from the Autonomy database for all dataset types defined by the **TC_fts_indexed_types** preference:

```
$TC_BIN/build_fts_index
-u=user-name -p=password -g=group-name\
-f=delete
```

- The following example deletes invalid index entries from the Autonomy database for **Text** type datasets:

```
$TC_BIN/build_fts_index
-u=user-name -p=password -g=group-name\
-f=delete
-type=Text
```

default_queries

Reinstalls one or more of the default query forms. When you run the utility, it searches for these default query forms and automatically reinstalls any that have been deleted. If a default query form has become corrupted, you must delete it from the database before running this utility.

This utility can operate in one or more locales provided the locales are supported by the encoding of the **TCServer** machine. To set the correct locale, use the [preferences_manager](#) utility to set the **TC_language_default** preference as shown in the following example:

```
preferences_manager -u=infodba -p=password -g=dba
-mode=import -scope=SITE -preference=TC_language_default
-values=xx -action=OVERRIDE
```

xx is one of the supported locales.

For more information about the **TC_language_default** preference and a list of supported locales, see the *Preferences and Environment Variables Reference*.

Caution Siemens PLM Software recommends that you run this utility as the default Teamcenter system administration user (**infodba**). This ensures that the query forms are protected from unauthorized modification by other users because they are owned by the **infodba** user.

SYNTAX

```
default_queries [-u=user-id {-p=password | -pf=password-file} -g=group]
-locales=locale-code | ALL [-recreate] [-validate_query_name]
[-validate_query_descs] [-modify_queriesquery_name(s) | ALL] [-v] [-h]
```

ARGUMENTS

-u

Specifies the user ID.

This is generally **infodba** or another user with administration privileges. If this argument is used without a value, the operating system user name is used.

Note If Security Services single sign-on (SSO) is enabled for your server, the **-u** and **-p** arguments are authenticated externally through SSO rather than being authenticated against the Teamcenter database. If you do not supply these arguments, the utility attempts to join an existing SSO session. If no session is found, you are prompted to enter a user ID and password.

-p

Specifies the password.

If used without a value, the system assumes a null value.

If this argument is not used, the system assumes the *user-ID* value to be the password.

This argument is mutually exclusive with the **-pf** argument.

-pf

Specifies the password file. If used without a value, the system assumes a null value. If this argument is not used, the system assumes the *user-ID* value to be the password.

For more information about managing password files, see [Manage password files](#).

This argument is mutually exclusive with the **-p** argument.

-g

Specifies the group associated with the user.

If used without a value, the user's default group is assumed.

-locales

Specifies the locale, using locale codes or **ALL**, for which translated query names and descriptions are installed. You can specify a single locale or you can specify multiple locales in a comma-separated list, for example **en_US,de_DE,fr_FR**. Using the **ALL** value installs all locales supported by your Teamcenter system.

For a list of locale codes, see the [Localization Guide](#).

-recreate

Optional parameter. Recreates the default query.

-validate_query_name

Optional parameter. Validates the query name does not exceed the maximum length.

-validate_query_descs

Validates the query description does not exceed the maximum length.

-modify_queries

Updates the query clauses of the specified queries with the default query clause. You can specify a single query or you can specify multiple queries in a comma-separated list, for example, ItemRevision,Item,Dataset. Use the **ALL** value to modify all your queries.

-v

Specifies verbose mode.

-h

Displays help for this utility.

ENVIRONMENT

As specified in [Manually configuring your environment for Teamcenter utilities](#).

FILES

As specified in [Log files produced by Teamcenter](#).

RESTRICTIONS

None.

EXAMPLES

None.

find_appearances

Queries the database for information about appearances. Arguments can be specified to restrict the search, as follows:

- Search for a specific appearance set or all appearance sets.
- Search for appearances configured on a given date.
- Search for appearances configured for a particular unit number.
- Search for the appearance associated with a specific component by item ID, key, or other attribute.

Note The output of this utility can be configured to print effectivity values.

SYNTAX

```
find_appearances [-u=user-id {-p=password | -pf=password-file} -g=group]
[-item_id=item-id] | [-key=[keyAttr1=keyVal1][,keyAttr2=keyVal2...] [,keyAttrN=keyValN]]
[-config_rule=config-rule] [-view_type=view-type] [-root_only]
[-date=date | now | today] [-unit_no=unit-number]
[-component_item_id=component-item-id]
[-query=saved-query [-query_entries=entry=value [entry=value,...]]]
[-list] [-verbose] [-print_cols=all | col[col...] ] [-history] [-attrs]
[-single_line] [-no_transform] [-print_queue] [-timing]
[-validate] [-check] [-queue_check] [-h]
```

ARGUMENTS

-u

Specifies the user ID.

This is generally **infodba** or another user with administration privileges. If this argument is used without a value, the operating system user name is used.

Note If Security Services single sign-on (SSO) is enabled for your server, the **-u** and **-p** arguments are authenticated externally through SSO rather than being authenticated against the Teamcenter database. If you do not supply these arguments, the utility attempts to join an existing SSO session. If no session is found, you are prompted to enter a user ID and password.

-p

Specifies the password.

If used without a value, the system assumes a null value.

If this argument is not used, the system assumes the *user-ID* value to be the password.

This argument is mutually exclusive with the **-pf** argument.

-pf

Specifies the password file. If used without a value, the system assumes a null value. If this argument is not used, the system assumes the *user-ID* value to be the password.

For more information about managing password files, see [Manage password files](#).

This argument is mutually exclusive with the **-p** argument.

-g

Specifies the group associated with the user.

If used without a value, the user's default group is assumed.

-item_id

Tracks the given item to find appearances for the appearance roots.

-key

Tracks the given item, specified by key, to find appearances for the appearance roots. Use the following format:

```
[keyAttr1=keyVal1] [,keyAttr2=keyVal2]...[,keyAttrN=keyValN]
```

To find the key of an object, use the [get_key_string](#) utility.

For more information, see the [Business Modeler IDE Guide](#).

-config_rule

Tracks the given configuration rule to find appearances for the appearance roots.

-view_type

Tracks the given view type to find appearances for the appearance roots.

-root_only

Lists only the appearance roots, not the appearances.

-date

Locates appearances configured on the specified date. Valid values are:

date Specifies a date in the format *yyyy MM dd hh mm ss*.

now Specifies appearances configured at this moment.

today Specifies appearances configured from midnight until the present time of the current day.

-unit_no

Locates appearances configured for the specified unit number.

-component_item_id

Specifies that appearances be filtered corresponding to the specified component item.

-query

With component Item(Revision) satisfying the given query, either:

- **-ics=class[,class,...]**

OR

- **-ics=class,attr-id:value-clause[,attr-id:value-clause,...]**

With component Item(Revision) satisfying the given Classification data:

mapped_attrs=name operator value[,name operator value,...]

Where *operator* is one of the following:

=, !=, >, >=, < or <=

... with the given mapped attribute values.

-list

Specifies that the appearances are output in a simple list.

-verbose

Specifies that the appearances are output as an indented BOM.

-print_cols

Use with the **-verbose** argument to specify the columns to display. Valid values are **all** or *column-id*, where column ids are **comp**, **dates**, **unit_nos**, **occ**, **parent**, **creation_date**, **precise**, **Component Item Rev**.

-history

Use with the **-verbose** argument to include appearance history in the output.

-attrs

Use with the **-verbose** argument to print mapped attributes in the output.

-single_line

Use with the **-attrs** argument to output mapped attributes on the same line as main appearance attributes.

-no_transform

Use with the **-attrs** argument to print mapped attributes but not the transform matrix.

-print_queue

Prints information about all the primary queue entries affecting the appearance root (includes the processed entries).

-timing

Prints timing information in the output.

-validate

Compares the appearances with the equivalent product structure.

Note

You must also supply the **-date** argument, **-unit_no** argument, or both arguments, depending on the revision rule.

-check

Checks each appearance for duplication.

-queue_check

Places entries in the queue to check the appearance root.

-h

Displays help for this utility.

ENVIRONMENT

As specified in *Manually configuring your environment for Teamcenter utilities*.

FILES

As specified in *Log files produced by Teamcenter*.

RESTRICTIONS

None.

EXAMPLES

The following code segment shows the invocation of this utility.

```
L:\>find appearances -item id=APPR 0720 I1 BH -print_cols=dates,precise
-verbose -date=now Found 1 AppearanceRoot
AppearanceRoot[0] = 00000d93 = g5OFI8KcAAgcRA (Item ID: APPR 0720 I1 BH (View: view)
Revision Rule: 0150 Precise Production Appearances)), ok at 2004-10-28 13:55:12
APPR extent took 0.41 real-secs, 0.01 cpu-secs, 0.00 child-cpu-secs
n_appearances = 18
  Appearance
--> AJDFI85zAAgcRA      In Date      Out Date      Precise      Component Item Rev
--> AxAFI8qKAAGcRA      04/10/28 13:55:12 99/12/30 23:59:59 - -
--> A5HFI8qKAAGcRA      04/10/28 13:56:03 99/12/30 23:59:59 - -
--> QBNFI8qKAAGcRA      04/10/28 13:56:03 99/12/30 23:59:59 - -
--> QBPFI8qKAAGcRA      04/10/28 13:56:03 99/12/30 23:59:59 - -
--> QFBFI8qKAAGcRA      04/10/28 13:56:03 99/12/30 23:59:59 - -
--> QNJFI8qKAAGcRA      04/10/28 13:56:03 99/12/30 23:59:59 - -
--> A5FFI87ZAAgcRA      04/10/28 13:57:40 99/12/30 23:59:59 Precise APPR_0720_B1P1_BH/D-Pre
--> QJFPI87ZAAgcRA      04/10/28 13:57:40 99/12/30 23:59:59 Precise APPR_0720_B1P3_BH/A-Pre
--> QBGFI87ZAAgcRA      04/10/28 13:57:40 99/12/30 23:59:59 Precise APPR_0720_B1P2_BH/B-Pre
--> QFDFI87ZAAgcRA      04/10/28 13:57:40 99/12/30 23:59:59 Precise APPR_0720_B1P4_BH/B-Pre
--> QJLFI87ZAAgcRA      04/10/28 13:57:40 99/12/30 23:59:59 Precise APPR_0720_B1P8_BH/A-Pre
--> QJNFI87ZAAgcRA      04/10/28 13:57:40 99/12/30 23:59:59 Precise APPR_0720_B1P9_BH/A-Pre
--> QNLF87ZAAgcRA      04/10/28 13:56:03 99/12/30 23:59:59 - -
--> A5EFI8roAAgcRA      04/10/28 13:57:02 99/12/30 23:59:59 Precise APPR_0720_B2P1_BH/B-Pre
--> QBFFI8roAAgcRA      04/10/28 13:57:02 99/12/30 23:59:59 Precise APPR_0720_B2P2_BH/A-Pre
--> QBHFI8roAAgcRA      04/10/28 13:57:02 99/12/30 23:59:59 Precise APPR_0720_B2P3_BH/A-Pre
--> QNHFI8qKAAGcRA      04/10/28 13:56:03 99/12/30 23:59:59 - -
=====
There were 9 notes and 0 errors during this run
Please see log file in \teamcenter_wnti\log\find_appearancescc10df80.syslog
=====
```

find_appearances utility

find_recently_saved_item_rev

Allows you to search for item revisions with a **UGMASTER** dataset or BOM view revision that has been modified during a range of dates. Use a date before the earliest assembly was created to ensure a listing of all changed items.

SYNTAX

```
find_recently_saved_item_rev [-u=user-id {-p=password |  
-pf=password-file} -g=group]  
-start_date=DD-MMM-YYYY HH:MM:SS -end_date=DD-MMM-YYYY  
HH:MM:SS -obj_type=object-type [-out_file=output-filename |  
-outItemRevKeyFile=output-filename] -h
```

ARGUMENTS**-u**

Specifies the user ID.

This is generally **infodba** or another user with administration privileges. If this argument is used without a value, the operating system user name is used.

Note

If Security Services single sign-on (SSO) is enabled for your server, the **-u** and **-p** arguments are authenticated externally through SSO rather than being authenticated against the Teamcenter database. If you do not supply these arguments, the utility attempts to join an existing SSO session. If no session is found, you are prompted to enter a user ID and password.

-p

Specifies the password.

If used without a value, the system assumes a null value.

If this argument is not used, the system assumes the *user-ID* value to be the password.

This argument is mutually exclusive with the **-pf** argument.

-pf

Specifies the password file. If used without a value, the system assumes a null value. If this argument is not used, the system assumes the *user-ID* value to be the password.

For more information about managing password files, see [Manage password files](#).

This argument is mutually exclusive with the **-p** argument.

-g

Specifies the group associated with the user.

If used without a value, the user's default group is assumed.

-start_date

Defines the date and time from which the item revisions are searched. The time specifies a time in the current time zone for the machine where the program is running. Use the following format: *dd-mmm-yyyy hh:mm:ss*. For example, 01-Jan-2002 13:00:00.

This argument is required.

-end_date

Defines the date and time before which the item revisions are searched. The time specifies a time in the current time zone for the machine where the program is running.

Use the following format: *dd-mmm-yyyy hh:mm:ss*. For example, 01-Jan-2002 13:00:00.

If this argument is not defined, the item revisions are searched until the current date.

This argument is optional.

-obj_type

Specifies the item revision type to be searched. This argument is optional; if not defined, objects of all item revision types are searched.

-out_file

Specifies the name of the file to which the list of item revisions is sent. This argument is optional; if not defined, the output is written to the standard output.

The output includes key values.

-outItemRevKeyFile

Specifies the name of the item revision key file to which the list of item revisions is sent. This argument is optional; if not defined, the output is written to the standard output.

-h

Displays help for this utility.

ENVIRONMENT

This utility should be run from a shell where the Teamcenter environment is set.

FILES

As specified in [Log files produced by Teamcenter](#).

RESTRICTIONS

None.

EXAMPLES

To list the item revisions saved after January 01, 2002:

```
$TC_ROOT/bin/find_recently_saved_item_rev
-start_date="01-Jan-2002 00:00:00" -out_file=saved_items.txt
```

find_released_item_rev

Allows you to create a query based on date and object type. The query is performed on the Teamcenter database and generates the released item revision list to identify released item revisions. Use a date before the earliest assembly was created to ensure a listing of all released items.

SYNTAX

find_released_item_rev [-u=*user-id* {-p=*password* | -pf=*password-file*} -g=*group*]

ARGUMENTS**-u**

Specifies the user ID.

This is generally **infodba** or another user with administration privileges. If this argument is used without a value, the operating system user name is used.

Note

If Security Services single sign-on (SSO) is enabled for your server, the **-u** and **-p** arguments are authenticated externally through SSO rather than being authenticated against the Teamcenter database. If you do not supply these arguments, the utility attempts to join an existing SSO session. If no session is found, you are prompted to enter a user ID and password.

-p

Specifies the password.

If used without a value, the system assumes a null value.

If this argument is not used, the system assumes the *user-ID* value to be the password.

This argument is mutually exclusive with the **-pf** argument.

-pf

Specifies the password file. If used without a value, the system assumes a null value. If this argument is not used, the system assumes the *user-ID* value to be the password.

For more information about managing password files, see [Manage password files](#).

This argument is mutually exclusive with the **-p** argument.

-g

Specifies the group associated with the user.

If used without a value, the user's default group is assumed.

-start_date

Defines the date and time from which the item revisions are searched. The time specifies a time in the current time zone for the machine where the program is running. This argument is required.

Use the following format: *dd-mmm-yyyy hh:mm:ss*. For example, 01-Jan-2002 13:00:00.

-end_date

Defines the date and time before which the item revisions are searched. The time specifies a time in the current time zone for the machine where the program is running.

Use the following format: *dd-mmm-yyyy hh:mm:ss*. For example, 01-Jan-2002 13:00:00.

If this argument is not defined, the item revisions are searched until the current date.

This argument is optional.

-obj_type

Specifies the object type to be searched. This argument is optional; if not defined, objects of all types are searched.

-out_file

Specifies the name of the file to which the list of item revisions is sent. This argument is optional; if not defined, the output is written to the standard output.

The output includes key values.

-outItemRevKeyFile

Specifies the name of the item revision key file to which the list of item revisions is sent. This argument is optional; if not defined, the output is written to the standard output.

-h

Displays help for this utility.

ENVIRONMENT

This utility should be run from a shell where the Teamcenter environment is set.

FILES

As specified in [Log files produced by Teamcenter](#).

RESTRICTIONS

None.

EXAMPLES

To list all the item revisions released after January 01, 2002:

```
$TC_ROOT/bin/find_released_item_rev -start_date="01-Jan-2002 00:00:00"
-out_file=released_items.txt
```

query_xml

Creates, modifies, writes, deletes, and runs queries from an XML formatted file.

SYNTAX

query_xml [-u=*user-id* {-p=*password* | -pf=*password-file*} -g=*group*]
-v -f=*xml-command-file* [-o=*output-file-name*] [-h]

ARGUMENTS**-u=**

Specifies the user ID.

This is generally **infodba** or another user with administration privileges. If this argument is used without a value, the operating system user name is used.

Note

If Security Services single sign-on (SSO) is enabled for your server, the **-u** and **-p** arguments are authenticated externally through SSO rather than being authenticated against the Teamcenter database. If you do not supply these arguments, the utility attempts to join an existing SSO session. If no session is found, you are prompted to enter a user ID and password.

-p=

Specifies the password.

If used without a value, the system assumes a null value.

If this argument is not used, the system assumes the *user-ID* value to be the password.

This argument is mutually exclusive with the **-pf** argument.

-pf

Specifies the password file. If used without a value, the system assumes a null value. If this argument is not used, the system assumes the *user-ID* value to be the password.

For more information about managing password files, see [Manage password files](#).

This argument is mutually exclusive with the **-p** argument.

-g=

Specifies the group associated with the user.

If used without a value, the user's default group is assumed.

-f= *xml-command-file*

Specifies the fully qualified name of the XML file used to control processing. This argument is mandatory.

-o= *output-file-name*

Specifies the name of the file to which the output from the write and execute processes are written. This argument is optional. If unspecified, the output is sent to the console. This parameter is required for the **execute** and **execute_tuple** command types.

The **-o** argument specified for outputting a generated query result file does not support appending the results of multiple queries to the specified output file. If you

have more than one query specified in the **query_xml** utility, only the last query is included in the defined output file. The documentation for **-o** usage is correct for a single query but does not describe the scenario or required parameter for multiple queries and output files.

If you execute multiple queries using **query_xml**, add a parameter called **outputFileName** to the query definition to specify a unique output file for each query. Because queries can result in structurally different XML content per context, queries cannot be appended together and must be treated as unique files per query.

For example:

```
?xml version="1.0" encoding="UTF-8"?>
<ImanQueryCommandFile site_name="arh" site_id="id">
  <ImanQueryCommand command="execute">
    <name value="Dataset Name"/>
    <query_input_parameter name="Dataset Name" value="test1"/>
    <query_pff_post pffName="Admin - Objects By Status"
      outputFileName="d:\\outpff1.xml"/>
  </ImanQueryCommand>
  <ImanQueryCommand command="execute">
    <name value="Dataset Name"/>
    <query_input_parameter name="Dataset Name" value="test2"/>
    <query_pff_post pffName="Admin - Objects By Status"
      outputFileName="d:\\outpff2.xml"/>
  </ImanQueryCommand>
  <ImanQueryCommand command="execute">
    <name value="Dataset Name"/>
    <query_input_parameter name="Dataset Name" value="test3"/>
    <query_pff_post pffName="Admin - Objects By Status"
      outputFileName="d:\\outpff3.xml"/>
  </ImanQueryCommand>
</ImanQueryCommandFile>
```

Running **query_xml -u=infodba -p=infodba -input=myfile** generates three output files (**outpff1.xml**, **outpff2.xml**, **outpff3.xml**).

-v

Specifies verbose mode.

-h

Displays help for this utility.

ENVIRONMENT

As specified in [Manually configuring your environment for Teamcenter utilities](#). If the **TC_TMP_DIR** variable is not set, set it to a temporary location.

FILES

As specified in [Log files produced by Teamcenter](#) and the following files:

- **qry_filerunner_def.dtd**
Defines the format of the driving file.
- **pffdef.dtd**

Defines the output format when the PFF option is used.

RESTRICTIONS

None.

**RETURN
VALUES**

**Return value upon
success** 0

**Return value upon
failure** Nonzero

EXAMPLES

To create a query, enter the following command on the command line:

```
query_xml -f=xml-file-name -u=infodba -p=infodba -o=output-file
```

Note

To receive output for the **execute** and **execute_tuple** command types, you must supply a **pffName** in the input XML command file and an output file name in the **-o=*output-file-name*** parameter on the **query_xml** call. Output file names specified in the input XML command file are ignored.

XML FILE FORMAT AND EXAMPLES

The XML file must conform to the format shown in the following code segments.

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE ImanQueryCommandFile [
<!-- this is all we need to drive the iman query command line processor query_xml -->
<!-- the redundant query definition -->
  <!ELEMENT name EMPTY>
  <!ATTLIST name value CDATA #REQUIRED>
  <!ELEMENT description EMPTY>
  <!ATTLIST description value CDATA #REQUIRED>
  <!ELEMENT class EMPTY>
  <!ATTLIST class value CDATA #REQUIRED>
  <!ELEMENT clauses_real (#PCDATA)>
  <!ELEMENT clauses_display (#PCDATA)>
  <!ELEMENT uniqueid EMPTY>
  <!ATTLIST uniqueid value CDATA #REQUIRED>
  <!ELEMENT iflag EMPTY>
  <!ATTLIST iflag value CDATA #REQUIRED>
  <!ELEMENT ImanQueryDefinition (name, description, class, clauses_real,
clauses_display?, uniqueid, iflag)>
<!-- if we are executing a query - the name and value of the search parameters... -->
  <!ELEMENT query_input_parameter EMPTY>
  <!ATTLIST query_input_parameter name CDATA #REQUIRED>
  <!ATTLIST query_input_parameter value CDATA #REQUIRED>
<!-- if we want the output of the query put through a pff and written to a file... -->
  <!ELEMENT query_pff_post EMPTY>

  <!ATTLIST query_pff_post pffName CDATA #REQUIRED> <!-- the pff to use (must be in db)-->
    outputFileName CDATA #REQUIRED> <!-- file to write data (no longer used)-->
<!-- the encapsulation of the command. the attribute says it all. -->
<!-- for the create and modify the program expects the ImanQueryDefinition -->
<!-- for the execute delete and write the name is sufficient though the -->
```

XML file format (Continued)

```

<!-- full definition of the query will work.                                -->
  <!-- ELEMENT ImanQueryCommand ((name | ImanQueryDefinition), (query_input_parameter)*,
    (query_pff_post)? )>
    <!-- ATTLIST ImanQueryCommand command (create | modify | execute | execute_tuples |
delete | write_query) #REQUIRED>
<!-- the command file can contain a list of commands... -->
<!-- the site_name and site_id allow sys-admins to -->
<!-- reconcile attribute differences based on site -->
  <!-- ELEMENT ImanQueryCommandFile (ImanQueryCommand)*>
    <!-- ATTLIST ImanQueryCommandFile site_name CDATA #IMPLIED
      site_id CDATA #IMPLIED>
]> <ImanQueryCommandFile site_name="fred" site_id="id">
  <ImanQueryCommand command="create">
    <ImanQueryDefinition>
      <name value="mjsABCXML_commandfilein"/>
      <description value="no description"/>
      <class value="ItemRevision"/>
      <clauses_real>
        SELECT qid FROM ItemRevision WHERE
"Form:IMAN_specification.ECOSample:data_file.charge_number"
= "${charge = }" </clauses_real>
      <uniqueid value="0"/>
      <iflag value="0"/>
    </ImanQueryDefinition>
  </ImanQueryCommand>
<ImanQueryCommand command="modify">
  <ImanQueryDefinition>
    <name value="mjsABC"/>
    <description value="a better description"/>
    <class value="ItemRevision"/>
    <clauses_real>
      SELECT qid FROM ItemRevision WHERE
"Form:IMAN_specification.ECOSample:data_file.charge_number"
= "${charge = }" </clauses_real>
    <uniqueid value="0"/>
    <iflag value="0"/>
  </ImanQueryDefinition>
</ImanQueryCommand>
<ImanQueryCommand command="execute">
  <name value="i2ir"/>
  <query_input_parameter name="ID" value=""/>
  <query_input_parameter name="Revision" value="B"/>
  <query_pff_post pffName="PFF Name 2" outputFileName="z:\\junkpff.xml"/>
</ImanQueryCommand>
<ImanQueryCommand command="execute_tuples">
  <name value="i2ir"/>
  <query_input_parameter name="ID" value=""/>
  <query_input_parameter name="Revision" value="B"/>
</ImanQueryCommand>
<ImanQueryCommand command="write_query">
  <name value="i2ir"/>
</ImanQueryCommand>
<ImanQueryCommand command="delete">
  <name value="mjsABCXML_commandfilein"/>
</ImanQueryCommand>
</ImanQueryCommandFile>

```

XML file format

```
<?xml version="1.0" encoding="UTF-8"?>
<ImanQueryCommandFile site_name="arh" site_id="id">
  <ImanQueryCommand command="create">
    <ImanQueryDefinition>
      <name value="command2file"/>
      <description value="no description"/>
      <class value="ItemRevision"/>
      <clauses_real>
        SELECT qid FROM ItemRevision
          WHERE "object_name" = "${Name = }"
          AND   "item_revision_id" = "${Revision = }"
      </clauses_real>
      <uniqueid value="0"/>
      <iflag value="0"/>
    </ImanQueryDefinition>
  </ImanQueryCommand>
  <ImanQueryCommand command="execute">
    <name value="command2file"/>
    <query_input_parameter name="Name" value="newnewnew"/>
    <query_pff_post pffName="Admin - Objects By Status"
      outputFile="d:\\temp\\outpff4.xml"/>
  </ImanQueryCommand>
  <ImanQueryCommand command="delete">
    <name value="command2file"/>
  </ImanQueryCommand>
</ImanQueryCommandFile>
```

XML file example

```
<?xml version="1.0" encoding="UTF-8"?>
<ImanQueryCommandFile site_name="arh" site_id="id">
  <ImanQueryCommand command="execute">
    <name value="Item Revision..."/>
    <query_input_parameter name="Revision" value="A"/>
    <query_pff_post pffName="Admin - Objects By Status"
      outputFile="d:\\Users\\outpff.xml"/>
  </ImanQueryCommand>
  <ImanQueryCommand command="write_query">
    <name value="Item Revision..."/>
  </ImanQueryCommand>
</ImanQueryCommandFile>
```

XML file example

QUERY TYPES

Create, modify, and delete the following query types using the **query_xml** utility. Use the following values in the **iflag** field to specify the query type in the **.xml** file for create and modify tasks. The value for delete remains the same.

Local Query	"0"
Remote Query	"1"
User Exit Query	"8"
User Query	"16"
Keyword Search Query	"24"
Structure Query	"40"

For information about where these query types are applicable, see the [Query Builder Guide](#).

The following code is an example for a BOM structure query modification.

```
<?xml version="1.0" encoding="UTF-8"?>
<ImanQueryCommandFile site_name="arh" site_id="id">
  <ImanQueryCommand command="modify">
    <ImanQueryDefinition>
      <name value="mjsABC"/>
      <description value="a better description"/>
      <class value="ItemRevision"/>
      <clauses_real>
        SELECT qid FROM ItemRevision WHERE
          "Form:IMAN_specification.ECOSample:data_file.charge_number"
          = "${charge = }"    </clauses_real>
      <uniqueid value="0"/>
      <iflag value="40"/>
    </ImanQueryDefinition>
  </ImanQueryCommand>
</ImanQueryCommandFile>
```

BOM structure query modification

tc_set_query_where_run

Runs a saved query. You can choose to run the saved query using a user exit.

See the `TC_ROOT\sample\examples\user_query.c` file for user exit sample code that can be used with this utility.

SYNTAX

```
tc_set_query_where_run [-u=user-id {-p=password | -pf=password-file} -g=group]
-query=query-name -run={iman | query | user} [-h]
```

ARGUMENTS

-u

Specifies the user ID.

This is generally **infodba** or another user with administration privileges. If this argument is used without a value, the operating system user name is used.

Note If Security Services single sign-on (SSO) is enabled for your server, the user and password arguments are authenticated externally through SSO rather than being authenticated against the Teamcenter database. If you do not supply these arguments, the utility attempts to join an existing SSO session. If no session is found, you are prompted to enter a user ID and password.

-p

Specifies the user's password.

If used without a value, the system assumes a null value. If this argument is not used, the system assumes the *user-ID* value to be the password.

This argument is mutually exclusive with the **-pf** argument.

-pf

Specifies the password file. If used without a value, the system assumes a null value. If this argument is not used, the system assumes the *user-ID* value to be the password.

For more information about managing password files, see [Manage password files](#).

This argument is mutually exclusive with the **-p** argument.

-g

Specifies the group associated with the user.

If used without a value, the user's default group is assumed.

-query

Specifies the name of saved query to run.

-run

Specifies how to run the query:

- **iman**
Run the query normally.
- **query**

Run using a user exit.

- **user**

Run using a user exit and display results in a text table.

-h

Displays help for this utility.

ENVIRONMENT

As specified in [Manually configuring your environment for Teamcenter utilities](#).

FILES

As specified in [Log files produced by Teamcenter](#).

RESTRICTIONS

None.

Chapter

13 Maintenance utilities

Installation

install

Performs limited Teamcenter maintenance and Oracle database administration.

SYNTAX

```
install
  {-p=password | -pf=password-file}
  [-add_func_index user-id password grp class index-name unique-flag
  fn class-name attr1 attr2...]
  [-add_index user-id password group class index-name unique-flag
  class-name attr1 attr2...]
  [-ask_version]
  [-ask_xmit_file user-id password group object-file]
  [-drop_index user-id password group class-name index-name]
  [-encrypt]
  [-encryptpwf -e=environment-variable-name -f=pw-file-name]
  [-find_control_chars user-id password group class-name attribute]
  [-gen_xmit_file user-id password group]
  [-h]
  [-lock_db user-id password group]
  [-pom_object_index user-id password group]
  [-regen_schema_file user-id password group]
  [-replace_control_chars user-id password group class hex-bad-char replacement-string]
  [-reset_last_login_time user-id password group]
  [-temp_table user-id password group list ]
  [-unlock_db user-id password group]
```

ARGUMENTS**-p**

Specifies the password associated with the *user-ID* value.

This argument is mutually exclusive with the **-pf** argument. One of the two mutually exclusive password elements is required.

-pf

Specifies the password file.

For more information about managing password files, see [Manage password files](#).

This argument is mutually exclusive with the **-p** argument. One of the two mutually exclusive password elements is required.

-add_func_index *user-id password grp class index-name unique-flag fn class-name attr1 attr2...*

Generates an index on the **upper** field name of the given attribute to allow case-insensitive search. Other functions should only be used upon guidance from Siemens PLM Software. The following values must be supplied in order:

user-id

Specifies a system administration user ID. In most case, this is **infodba** or another user ID with similar privileges.

password

Specifies the password or password file associated with the *user-ID* value.

For more information, see the descriptions of the **-p** and **-pf** arguments.

grp

Specifies the group associated with the *user-ID* value. In most cases, the group is **dba**.

index-name

Specifies the name of the index. This is internal to POM and not the name used in Oracle.

unique-flag

1 indicates that the index is unique. **0** indicates that the index allows duplicates.

fn

If this argument is omitted, the **upper** field name is assumed.

class-name

Specifies the name of the class.

list-of-attributes

Specifies the list of attributes of the class separated by a blank space.

Note

- The order of attributes is important. Take care when creating indexes on a group of attributes.
- The user, password, and group arguments can alternatively be supplied in one of the following forms:
 - o **-u=user-id -p=password -pf=password file**
 - o **-g=group**

-add_index *user-id password group class index-name unique-flag class-name attr1 attr2...*

Generates a new POM schema file. Requires Teamcenter system administration privileges. The following values must be supplied in order:

user-id

Specifies a system administration user ID. In most case, this is **infodba** or another user ID with similar privileges.

password

Specifies the password or password file associated with the *user-ID* value.

For more information, see the descriptions of the **-p** and **-pf** arguments.

group

Specifies the group associated with the *user-ID* value. In most cases, the group is **dba**.

index-name

Specifies the name of the index. This is internal to POM and not the name used in Oracle.

unique-flag

1 indicates that the index is unique. **0** indicates that the index allows duplicates.

class-name

Specifies the name of the class.

list-of-attributes

Specifies the list of attributes of the class separated by a blank space.

Note The order of attributes is important. Take care when creating indexes on a group of attributes.

-ask_version

Returns the current version of Teamcenter stored in the Oracle database.

-add_xmit_file *user-id password group object-file*

Returns the **transmit file** required by a Classic Multi-Site **objects.meta** file. This requires **POM_TRANSMIT_DIR** to be set. The following arguments must be supplied in order:

user-id

Specifies a system administration user ID. In most case, this is **infodba** or another user ID with similar privileges.

password

Specifies the password or password file associated with the *user-ID* value.

For more information, see the descriptions of the **-p** and **-pf** arguments.

group

Specifies the group associated with the *user-ID* value. In most cases, the group is **dba**.

object-file

Specifies the object file to transmit. Alternatively, use this syntax: **-f=object-file**.

- Note**
- The order of attributes is important. Take care when creating indexes on a group of attributes.
 - The user, password, and group arguments can alternatively be supplied in one of the following forms:
 - o **-u=user-id -p=password -pf=password file**
 - o **-g=group**

-drop_index *user-id password group class-name index-name*

Drops an index to revert a change made using **install -add_index**. The following arguments must be supplied in order:

user-id

Specifies a system administration user ID. In most case, this is **infodba** or another user ID with similar privileges.

password

Specifies the password or password file associated with the *user-ID* value.

For more information, see the descriptions of the **-p** and **-pf** arguments.

group

Specifies the group associated with the *user-ID* value. In most cases, the group is **dba**.

class-name

Specifies the name of the class.

index-name

Specifies the name of the index to drop.

Note

- The order of attributes is important. Take care when creating indexes on a group of attributes.
- The user, password, and group arguments can alternatively be supplied in one of the following forms:
 - o **-u=user-id -p=password -pf=password file**
 - o **-g=group**

-encrypt

Reads a database connect string from the **TC_DB_CONNECT** environment variable and displays on the console that connect string with the password encrypted.

To change the database password, you need to change the password (see the [System Administration Guide](#)), set the **TC_DB_CONNECT** environment variable to contain that new password, run this utility specifying the **-encrypt** option, and copy the new version of the connect string it outputs into the **tc_profilevars** file.

-encryptpwf

Creates an encrypted password file. Requires the **-e** argument to identify the environment variable that contains the password to encrypt and the **-f** argument that indicates the name and location of the file containing the encrypted password.

-e= environment-variable-name

Specifies the environment variable that contains the password to encrypt. This argument is valid only if you specify the **-encryptpwf** argument. You can specify any environment variable that contains a clear text password value that you want to encrypt. For security reasons, chose an obscure and unique name for the environment variable and immediately unset the environment variable after you run the utility.

-f= password-file-name

Specifies the name and location for the file containing the encrypted password used to connect to the Teamcenter database. This file is used by Teamcenter utilities when using the **-pf** argument and by some other Teamcenter functions. This argument is valid only if you specify the **-encryptpwf** argument. If the **-f=** argument is omitted, the utility uses the fourth command line argument as the file name. For example, the following commands are considered equivalent:

-

```
install -encryptpwf -e=ENV_VAL -f=pwd.txt
```

-

```
install -encryptpwf -e=ENV_VAL pwd.txt
```

-find_control_chars *user-id password group class-name attribute*

Finds control characters in the Teamcenter database. Control characters can enter the Teamcenter database through ITK operations that are legitimate codes for the server or database code pages, but cause problems for XML because they are illegal characters and thus cause problems for SOA based integrations in particular. To find such characters, the following arguments must be supplied in order:

user-id

Specifies a system administration user ID. In most case, this is **infodba** or another user ID with similar privileges.

password

Specifies the password or password file associated with the *user-ID* value.

For more information, see the descriptions of the **-p** and **-pf** arguments.

group

Specifies the group associated with the *user-ID* value. In most cases, the group is **dba**.

class-name

Specifies the name of the class.

attribute

Specifies the attribute.

Note

- The order of attributes is important. Take care when creating indexes on a group of attributes.
- The user, password, and group arguments can alternatively be supplied in one of the following forms:
 - o **-u=user-id -p=password -pf=password file**
 - o **-g=group**

-gen_xmit_file *user-id password group*

Generates the transmit file containing a copy of the Teamcenter schema that is used by POM during import to compare the exporting site schema definition to the importing site schema definition for all classes. The file resides in the **\$POM_TRANSMIT_DIR** directory.

user-id

Specifies a system administration user ID. In most case, this is **infodba** or another user ID with similar privileges.

password

Specifies the password or password file associated with the *user-ID* value.

For more information, see the descriptions of the **-p** and **-pf** arguments.

group

Specifies the group associated with the *user-ID* value. In most cases, the group is **dba**.

Note The user, password, and group arguments can alternatively be supplied in one of the following forms:

- **-u**=*user-id* **-p**=*password* **-pf**=*password file*
- **-g**=*group*

-h

Displays help for this utility.

-lock_db

Locks the sites against further use. The lock remains in place until unlocked with the **-unlock_db** argument. The following values must be supplied in this order:

user-id

Specifies a system administration user ID. In most cases, this is **infodba** or another user ID with similar privileges.

password

Specifies the password or password file associated with the *user-ID* value.

For more information, see the descriptions of the **-p** and **-pf** arguments.

group

Specifies the group associated with the *user-ID* value. In most cases, the group is **dba**.

See restrictions #1 and #4.

-pom_object_index *user-id password group*

Specifies an optional index on the **POM_object** class that allows queries to answer high frequency queries directly from the indexes (instead of referring to the table row). This is not a default argument because it requires additional database server memory.

This affects the following indexes:

- **PPOM_OBJECT_SLN** on **puid** and **ptimestamp**
- **PPOM_OBJECT_UPI** on **puid** and **ppid**

user-id

Specifies a system administration user ID. In most case, this is **infodba** or another user ID with similar privileges.

password

Specifies the password or password file associated with the *user-ID* value.

For more information, see the descriptions of the **-p** and **-pf** arguments.

group

Specifies the group associated with the *user-ID* value. In most cases, the group is **dba**.

Note The user, password, and group arguments can alternatively be supplied in one of the following forms:

- **-u**=*user-id* **-p**=*password* **-pf**=*password file*
- **-g**=*group*

-regen_schema_file

Generates a new POM schema file. Requires Teamcenter system administration privileges. The following values must be supplied in order:

user-id

Specifies a system administration user ID. In most cases, this is **infodba** or another user ID with similar privileges.

password

Specifies the password or password file associated with the *user-ID* value.

For more information, see the descriptions of the **-p** and **-pf** arguments.

group

Specifies the group associated with the *user-ID* value. In most cases, the group is **dba**. See restrictions #1 and #3.

-replace_control_chars *user-id password group class hex-bad-char replacement-string*

Replaces control characters found by the **-find_control_chars** argument.

An temporary alternative to fixing the data in the database is to set an environment variable (in the *TC_DATA\tcprofilevars.bat* file) to replace control characters. For example, to replace control characters with spaces (ASCII character 32), set the following variable:

POM_STRIP_CTRL_CHARS=32

To use the **-replace_control_chars** argument, the following values must be supplied in this order:

user-id

Specifies a system administration user ID. In most case, this is **infodba** or another user ID with similar privileges.

password

Specifies the password or password file associated with the *user-ID* value.

For more information, see the descriptions of the **-p** and **-pf** arguments.

group

Specifies the group associated with the *user-ID* value. In most cases, the group is **dba**.

class

Specifies the class.

hex-bad-char

Specifies the hexadecimal code of the control character to replace.

replacement-string

Specifies the string to insert in place of control characters.

Note The user, password, and group arguments can alternatively be supplied in one of the following forms:

- **-u=user-id -p=password -pf=password file**
- **-g=group**

-reset_last_login_time *user-id password group*

Updates the last login time for the user authenticating through a command line. This argument is suitable for batch processing. The following values must be supplied in this order:

user-id

Specifies a system administration user ID. In most case, this is **infodba** or another user ID with similar privileges.

password

Specifies the password or password file associated with the *user-ID* value.

For more information, see the descriptions of the **-p** and **-pf** arguments.

group

Specifies the group associated with the *user-ID* value. In most cases, the group is **dba**.

Note The user, password, and group arguments can alternatively be supplied in one of the following forms:

- **-u=user-id -p=password -pf=password file**
- **-g=group**

-temp_table *user-id password group list*

Specifies periodic checking of whether Oracle temporary table definitions are accumulating.

Oracle temporary table definitions can accumulate when the server exits abruptly, for example, when a database connection is lost.

You can list these table definitions using the **list** option or purge them using the **drop** option with a specific name or date, or purge all.

The following values must be supplied in this order:

user-id

Specifies a system administration user ID. In most case, this is **infodba** or another user ID with similar privileges.

password

Specifies the password or password file associated with the *user-ID* value.

For more information, see the descriptions of the **-p** and **-pf** arguments.

group

Specifies the group associated with the *user-ID* value. In most cases, the group is **dba**.

list

Specifies to list accumulated temporary table definitions.

Alternatively, use the **drop** option instead of **list** to drop temporary table definitions:

- **drop all**
Drops all definitions
- **drop older_than_date=YYYY/MM/DD HH:MI:SS**
Drops definitions older than the specified date and time.
- **drop table_name=temp-table-name**
Drops the specified temporary table.

Note

The user, password, and group arguments can alternatively be supplied in one of the following forms:

- **-u=user-id -p=password -pf=password file**
- **-g=group**

-unlock_db *user-id password group*

Releases locks set with the **-lock_db** argument. The following values must be supplied in this order:

user-id

Specifies a system administration user ID. In most cases, this is **infodba** or another user ID with similar privileges.

password

Specifies the password or password file associated with the *user-ID* value.

For more information, see the descriptions of the **-p** and **-pf** arguments.

group

Specifies the group associated with the *user-ID* value. In most cases, the group is **dba**.

See restriction #1.

ENVIRONMENT

As specified in *Manually configuring your environment for Teamcenter utilities* and the following environment variables:

TC_DB_CONNECT
POM_SCHEMA
POM_TRANSMIT_DIR

For more information, see the *Preferences and Environment Variables Reference*.

FILES

As specified in *Log files produced by Teamcenter* and the following:

- **\$TC_DATA/tc_profilevars**
Stores site environment variable settings. This file is modified by the **-encrypt** argument.
- POM schema file data file created by the **install** utility with the **-regen_schema_file** argument.
Full file specification (directory path and file name) is set by the **POM_SCHEMA** environment variable.
- POM transmit schema file data file created by the **install** utility with the **-gen_xmit_file** argument.
Full file specification (directory path and file name) is set by the **POM_TRANSMIT_DIR** environment variable.

RESTRICTIONS

1. Common command line argument syntax for *user-id*, *password*, and *group* arguments is not supported. Values for these arguments must be separated by an equal sign (=). For example, the following syntax works:

```
$TC_BIN/install -regen_schema_file infodba password dba
$TC_BIN/install -regen_schema_file -u=infodba -p=password -g=dba
```
2. Requires Teamcenter system administration privileges and exclusive access to the system for this operation.
3. Common **-regen_schema_file** failures:
 - POM_db_connect_fail**
Unable to connect to database.
 - POM_logins_are_disabled**
Login to database is disabled.
 - POM_invalid_site_id**
Database is not populated.
 - POM_not_installed**
Database missing data.
 - POM_find_schema_failed**
Unable to create new POM schema file. Directory does not exist, cannot be written to, or the **POM_SCHEMA** environment variable is not set.
 - POM_schema_exists**
File pointed to by the **POM_SCHEMA** environment variable already exists. Delete or move this file and retry.
4. The **-lock_db** does not force logout of existing users, but does prevent additional users from logging on.
5. Only the following tokens can be changed on an existing (saved class):

POM_attr_export_as_string
POM_attr_follow_on_export
POM_attr_is_candidate_key
POM_null_is_valid
POM_public
POM_public_read
POM_public_write

For additional information, see the *Server Customization Programmer's Guide*.

6. When adding a new custom privilege, you must have previously added that privilege to the **am_text.uil** file and recompiled the file.

For additional information, see the *Server Customization Programmer's Guide*.

7. Rules-based object protection must be enabled in order to add and use new custom privileges.

EXAMPLES

To regenerate the POM transmit schema file, enter the following command on a single line:

```
$TCROOT/bin/install -gen_xmit_file infodba password dba
```

tem.sh/.bat

Starts the Teamcenter Environment Manager utility. Use the **-s** to bypass the Teamcenter Environment Manager user interface and run the installation in the background. There is no feedback when the silent install is running.

SYNTAX

application_root/install/tem -s=file-name

ARGUMENTS**-s**

Performs a silent install. Teamcenter Environment Manager (TEM) looks in the current working directory for the configuration file to use. If a file name is specified for this argument, TEM uses the specified file as input for the silent install.

ENVIRONMENT

As specified in *[Manually configuring your environment for Teamcenter utilities](#)*.

FILES

As specified in *[Log files produced by Teamcenter](#)*.

RESTRICTIONS

None.

Audit Manager

migrate_audit_auditdefinitions

Exports audit definition objects from the legacy Audit Manager to an XML file which is a Business Modeler IDE template.

In addition to the XML file containing audit definition data, this utility also generates a file containing localization data. This file has a **_lang** suffix and is generated in a folder called **lang**.

You can later migrate these files to the new Audit Manager application by importing and deploying it using the Business Modeler IDE.

Note If the names of the logged properties in the legacy Audit Manager are the same as the properties in **Fnd0AuditLog** object and its sub classes, this utility adds a **_local** suffix to the legacy Audit Manager property. This is done so that the legacy Audit Manager properties do not overwrite the meta definitions of various **Fnd0AuditLog** classes.

SYNTAX

```
migrate_audit_auditdefinitions [-u=user-id {-p=password |  
-pf=password-file} -g=group]-outfile  
[-h]
```

ARGUMENTS**-u**

Specifies the user ID.

This is generally **infodba** or another user with administrator privileges. If this argument is used without a value, the operating system user name is used.

Note If Security Services single sign-on (SSO) is enabled for your server, the **-u** and **-p** arguments are authenticated externally through SSO rather than being authenticated against the Teamcenter database. If you do not supply these arguments, the utility attempts to join an existing SSO session. If no session is found, you are prompted to enter a user ID and password.

-p

Specifies the password.

If used without a value, the system assumes a null value.

If this argument is not used, the system assumes the *user-id* value to be the password.

This argument cannot be replaced with the **-pf** argument.

-pf

Specifies the password file. If used without a value, the system assumes a null value. If this argument is not used, the system assumes the *user-id* value to be the password.

For more information about managing password files, see the [Utilities Reference](#).

This argument is mutually exclusive with the **-p** argument.

-g

Specifies the group associated with the user.

If used without a value, the user's default group is assumed.

-outfile

Specifies the path and filename where the XML file containing the audit definition data language file is generated. The language file is also generated in the same path.

-h

Displays help for this utility.

RESTRICTIONS

Requires Teamcenter administrator privileges.

EXAMPLES

- To export audit definition objects from the old Audit Manager application:

```
migrate_audit_auditdefinitions -u=userid -p=password -g=group -outfile=  
D:\audit_export\audit_configs.xml
```

migrate_audit_data

Exports data from the legacy Audit Manager application in a TC XML file that you can then migrate to the new Audit Manager data model. The contents of the TC XML file are based on the arguments you specify when you run this utility.

To migrate the audit data to the new Audit Manager data model, you must import the TC XML file using the bulk loader option and use TIE to create instances in the database.

You must successfully execute the [migrate_audit_auditdefinitions](#) utility before you run this utility.

SYNTAX

```
migrate_audit_data [-u=user-id {-p=password | -pf=password-file} -g=group]  
-path=location-to-export-the-TC-XML-file  
[-create_before=date-before-which-audit-logs-were-created]  
[-create_after=date-after-which-audit-logs-were-created]  
[-proc_history=]  
[-object_type=name-of-the-object]  
[-log_type=name-of-the-audit-log-object-in-the-new-Audit-Manager-applicaiton]  
[-h]
```

ARGUMENTS**-u**

Specifies the user ID.

This is generally **infodba** or another user with administrator privileges. If this argument is used without a value, the operating system user name is used.

Note If Security Services single sign-on (SSO) is enabled for your server, the **-u** and **-p** arguments are authenticated externally through SSO rather than being authenticated against the Teamcenter database. If you do not supply these arguments, the utility attempts to join an existing SSO session. If no session is found, you are prompted to enter a user ID and password.

-p

Specifies the password.

If used without a value, the system assumes a null value.

If this argument is not used, the system assumes the *user-id* value to be the password.

This argument cannot be replaced with the **-pf** argument.

-pf

Specifies the password file. If used without a value, the system assumes a null value. If this argument is not used, the system assumes the *user-id* value to be the password.

For more information about managing password files, see the [Utilities Reference](#).

This argument is mutually exclusive with the **-p** argument.

-g

Specifies the group associated with the user.

If used without a value, the user's default group is assumed.

-path

Specifies the location to export the TC XML file.

If you do not specify the **-path** argument, this utility exports the TC XML to the location from where the utility is generated.

-create_before

Specifies the date before which the audit logs were created.

The date format is `%d-%b-%Y %H:%M`. For example, **20-May-2010 12:00**.

-create_after

Specifies the date after which the audit logs were created.

The date format is `%d-%b-%Y %H:%M`. For example: **20-May-2010 12:00**.

proc_history

Specifies the process history audit log

-object_type

Specifies the object type of the audit log.

-log_type

Specifies the audit log object of the new Audit Manager application. The old audit logs will be migrated to this audit log object.

The valid log types are:

- **Fnd0GeneralAudit**
- **Fnd0FileAccessAudit**
- **Fnd0LicenseChangeAudit**
- **Fnd0LicenseExportAudit**
- **Fnd0OrganizationAudit**
- **Fnd0ScheduleAudit**
- **Fnd0StructureAudit**
- **Fnd0WorkflowAudit**

-h

Displays help for this utility.

RESTRICTIONS

Requires Teamcenter administrator privileges.

EXAMPLES

- To export all the audit data to the **D:/audit_logs** location :

```
migrate_audit_data -u=userid -p=password -g=group -path=D:/audit_logs
```

- To export audit data by date range:

```
migrate_audit_data -u=userid -p=password -g=group -path=D:/audit_logs
-create_after='20-May-2010 12:00' -create_before=' 20-May-2012 12:00'
```

- To export audit data of specific object type:

```
migrate_audit_data -u=userid -p=password -g=group -path=D:/audit_logs
```

```
-object_type='Item'
```

- To export process audit history:

```
migrate_audit_data -u=userid -p=password -g=group -path=D:/audit_logs  
-proc_history
```

- To export audit data of a specific log type:

```
migrate_audit_data -u=userid -p=password -g=group -path=D:/audit_logs  
-log_type='Fnd0GeneralAudit'
```

audit_purge

Archives audit logs in TC XML format based on retention period and audit log type. This utility also purges audit log records.

SYNTAX

```
audit_purge [-u=user-id {-p=password | -pf=password-file} -g=group]
-logtype=audit-log-business-object-name
{-purge | -archive}
[-sublogtype=process_audit | signoff_audit]
[-retentionperiod=retention-period-of-audit-log-business-object]
[-archivelocation=path-of-archive-location]
[-force_retraverse]
[-h]
```

ARGUMENTS

-u

Specifies the user ID.

This is generally **infodba** or another user with administrator privileges. If this argument is used without a value, the operating system user name is used.

Note

If Security Services single sign-on (SSO) is enabled for your server, the **-u** and **-p** arguments are authenticated externally through SSO rather than being authenticated against the Teamcenter database. If you do not supply these arguments, the utility attempts to join an existing SSO session. If no session is found, you are prompted to enter a user ID and password.

-p

Specifies the password.

If used without a value, the system assumes a null value.

If this argument is not used, the system assumes the *user-id* value to be the password.

This argument cannot be replaced with the **-pf** argument.

-pf

Specifies the password file. If used without a value, the system assumes a null value. If this argument is not used, the system assumes the *user-id* value to be the password.

For more information about managing password files, see the [Utilities Reference](#).

This argument is mutually exclusive with the **-p** argument.

-g

Specifies the group associated with the user.

If used without a value, the user's default group is assumed.

-logtype

Specifies the name of the audit log business object. For example, **Fnd0WorkflowAudit**, **Fnd0ScheduleAudit**.

Note If the audit log is **Fnd0WorkflowAudit**, you must also use the **-sublogtype** argument.

-purge

Purges the specified audit logs

If you do not specify the retention period with this argument, Teamcenter selects the retention period from the **Fnd0RetentionPeriod** business constant.

-archive

Archives the audit log in TC XML format.

If you do not specify the archive location, Teamcenter uses the archive location specified in the **Fnd0ArchiveLocation** constant.

If you do not specify the retention period, Teamcenter uses the retention period specified in the **Fnd0RetentionPeriod** constant.

-sublogtype

Specifies the sublog types of the **Fnd0WorkflowAudit** object. The sublog types of the **Fnd0WorkflowAudit** object are **process_audit** and **signoff_audit**.

-retentionperiod

Specifies the retention period of the audit log.

The **-retentionperiod** argument overrides the retention period specified in the **Fnd0RetentionPeriod** constant.

-archivelocation

Specifies the path of the archive location.

The **-archivelocation** argument overrides the retention period specified in the **Fnd0ArchiveLocation** constant.

-force_retraverse

Re-archives audit logs. Use this argument with the **-archive** argument to archive the same objects again.

-h

Displays help for this utility.

RESTRICTIONS

Requires Teamcenter administrator privileges.

EXAMPLES

Note When archiving or purging the **Fnd0WorkflowAudit** audit log, you must use the **-sublogtype** argument with the **-logtype** argument.

The sublog types of the **Fnd0WorkflowAudit** audit log are **process_audit** and **signoff_audit**.

- To purge the process audit logs of the **Fnd0WorkflowAudit** audit log:

```
audit_purge -u=userid -p=password -g=group -logtype=Fnd0WorkflowAudit  
-sublogtype=process_audit -purge
```

- To purge the process audit logs of the **Fnd0WorkflowAudit** audit log with a retention period of 90 days:

```
audit_purge -u=userid -p=password -g=group -logtype=Fnd0WorkflowAudit  
-sublogtype=process_audit -purge -retentionperiod=90
```

- To archive the process audit logs of the **Fnd0WorkflowAudit** audit log:

```
audit_purge -u=userid -p=password -g=group -logtype=Fnd0WorkflowAudit  
-sublogtype=process_audit -archive
```

- To archive the process audit logs of the **Fnd0WorkflowAudit** audit log with a retention period 90 days:

```
audit_purge -u=userid -p=password -g=group -logtype=Fnd0WorkflowAudit  
-sublogtype=process_audit -archive -retentionperiod=90
```

- To archive the process audit logs of the **Fnd0WorkflowAudit** audit log with the archive location specified as **c:\archive**:

```
audit_purge -u=userid -p=password -g=group -logtype=Fnd0WorkflowAudit  
-sublogtype=process_audit -archive -retentionperiod=90  
-archivelocation=c:\archive
```

audit_archive

Note This utility is deprecated and will be removed in a future version.

Searches the database for audit log records based on input criteria. Once it finds the records that must be archived, it processes the archive. If the **-delete_record** argument is given, the utility deletes the audit log records. Audit log entries with an audit definition with a **days kept** value of **-1** are not archived, because **-1** indicates that the log record is permanent.

SYNTAX

```
audit_archive [-u=user-id {-p=password | -pf=password-file} -g=group]
[-delete_record] [-type=type-name] [-class=class-name] [-event=event-type]
[-id=object-id]
[-revid=object-rev-id] [-name=object-name] [-owner=object-owner-id]
[-created_before=archive date] [-created_after=archive date]
[-group_name=owner-group-name]
[-obj_seqno=object sequence number]
[-secobj_type=secondary object type name]
[-secobj_id=secondary-object-id]
[-secobj_name=secondary object type name]
[-secobj_revid=secondary-object-rev-id]
[-secobj_seqno=secondary-object-seq-no]
[-error_code=error-code-of-failed-action]
[-project_id=project-id] [-project_name=project-name]
[-overwrite] [-h]
```

ARGUMENTS**-u**

Specifies the user ID.

This is generally **infodba** or another user with administration privileges. If this argument is used without a value, the operating system user name is used.

Note If Security Services single sign-on (SSO) is enabled for your server, the **-u** and **-p** arguments are authenticated externally through SSO rather than being authenticated against the Teamcenter database. If you do not supply these arguments, the utility attempts to join an existing SSO session. If no session is found, you are prompted to enter a user ID and password.

-p

Specifies the password.

If used without a value, the system assumes a null value.

If this argument is not used, the system assumes the *user-ID* value to be the password.

This argument cannot be replaced with the **-pf** argument.

-pf

Specifies the password file. If used without a value, the system assumes a null value. If this argument is not used, the system assumes the *user-ID* value to be the password.

For more information about managing password files, see [Manage password files](#).

This argument is mutually exclusive with the **-p** argument.

-g

Specifies the group associated with the user.

If used without a value, the user's default group is assumed.

-delete_record

Specifies that audit log records are deleted after being archived.

-type

Specifies that the audit logs of the specified object type needs to be archived.

-class

Specifies that the audit logs of the specified class of the object type needs to be archived.

-event

Specifies that the audit logs of the specified event type needs to be archived.

-id

Specifies that the audit logs of the specified ID of the object needs to be archived.

-revid

Specifies that the audit logs of the specified revision ID of the object needs to be archived.

-name

Specifies that the audit logs of the specified name of the object needs to be archived.

-owner

Specifies that the audit logs of the specified user who created the object needs to be archived.

-created_before

Specifies that the audit logs of objects created before the specified date need to be archived.

-created_after

Specifies that audit logs of objects created after the specified date need to be archived.

-group_name

Specifies that the audit logs of the specified group name associated with the user who created the object needs to be archived.

-obj_seqno

Specifies that the audit logs of the specified sequence number of the object needs to be archived.

-secobj_type

Specifies that the audit logs of the specified type of the secondary object associated with the object needs to be archived.

-secobj_ID

Specifies that the audit logs of the specified ID of the secondary object associated with the object needs to be archived.

-secobj_name

Specifies that the audit logs of the specified name of the secondary object associated with the object needs to be archived.

-secobj_revid

Specifies that the audit logs of the specified revision ID of the secondary object associated with the object needs to be archived.

-secobj_seqno

Specifies that the audit logs of the specified sequence number of the secondary object associated with the object needs to be archived.

-error_code

Specifies that the audit logs of the specified error code associated with the failed actions on the object needs to be archived.

-proj_id

Specifies that the audit logs of the specified ID of the project needs to be archived.

-proj_name

Specifies that the audit logs of the specified project name needs to be archived.

-overwrite

Archives permanent audit records (those with days kept value of **-1**). When used in conjunction with the **-delete_record** argument, the permanent audit records are removed from the database.

-h

Displays help for this utility.

ENVIRONMENT

As specified in the [Manually configuring your environment for Teamcenter utilities](#).

FILES

As specified in the [Log files produced by Teamcenter](#).

RESTRICTIONS

Requires Teamcenter administrator privileges.

EXAMPLES

- To archive audit log entries associated with the **ITAR_license_01** ITAR license for actions performed by the **user1** user belonging to **ADASiteAdministrator** group and created before 30th June 2009:

```
audit_archive -u=admin -p=admin -g=dba -class=ADA_License -type=ITAR_License
-id=ITAR_license_01 -owner=user1 -group=ADASiteAdministrator
-created_before=30-jun-2009 12:00
```

- To archive audit log entries related to all **__Attach_License** events associated with ITAR licenses attached to the **Part12** item, revision **A**, sequence **2**, created after 31st January 2009:

```
audit_archive -u=admin -p=admin -g=dba -class=ADA_License -type=ITAR_License
-event=__Attach_License -sec_id=Part12 -sec_revid=A -sec_seqno=2
-created_after=31-jan-2009 12:00
```

- To archive audit log entries associated with **project1** project name for actions performed by **user1** user belonging to **ProjectAdministrator** group and created before 30th September 2010:

```
audit_archive -u=admin -p=admin -g=dba -class=Requirement -type=Requirement  
-project_name=project1 -owner=user1 -group= ProjectAdministrator  
-created_before=30-sep-2010 12:00
```

combine_audit_files

Note This utility is deprecated and will be removed in a future version.

Combines all the log files into an **TcAuditLog.txt** or **TcAuditLog.xml** file.

SYNTAX

combine_audit_files.pl

ARGUMENTS**source_dir**

Specifies the source directory containing audit log files generated during Teamcenter sessions.

target_dir

Specifies the target directory containing the combined audit log file. The target directory must not be the same as the source directory, because the program tries to move the audit files from the source directory to the target directory, combine them, and delete them. The **source_dir** and **target_dir** values can be either an absolute path or a relative path.

ENVIRONMENT

This utility works on any UNIX or Windows platforms that install Perl, and the program is in their path.

FILES

The audit log files to be combined (**tc_auditlog_****.txt**, **tc_auditlog_****.xml**) are at **source_dir** directory. The combined master log files (**TcAuditLog.txt**, **TcAuditLog.xml**) are at **target_dir** directory. If the master files **TcAuditLog.txt**, **TcAuditLog.xml** are not found when running this utility, first create them, then append the original audit files to them.

RESTRICTIONS

None.

EXAMPLES

- Suppose the utility **combine_audit_files.pl** and all original audit files are in the current directory, and the master audit files are in the **C:\temp\audit** directory. The program searches all **tc_auditlog_****.xml** and **tc_auditlog_****.txt** files, moves them to the **C:\temp\audit** directory, combines them, and appends them to the **TcAuditLog.xml** and **TcAuditLog.txt** files in the **C:\temp\audit** directory. Finally, all the original audit files that are moved and appended are deleted.

```
perl combine_audit_files.pl . C:\temp\audit
```

- Here, source and target are subdirectories of the current directory. Suppose the utility program is in the current directory, original audit files are in the source directory, and the master audit files are in the target directory. It will look for all **tc_auditlog_****.txt** and **tc_auditlog_****.xml** files in the source directory, move them to the target directory, combine them, and append them to the **TcAuditLog.txt** and **TcAuditLog.xml** files in the target directory. Finally, all original audit files that are moved and appended are deleted.

In general, all audit files are generated by the program and you should not manually edit them. However, in the following situations, you must manually modify the master files using any text editor, such as **vi** or Notepad:

```
perl combine_audit_files.pl source target
```

- o If the **TC_audit_delimiter** preference is changed to a value other than the default value (^), you must manually edit the first line of the **TcAuditLog.txt** master audit file to reflect the new delimiter.

For example, if the new delimiter is a dollar sign (\$), the first line of the **TcAuditLog.txt** file must be changed to:

```
ObjectUID$ObjectId$ObjectName$revision$ObjectName$eventTypeName
$userId$loggedDate$properties
```

- o If the **TC_XML_ENCODING** environment variable in the **tc_profilevars.bat** file is changed to a value other than the default value **iso-8859-1**, you must edit the first line of the **TcAuditLog.xml** master audit file and the **TcAuditLog.xsl** and **TcAuditLogSchema.xsd** files in the sample/audit directory.

For example, if the new encoding is **Shift_JIS** (Japanese), the first line of the **TcAuditLog.xml** file is changed to:

```
<?xml version="1.0" encoding=" Shift_JIS"?>
```

Siemens PLM Software recommends that you run this utility and archive the master audit file periodically, daily, weekly, or monthly, depending upon the data growth. If the master file grows too big it would be difficult to open.

A sample of the XML program files is provided in the **sample/audit/** directory to view the XML audit data in the web browser. The four files (**TcAuditLog.xml**, **TcAuditLog.xsl**, **TcAuditLogSchema.xml**, and **TcAuditLog.js**) must be in the same directory. Opening **TcAuditLog.xml** in Microsoft Internet Explorer 5.5 or higher presents audit data in a table similar to the following that can be sorted and filtered by column.

The screenshot shows a web browser window titled 'View Audit XML Log - Microsoft Internet Explorer'. The address bar shows 'C:\temp\test\TcAuditLog.xml'. The page has a search and filter interface at the top with options for sorting (by column, descending/ascending) and filtering (by column, query string, or characters). Below this is the 'Audit Log Table' with the following data:

objectID	objectId	objectName	revision	objectTypeName	eventTypeName	userId	loggedDate	properties
wM8wEHWavvqND		edf	B	ItemRevision	__Modify	cham	2002-01-15 12:00:48	creation_date=15-Jun-2002 11:50
wM8wEHWavvqND		edf	B	ItemRevision	__Modify	shen	2002-01-15 12:00:47	creation_date=15-Jun-2002 11:50
wM8wEHWavvqND		edf	B	ItemRevision	__Modify	choim	2002-01-15 12:00:46	creation_date=15-Jun-2002 11:50
wM8wEHWavvqND		edf	B	ItemRevision	__Check_In	shen	2002-01-15 12:00:39	Change ID=sh ~ Reason=
wM8wEHWavvqND		edf	B	ItemRevision	__Check_Out	shen	2002-01-15 12:00:35	Change ID=sh ~ Reason=
wM8wEHWavvqND	temp2	edf		Item	__Check_In	shen	2002-01-15 12:00:35	Change ID=sh ~ Reason=

Audit data table

The files are described as follows:

TcAuditLog.xml

XML data source of audit records. The file is produced by executing the **combine_audit_files.pl** script.

TcAuditLog.xsl

XML style sheet for displaying the **TcAuditLog.xml** file in the Microsoft Internet Explorer Web browser.

TcAuditLogSchema.xsd

XML schema for defining XML data structure and data types.

TcAuditLog.js

JavaScript for adding dynamic effects to the HTML presentation.

If you use the **TcAuditLog.xml** for purposes other than displaying the audit log on a Web browser, you can modify any of the files. For example, you can modify the **TcAuditLog.xml** file so that the style sheet or schema is not loaded.

None of the files introduced in this section (**combine_audit_files.pl**, **TcAuditLog.xml**, **TcAuditLog.xsl**, **TcAuditLogSchema.xsd**, and **TcAuditLog.js**) require a Teamcenter environment. You can run them anywhere as long as you have Perl and Microsoft Internet Explorer.

define_auditdefs

Note This utility is deprecated and will be removed in a future version.

Creates **AuditDefinition** objects in the Teamcenter database. It scans the input file given by the argument **-f** argument.

INPUT FILE FORMAT

The input file contains records to define each AuditDefinition object in the database. Each record is separated by a blank line and conforms to the following format:

```
TYPE_NAME=object-type-name
CLASS_NAME=object-class-name
EVENT_TYPE=event-type-name
PROP_COUNT=property-count /* number of PROP_NAME entries */
PROP_NAME=property-list /* Optional entry */
PROP_NAME=property-list /* Optional entry */
MAX_DAY=max-days-kept
MEDIA_NAME=media-name /* Optional entry */
HANDLER_ID=handler-id /* Optional entry */
```

Tip You can see the list of available events in the **Event Type** editor of Business Modeler IDE.

For more information, see the [Business Modeler IDE Guide](#).

SYNTAX

```
define_auditdefs [-u=user-id {-p=password | -pf=password-file} -g=group]
[-v] [-f=input-file-name] [-h]
```

ARGUMENTS

-u

Specifies the user ID.

This is generally **infodba** or another user with administration privileges. If this argument is used without a value, the operating system user name is used.

Note If Security Services single sign-on (SSO) is enabled for your server, the **-u** and **-p** arguments are authenticated externally through SSO rather than being authenticated against the Teamcenter database. If you do not supply these arguments, the utility attempts to join an existing SSO session. If no session is found, you are prompted to enter a user ID and password.

-p

Specifies the password.

If used without a value, the system assumes a null value.

If this argument is not used, the system assumes the *user-ID* value to be the password.

This argument is mutually exclusive with the **-pf** argument.

-pf

Specifies the password file. If used without a value, the system assumes a null value. If this argument is not used, the system assumes the *user-ID* value to be the password.

For more information about managing password files, see [Manage password files](#).

This argument is mutually exclusive with the **-p** argument.

-g

Specifies the group associated with the user.

If used without a value, the user's default group is assumed.

-f

Specifies the name of the text file containing the audit definition records.

-v

Specifies verbose mode.

-h

Displays help for this utility.

ENVIRONMENT

As specified in [Manually configuring your environment for Teamcenter utilities](#).

FILES

As specified in [Log files produced by Teamcenter](#).

RESTRICTIONS

None.

EXAMPLES

None.

pom_audit_manager

Allows a site to configure a list of users for whom failed authentication attempts must be logged and provides the ability to later extract that information for analysis.

SYNTAX

```
pom_audit_manager [-u=user-id {-p=password | -pf=password-file} -g=group]
[-install ]
[-report [-before]]
[-purge [-before]]
[-auditable]
[-set event-name user-name ON|OFF ]
[-delete event-name user-name]
[-h]
```

ARGUMENTS

-u

Specifies the user ID.

This is generally **infodba** or another user with administration privileges. If this argument is used without a value, the operating system user name is used.

Note

If Security Services single sign-on (SSO) is enabled for your server, the **-u** and **-p** arguments are authenticated externally through SSO rather than being authenticated against the Teamcenter database. If you do not supply these arguments, the utility attempts to join an existing SSO session. If no session is found, you are prompted to enter a user ID and password.

-p

Specifies the password.

If used without a value, the system assumes a null value. This argument is mutually exclusive with the **-pf** argument.

If this argument is not used, the system assumes the *user-ID* value to be the password.

-pf

Specifies the password file. If used without a value, the system assumes a null value. If this argument is not used, the system assumes the *user-ID* value to be the password.

For more information about managing password files, see [Manage password files](#).

This argument is mutually exclusive with the **-p** argument.

-g

Specifies the group associated with the user.

If used without a value, the user's default group is assumed.

-install

Installs audit functionality.

-report [-before]

Shows audit reports. Optionally, you can use the **-before** argument to show only those records before the specified date.

The **-before** argument must be followed by an 8 digits ISO date such as 20071225 for Christmas day 2007.

-purge [-before]

Permanently removes event records from the user access logging list. Optionally, you can use the **-before** argument to filter purging to only those records before the specified date.

The **-before** argument must be followed by an 8 digits ISO date such as 20071225 for Christmas day 2007.

-auditable

Shows which events are auditable.

Currently **POM_AUDIT_purge_audit**, **POM_AUDIT_bad_password_login**, **POM_AUDIT_bad_password_check**, **POM_AUDIT_change_password** events are auditable.

-set event-name user-name ON|OFF

Adds or removes audit functionality on an event.

-delete event-name user-name

Deletes audit logs for specific events and users.

You can specify the **ALL** value for either event-name or user-name.

Currently **POM_AUDIT_purge_audit**, **POM_AUDIT_bad_password_login**, **POM_AUDIT_bad_password_check**, **POM_AUDIT_change_password** events are auditable.

-h

Displays help for this utility.

ENVIRONMENT

Standard runtime environment only.

FILES

None.

RESTRICTIONS

None.

EXAMPLES

- To see audit reports before a particular date:

```
pom_audit_manager -u=admin -p=admin -g=dba
-report -before 20071201
```

- To add audit functionality on an event:

```
pom_audit_manager -u=admin -p=admin -g=dba
-set POM_AUDIT_bad_password_login ON
```

Backup and Recovery

backup_modes

Manages the hot backup of the Teamcenter database and volumes by third-party backup systems. Use hot backup to avoid shutting down Teamcenter for routine backups, and to run the system in a near-continuous mode. Manage hot backup functionality by using this utility to set different backup modes (read-only, blobby volume, normal) on Teamcenter volumes.

SYNTAX

backup_modes [-u=*user-id* {-p=*password* | -pf=*password-file*} -g=*group*] [-pf=*password-file*] [-m= {rdonly | normal | blobby | current}] [-f= *opencnt*] [-h]

ARGUMENTS

-u

Specifies the user ID.

This is generally **infodba** or another user with administration privileges. If this argument is used without a value, the operating system user name is used.

Note

If Security Services single sign-on (SSO) is enabled for your server, the **-u** and **-p** arguments are authenticated externally through SSO rather than being authenticated against the Teamcenter database. If you do not supply these arguments, the utility attempts to join an existing SSO session. If no session is found, you are prompted to enter a user ID and password.

-p

Specifies the password.

If used without a value, the system assumes a null value.

If this argument is not used, the system assumes the *user-ID* value to be the password.

This argument is mutually exclusive with the **-pf** argument.

-pf

Specifies the password file. If used without a value, the system assumes a null value. If this argument is not used, the system assumes the *user-ID* value to be the password.

For more information about managing password files, see [Manage password files](#).

This argument is mutually exclusive with the **-p** argument.

-g

Specifies the group associated with the user.

If used without a value, the user's default group is assumed.

-m

Sets Teamcenter volumes to read-only, normal, or blobby mode. This argument can also be used to obtain the current backup mode.

rdonly

Places Teamcenter into read-only state. This state holds writing files to the volume during backup.

normal	Places Teamcenter back in normal mode from read-only or blobby volume mode.
blobby	Places Teamcenter in blobby (temporary) volume mode. Teamcenter can be switched into this mode after the third-party backup software takes a snapshot of the data. This allows continuous Teamcenter availability.
current	Returns the current Teamcenter mode.
-f	Obtains information about the Teamcenter volume files opened for the write operation.
-h	Displays help for this utility.

ENVIRONMENT

The proper values must be set for the following preferences:

- **blobbyVolume_NT**
- **blobbyVolume_UNX**
- **TC_enable_backup_modes**

FILES

As specified in *Log files produced by Teamcenter*.

RESTRICTIONS

- Before setting Teamcenter to blobby volume mode, ensure that Teamcenter is in read-only mode.
- You must assign values to the **blobbyVolume_NT** and **blobbyVolume_UNX** preferences even if you are not operating in a heterogeneous environment.

EXAMPLES

- To place Teamcenter volumes in read-only mode, enter the following command:

```
backup_modes -u=infodba -p=password -g=dba -m=rdbonly
```
- To place Teamcenter volumes in normal mode, enter the following command:

```
backup_modes -u=infodba -p=password -g=dba -m=normal
```
- To place Teamcenter volumes in blobby volume mode, enter the following command:

```
backup_modes -u=infodba -p=password -g=dba -m=blobby
```
- To obtain the current Teamcenter backup mode, enter the following command:

```
backup_modes -u=infodba -p=password -g=dba -m=current
```
- To obtain information about Teamcenter volume files opened for write, enter the following command:

```
backup_modes -u=infodba -p=password -g=dba -f=openent
```

backup_xmlinfo

Provides information about Teamcenter volumes defined for a site in XML format. Third-party backup systems require this information for 24x7 hot backup of Teamcenter volumes and databases. The program creates two output files, **backup.xml** and **backup.dtd**, in the directory from which the utility is executed.

SYNTAX

backup_xmlinfo [-u=*user-id* {-p=*password* | -pf=*password-file* -g=*group*} -h

ARGUMENTS**-u**

Specifies the user ID.

This is generally **infodba** or another user with administration privileges. If this argument is used without a value, the operating system user name is used.

Note

If Security Services single sign-on (SSO) is enabled for your server, the **-u** and **-p** arguments are authenticated externally through SSO rather than being authenticated against the Teamcenter database. If you do not supply these arguments, the utility attempts to join an existing SSO session. If no session is found, you are prompted to enter a user ID and password.

-p

Specifies the password.

If used without a value, the system assumes a null value.

If this argument is not used, the system assumes the *user-ID* value to be the password.

This argument is mutually exclusive with the **-pf** argument.

-pf

Specifies the password file. If used without a value, the system assumes a null value. If this argument is not used, the system assumes the *user-ID* value to be the password.

For more information about managing password files, see [Manage password files](#).

This argument is mutually exclusive with the **-p** argument.

-g

Specifies the group associated with the user.

If used without a value, the user's default group is assumed.

-h

Displays help for this utility.

ENVIRONMENT

As specified in [Manually configuring your environment for Teamcenter utilities](#).

FILES

As specified in [Log files produced by Teamcenter](#).

RESTRICTIONS

None.

EXAMPLES

Generate backup information for Teamcenter volumes by executing the following command:

```
backup_xmlinfo -u=infodba -p=infodba -g=dba
```

The following example illustrates a sample XML output file:

```
<?xml version="1.0" standalone="yes" ?>
<!-- Backup Info : XML File -->
<!DOCTYPE backupInfo SYSTEM "backup.dtd">
<backupInfo>
  <volumeinfo>
    <VolumeName>tokra_vol</VolumeName>
    <VolumeUid>036440ca0b1c558e9f42</VolumeUid>
    <NodeName>ustrwlsun002</NodeName>
    <UnixPath>/netap/tceapps/TCE/TCEvols/tokra_vol</UnixPath>
  </volumeinfo>
  <volumeinfo>
    <VolumeName>satish1_vol</VolumeName>
    <VolumeUid>037840d6b8ac558e9f42</VolumeUid>
    <NodeName>uslvw1097a011</NodeName>
    <WntPath>c:\satish1_vol</WntPath>
  </volumeinfo>
</backupInfo>
```

sfr_instances

Creates and deletes single file recovery instances.

SYNTAX

```
sfr_instances [-u=user-id {-p=password | -pf=password-file} -g=group]
[-d=datasettype] [-ou=owning-user] [-og=owning-group]
[-v=volume-name [-ib=any-previous-backupLabel] -b=new-backup-label [-f=function]
{create | delete | list} [-h]
```

ARGUMENTS

-u

Specifies the user ID.

This is generally **infodba** or another user with administration privileges. If this argument is used without a value, the operating system user name is used.

Note

If Security Services single sign-on (SSO) is enabled for your server, the **-u** and **-p** arguments are authenticated externally through SSO rather than being authenticated against the Teamcenter database. If you do not supply these arguments, the utility attempts to join an existing SSO session. If no session is found, you are prompted to enter a user ID and password.

-p

Specifies the password.

If used without a value, the system assumes a null value.

If this argument is not used, the system assumes the *user-ID* value to be the password.

This argument is mutually exclusive with the **-pf** argument.

-pf

Specifies the password file. If used without a value, the system assumes a null value. If this argument is not used, the system assumes the *user-ID* value to be the password.

For more information about managing password files, see [Manage password files](#).

This argument is mutually exclusive with the **-p** argument.

-g

Specifies the group associated with the user.

If used without a value, the user's default group is assumed.

-d

Specifies the dataset type to which the specified function applies.

-ou

Specifies the owning user of the datasets to which the specified function applies.

-og

Specifies the owning group of the datasets to which the specified function applies.

-v

Specifies the volume to which the specified function applies.

-ib

Specifies the previous backup label.

-b

Specifies the backup label associated with the create, list or delete function. Use **-b=ALL** to delete all single file recovery instances.

-f= *function*

Specifies the function for the utility. **create** creates the single file recovery instance. **delete** deletes the single file recovery instances.

-h

Displays help for this utility.

ENVIRONMENT

As specified in [Manually configuring your environment for Teamcenter utilities](#). In case this is not already initialized, set the proper values for the following variables to enable further recovery:

TC_sfr_recovery_interval

TC_sfr_process_life_time

FILES

As specified in [Log files produced by Teamcenter](#).

RESTRICTIONS

None, however it is a good practice to run this utility after putting Teamcenter in read-only mode and before putting Oracle in Hot backup mode.

EXAMPLES

- To create single file recovery instances associated with the **backup_1** backup label, enter the following command:

```
sfr_instances -u=infodba -p=password -g=dba -b=backup_1 -f=create
```

- To delete single file recovery instances associated with the **backup_2** backup label, enter the following command:

```
sfr_instances -u=infodba -p=password -g=dba -b=backup_2 -f=delete
```

- To delete all single file recovery instances in the database, enter the following command:

```
sfr_instances -u=infodba -p=password -g=dba -b=ALL -f=delete
```

Dispatcher

dispatcher_create_rqst

Provides the ability to create a dispatcher request using command line arguments.

SYNTAX

The syntax of the **dispatcher_create_rqst** utility has two forms:

```
dispatcher_create_rqst
-i=item-ID
-r=revision-ID
[-rn=relation]
-dn=dataset-name
[-dv=dataset-version-number]
-dt=dataset-type-name
-pr= 1 | 2 | 3]
-pn=translator-provider-name
-tn=service-name
-ty=type-string
[-ta1=translation-argument1
  [-ta2=translation-argument2
    [-ta3=translation-argument3]]]
[-u=user-id -p=password | -pf=password-file -g=group]
-verbose | -debug
```

OR

```
dispatcher_create_rqst
-f=path-name
-dt=dataset-type-name
-pr= 1 | 2 | 3]
-pn=service-provider-name
-tn=service-name
-ty=type-string
[-u=user-id -p=password | -pf=password-file -g=group]
-verbose | -debug
```

ARGUMENTS

Note If a relation is not specified, the **IMAN_specification** relation is used.

-u

Specifies the user ID.

This is generally **infodba** or another user with administration privileges. If this argument is used without a value, the operating system user name is used.

Note If Security Services single sign-on (SSO) is enabled for your server, the **-u** and **-p** arguments are authenticated externally through SSO rather than being authenticated against the Teamcenter database. If you do not supply these arguments, the utility attempts to join an existing SSO session. If no session is found, you are prompted to enter a user ID and password.

-p

Specifies the password.

If used without a value, the system assumes a null value.

If this argument is not used, the system assumes the *user-ID* value to be the password.

This argument is mutually exclusive with the **-pf** argument.

-pf

Specifies the password file. If used without a value, the system assumes a null value. If this argument is not used, the system assumes the *user-ID* value to be the password.

For more information about managing password files, see [Manage password files](#).

This argument is mutually exclusive with the **-p** argument.

-g

Specifies the group associated with the user.

If used without a value, the user's default group is assumed.

-i

Specifies the item.

-r

Specifies the item revision.

-rn

Specifies the relation name to be used to find the dataset for the given item revision. This argument is optional. If the relation is not specified, the value of **IMAN_specification** is used.

-dn

Specifies the dataset name. This argument is optional.

-dv

Specifies the dataset version. This argument is optional. If no version number is specified, the latest version is used.

-dt

Specifies the type of the dataset to be translated

-pr

Specifies the translation priority. Accepted values are **1**, **2**, or **3** corresponding to low, medium and high translation scheduler priority.

-pn

Specifies the name of the translator provider, for example, Siemens.

-tn

Specifies the name of the translator service, for example **ideastojt**.

-ty

Specifies the type name, for example **COMMANDLINE**.

-verbose

Provides additional information. This argument is optional.

-f

Specifies an input file used to create one or more dispatcher requests. This argument is used in lieu of the item, revision, relation name, dataset name, and dataset version arguments. This argument is optional.

The format for the input file is as follows:

```
<item ID>,<revision ID>,[relation name],[dataset name],
[version number]
```

Note Commas are required.

-ta1

Specifies a translation argument.

ENVIRONMENT

As specified in [Manually configuring your environment for Teamcenter utilities](#).

FILES

As specified in [Log files produced by Teamcenter](#).

RESTRICTIONS

None.

EXAMPLES

- To create a dispatcher request for the latest version of the I-deas part dataset related to the **Block/A** item revision, enter the following command on a single line:


```
dispatcher_create_rqst -u=infodba -p=infodba -g=dba
-i=Block -r=A -dt=IdeasPart -pr=2 -pn=Siemens
-tn=ideastojt -ty=COMMAND_LINE
```
- To create a dispatcher request for version 2 of the **Block/A** item revision, enter the following command on a single line:


```
dispatcher_create_rqst -u=infodba -p=infodba -g=dba
-i=Block -r=A -dv=2 -dt=IdeasPart -pr=2 -pn=Siemens
-tn=ideastojt -ty=COMMAND_LINE
```
- To create two Siemens PLM Software **ideastojt** dispatcher requests with a priority of 2:
 - o One request for the latest version of the **Block** dataset, an **IdeasPart** or **IdeasAssembly** type dataset, associated with item revision **Block/A** by an **IMAN_specification** relation;
 - o One for the latest version of the **Asm** dataset, an **IdeasPart** or **IdeasAssembly** type dataset associated with item revision **Asm/A** by an **IMAN_specification** relation, enter the following command on a single line:

```
dispatcher_create_rqst -u=infodba -p=infodba -g=dba
-f=ctrl -dt=IdeasPart,IdeasAssembly -pr=2 -pn=Siemens
-tn=ideastojt -ty=COMMAND_LINE
```

The lines in the input file are:

```
Block,A,,Block,
Asm,A,,Asm,
```

Note All parts in the file are subject to the same translation, because the translator is specified on the command line.

dispatcher_util

Allows you to list, delete or resubmit dispatcher requests.

SYNTAX

```
dispatcher_util
[-u=user-id {-p=password | -pf=password-file} -g=group]
-a=list | delete | resubmit
[-force]
[-export=file:file path]
[-taskid=dispatcher task ID | file:file path]
[-provider=provider name]
[-service=service name]
[-priority=priority number]
[-state=dispatcher state]
[-h]
```

ARGUMENTS

-u

Specifies the user ID.

This is generally **infodba** or another user with administration privileges. If this argument is used without a value, the operating system user name is used.

Note If Security Services single sign-on (SSO) is enabled for your server, the **-u** and **-p** arguments are authenticated externally through SSO rather than being authenticated against the Teamcenter database. If you do not supply these arguments, the utility attempts to join an existing SSO session. If no session is found, you are prompted to enter a user ID and password.

-p

Specifies the password.

If used without a value, the system assumes a null value.

If this argument is not used, the system assumes the *user-ID* value to be the password.

This argument is mutually exclusive with the **-pf** argument.

-pf

Specifies the password file. If used without a value, the system assumes a null value. If this argument is not used, the system assumes the *user-ID* value to be the password.

For more information about managing password files, see [Manage password files](#).

This argument is mutually exclusive with the **-p** argument.

-g

Specifies the group associated with the user.

If used without a value, the user's default group is assumed.

-a

Performs the list, delete or resubmit actions.

Choose from the following action values:

- **list**: Lists dispatcher requests.
- **delete**: Deletes dispatcher requests.
- **resubmit**: Resubmits dispatcher requests.

-force

Forces an action without any prompts.

-export

Exports the information in a file.

-taskid

Specifies the task ID of the dispatcher request.

-provider

Specifies the name of the dispatcher provider, for example, Siemens.

-service

Specifies the service name, for example, tozipfile.

-priority

Specifies the dispatcher priority. Accepted values are **1**, **2**, or **3** corresponding to low, medium and high priority.

-state

Specifies the dispatcher state.

-h

Displays help for this utility.

ENVIRONMENT

As specified in [Manually configuring your environment for Teamcenter utilities](#).

FILES

As specified in [Log files produced by Teamcenter](#).

RESTRICTIONS

None.

EXAMPLES

- List all high priority dispatcher requests.

```
dispatcher_util -u=infodba -p=infodba -a=list -priority=3
```

- Delete all dispatcher requests in the **INITIAL** state.

```
dispatcher_util -u=infodba -p=infodba -a=list -state=INITIAL
```

- Resubmit all **tozipfile** dispatcher requests.

```
dispatcher_util -u=infodba -p=infodba -a=list -service=tozipfile
```

SS_GenSvcRqst

Generates a service request for all datasets that have a translation configuration specified but either do not have a target dataset present or whose target dataset is out of date.

This utility starts at the specified point and recurses down through the child items, item revisions, folders, and so on.

SYNTAX

SS_GenSvcRqst [-u=*user-id* {-p=*password* | -pf=*password-file*} -g=*group*]
 -folder=*folder-name* [-service=*service-name*] [-priority= 1 | 2 | 3]

ARGUMENTS

-u

Specifies the user ID.

This is generally **infodba** or another user with administration privileges. If this argument is used without a value, the operating system user name is used.

Note

If Security Services single sign-on (SSO) is enabled for your server, the **-u** and **-p** arguments are authenticated externally through SSO rather than being authenticated against the Teamcenter database. If you do not supply these arguments, the utility attempts to join an existing SSO session. If no session is found, you are prompted to enter a user ID and password.

-p

Specifies the password.

If used without a value, the system assumes a null value.

If this argument is not used, the system assumes the *user-ID* value to be the password.

This argument is mutually exclusive with the **-pf** argument.

-pf

Specifies the password file. If used without a value, the system assumes a null value. If this argument is not used, the system assumes the *user-ID* value to be the password.

For more information about managing password files, see [Manage password files](#).

This argument is mutually exclusive with the **-p** argument.

-g

Specifies the group associated with the user.

If used without a value, the user's default group is assumed.

-folder

Specifies the folder from which to start the integration.

-service

Specifies the service name for submitting. Specifying this argument partially limits the dataset types based on configuration. This argument is optional.

-priority

Specifies the priority to apply to the new task. This value overrides the priority specified in the translation configuration object. This argument is optional. Allowed values are **1** (low), **2** (medium), or **3** (high).

-h

Displays help for this utility.

ENVIRONMENT

This utility must be executed from the Teamcenter console.

FILES

None.

RESTRICTIONS

The user must have permission to create tasks and write to specified objects within the database.

Migration

convert_distribution_lists

Converts distribution lists to alias lists. Also allows users to create an alias list by importing data from a text file.

SYNTAX

```
convert_distribution_lists [-u=user-id {-p=password |  
-pf=password-file} -g=group]  
[-all] [-delete] [-dist_list_name=distribution-list-name]  
[-import_file=file-name] [-import=file-name] [-new_list_name=alias-list-name]  
[-overwrite] [-h]
```

ARGUMENTS

-u

Specifies the user ID.

This is generally **infodba** or another user with administration privileges. If this argument is used without a value, the operating system user name is used.

Note

If Security Services single sign-on (SSO) is enabled for your server, the **-u** and **-p** arguments are authenticated externally through SSO rather than being authenticated against the Teamcenter database. If you do not supply these arguments, the utility attempts to join an existing SSO session. If no session is found, you are prompted to enter a user ID and password.

-p

Specifies the password.

If used without a value, the system assumes a null value.

If this argument is not used, the system assumes the *user-ID* value to be the password.

This argument is mutually exclusive with the **-pf** argument.

-pf

Specifies the password file. If used without a value, the system assumes a null value. If this argument is not used, the system assumes the *user-ID* value to be the password.

For more information about managing password files, see [Manage password files](#).

This argument is mutually exclusive with the **-p** argument.

-g

Specifies the group associated with the user.

If used without a value, the user's default group is assumed.

-all

Converts all the distribution lists to alias lists.

-delete

Deletes distribution lists that were converted to alias lists.

-dist_list_name

Converts a specified distribution list to an alias list.

-import

Creates an alias list from the addresses specified in the file. The format of the ASCII file is:

```
abc1@xyz.com
abc2@xyz.com
abc3@xyz.com
```

-new_list_name

Specifies the name of the new alias list. This argument must be used with the **-import** option.

-overwrite

Overwrites the existing alias list.

-h

Displays help for this utility.

ENVIRONMENT

As specified in [Manually configuring your environment for Teamcenter utilities](#).

FILES

As specified in [Log files produced by Teamcenter](#).

RESTRICTIONS

None.

EXAMPLES

- To convert all distribution lists in the database to alias lists, enter the following command on a single line:

```
convert_distribution_lists -all
```

- To create an alias list using the distribution list **marketing_list** and then delete the distribution list:

```
convert_distribution_lists -dist_list_name=marketing_list -delete
```

- To create a new alias list with the name **Local_Alias_List**, populate the list with the addresses listed in the **address_local.txt** file, and overwrite the existing address list, enter the following command on a single line:

```
convert_distribution_lists -import=address_local.txt
-new_file_name=Local_Alias_List -overwrite
```

move_mso_forms

Finds all forms of type **OfficeDocForm** that are directly attached to folders, items, or item revisions and moves them to a corresponding dataset as a named reference when upgrading Teamcenter Engineering 8.x and 9.x databases to a Teamcenter 10.1 database. These forms are used for property synchronization between Microsoft Office and the Teamcenter database.

Note This utility is called by the upgrade script when upgrading from a previous version of Teamcenter Engineering to Teamcenter 10.1.

SYNTAX

move_mso_forms [-u=*user-id* {-p=*password* | -pf=*password-file*} -g=*group*] [-h]

ARGUMENTS

-u

Specifies the user ID.

This is generally **infodba** or another user with administration privileges. If this argument is used without a value, the operating system user name is used.

Note If Security Services single sign-on (SSO) is enabled for your server, the **-u** and **-p** arguments are authenticated externally through SSO rather than being authenticated against the Teamcenter database. If you do not supply these arguments, the utility attempts to join an existing SSO session. If no session is found, you are prompted to enter a user ID and password.

-p

Specifies the password.

If used without a value, the system assumes a null value.

If this argument is not used, the system assumes the *user-ID* value to be the password.

This argument is mutually exclusive with the **-pf** argument.

-pf

Specifies the password file. If used without a value, the system assumes a null value. If this argument is not used, the system assumes the *user-ID* value to be the password.

For more information about managing password files, see [Manage password files](#).

This argument is mutually exclusive with the **-p** argument.

-g

Specifies the group associated with the user.

If used without a value, the user's default group is assumed.

-h

Displays help for this utility.

ENVIRONMENT

As specified in [Manually configuring your environment for Teamcenter utilities](#).

FILES

As specified in *Log files produced by Teamcenter*.

RESTRICTIONS

None.

Portfolio, Program, and Project Management

create_project

Creates projects in the database based on command line input or input from a text file.

SYNTAX

```
create_project [-u=user-id {-p=password | -pf=password-file} -g=group]
{-id=project-id -name=project-name
[-desc=project-description -status=A | I
-teams=group1~role1~user1~group2~role2~user2...
[-privileged=group1~role1~user1~group2~role2~user2... ]
| [-teamadmin=group~role~user]}
{-input=full-path-to-input-file [-delimiter=delimiter-character]} -h
```

ARGUMENTS

-u

Specifies the user ID.

This is generally **infodba** or another user with administration privileges. If this argument is used without a value, the operating system user name is used.

Note

If Security Services single sign-on (SSO) is enabled for your server, the user and password arguments are authenticated externally through SSO rather than being authenticated against the Teamcenter database. If you do not supply these arguments, the utility attempts to join an existing SSO session. If no session is found, you are prompted to enter a user ID and password.

-p

Specifies the user's password.

If used without a value, the system assumes a null value. If this argument is not used, the system assumes the *user-ID* value to be the password.

This argument is mutually exclusive with the **-pf** argument.

-pf

Specifies the password file. If used without a value, the system assumes a null value. If this argument is not used, the system assumes the *user-ID* value to be the password.

For more information about managing password files, see [Manage password files](#).

This argument is mutually exclusive with the **-p** argument.

-g

Specifies the group associated with the user.

If used without a value, the user's default group is assumed.

-id

Specifies the ID of the project.

-name

Specifies the name of the project.

-desc

Specifies the project description.

-status

Specifies the status; either active (**A**) or inactive (**I**).

-teams

Specifies group members to be on the project team. This argument accepts valid user, role, and group names. Use the tilde character (~) as a delimiter when specifying group, role, and user, as follows:

```
-teams=group1~role1~user1~group2~role2~user2...
```

In addition, you can specify all members in a group as follows:

```
-teams=group1~*~*
```

-privileged

Defines privileged group members using the following format:

```
-privileged=group1~role1~user1...
```

-teamadmin

Defines the team administrator using the following format:

```
-teamadmin=group~role~user
```

If not specified, the default team administrator is the logged-on user who is running the utility.

-input

Specifies the path to a text file containing multiple entries of project id, project name, project description, teams, privileged members, and optional team administrator. Use this option to create multiple projects.

The syntax of the input file is as follows:

```
id|name|desc|A or I|group1~role1~user1~group2~role2~user2...|
group1~role1~user1
id|name|desc|A or I|group1~role1~user1~group2~role2~user2...|
group1~role1~user1
id|name|desc|A or I|group1~role1~user1~group2~role2~user2...|
group1~role1~user1
```

-delimiter

Specifies the delimiting character used in the input file to parse ID, name, description, status, and teams.

-h

Displays help for this utility.

ENVIRONMENT

As specified in [Manually configuring your environment for Teamcenter utilities](#).

FILES

Input file must conform to the syntax described in the **-input** argument description in the *Arguments* section.

RESTRICTIONS

None.

**RETURN
VALUES**

Return value upon success 0
Return value upon failure >1

EXAMPLES

- To create a project with ID **123456**, named **ABC Car 123 Model**, description **A high end version of ABC Car**, with an active status assigned to **Car 1** and **Car 2** groups, enter the following command on a single line:

```
create_project -u=user-id -p=password -g=group-name -id=123456  
-name="ABC Car 123 Model" -desc="A high end version of ABC Car  
-status=A -teams="Car 1"~Designer~Smith
```

- To create projects from an input file, enter the following command on a single line:

```
create_project  
-u=user-id -p=password -g=group-name -input=/tmp/project_input_file.txt
```

update_project_data

Updates project data in the Teamcenter database.

Note The **-f=update** option initiates the update of all project-related data in a database. This process can take a long time depending on the number of objects assigned to projects. When the project ID (**-pid**) is specified for one or more projects, the action applies only the given projects. When updating specific projects, other project-related data may also need to be updated by running the utility again.

SYNTAX

```
update_project_data [-u=user-id] {-p=password | -pf=password-file} -g=group]  
[-f=function] [-force] [-t=relation-type-name1 [, relation-type-name2...]]  
[-pid=project-ID1 [, project-ID2...]] [-h]
```

ARGUMENTS**-u**

Specifies the user ID.

This is generally **infodba** or another user with administration privileges. If this argument is used without a value, the operating system user name is used.

Note If Security Services single sign-on (SSO) is enabled for your server, the user and password arguments are authenticated externally through SSO rather than being authenticated against the Teamcenter database. If you do not supply these arguments, the utility attempts to join an existing SSO session. If no session is found, you are prompted to enter a user ID and password.

-p

Specifies the user's password.

If used without a value, the system assumes a null value. If this argument is not used, the system assumes the *user-ID* value to be the password.

This argument is mutually exclusive with the **-pf** argument.

-pf

Specifies the password file. If used without a value, the system assumes a null value. If this argument is not used, the system assumes the *user-ID* value to be the password.

For more information about managing password files, see [Manage password files](#).

This argument is mutually exclusive with the **-p** argument.

-g

Specifies the group associated with the user.

If used without a value, the user's default group is assumed.

-f=

Specifies the function performed by the utility. Must be one of the following options:

update

Updates project-related data and is generally used after site propagation rules are modified. This function can also be used to cleanse project-related data that may have been corrupted by system crashes.

This is the default function for this utility.

list

Lists relation types used for site propagation rules. It also lists the IDs, project administrator, and status of projects defined in the database.

add

Adds relation types of the site propagation rules for project assignment propagation and updates project data to reflect the change in the rules. The relation types are given as a comma-separated string.

remove

Removes relation types of the site propagation rules for project assignment propagation and updates all project data to reflect the change in the site propagation rules. The relation types are given as a comma-separated string.

delete

Deletes one or more projects identified by the **-pid** option.

bomviewon

Enables BOM view propagation. This allows BOM views and BOM view revisions of an item to be added to the project automatically when the item is added to a project.

bomviewoff

Disables BOM view propagation. By disabling BOM view propagation, BOM views and BOM view revisions of an item are not included in the project by default when the item is added to the project.

-h

Displays help for this utility.

ENVIRONMENT

As specified in [Manually configuring your environment for Teamcenter utilities](#).

FILES

As specified in [Log files produced by Teamcenter](#).

EXAMPLES

The following examples illustrate the use of the **update** function:

- Enter the following command to unconditionally update the specified project in the database using the current site propagation rules:

```
-f=update -pid=project-id
```

- Enter the following command to unconditionally update the specified list of projects using the current site propagation rules:

```
-f=update -pid=project-id1[,project-id2...]
```

The project IDs are given as a comma-separated string. For example, **-pid="Proj4000,Proj5000"** specifies that the action is performed on two projects: **Proj4000** and **Proj5000**.

- Enter the following command to update all projects in the database using the current site propagation rules:

```
-f=update
```

This command is normally used to update all project data after site propagation rules have been modified. The update algorithm updates project data for objects with a last project assignment date prior to the last site propagation rule modification date.

- Enter the following command to unconditionally update all projects in the database using the current site propagation rules:

```
-f=update -force
```

Note If the database contains a large number of projects, processing time could be considerable.

The following example illustrates the use of the **list** function:

- Enter the following command to list the project administrator, project status, and relation types used for site propagation rules:

```
-f=list
```

Note This function is not used with other parameters.

The following examples illustrate the use of the **add** function:

- Enter the following command to append relation types to the site propagation rules and update all project data in the database:

```
-f=add -t=relation-type-name1[,relation-type-name2...]
```

- Enter the following command to append relation types to the site propagation rules and update the project data related to the projects specified by the **-pid** option:

```
-f=add -t=relation-type-name1[,relation-type-name2...]
-pid=project-id1[,project-id2...]
```

This command is used under special circumstances when a user wants to update specific projects prior to updating the entire database. The database must be updated after the specific projects have been updated to ensure completeness of the project data. To update the remaining project data, run the utility using the **-update** option.

The following examples illustrate the use of the **remove** function:

- Enter the following command to remove relation types from the site propagation rules and update all project data in the database:

```
-f=remove -t=relation-type-name1[,relation-type-name2...]
```

- Enter the following command to remove relation types from the site propagation rules for specific projects:

```
-f=remove -t=relation-type-name1[,relation-type-name2...]
-pid=project-id1[,project-id2...]
```

This command is used under special circumstances when a user wants to update specific projects prior to updating the entire database. The database must be

updated after the specific projects have been updated to ensure completeness of the project data. To update the remaining project data, run the utility using the **-update** option.

The following example illustrates the use of the **delete** function:

- Enter the following command to delete specific projects and remove all project data associated with those projects:

```
-f=delete -pid=project-id1[,project-id2...]
```

Note

Processing time can be considerable depending on the number of objects in each project.

Server manager

installmgr.bat

Installs the J2EE-based server manager for four-tier environments as a Windows service. A copy of this script is created in the directory *TC_ROOT\pool_manager\confs\configuration-name* by Teamcenter Environment Manager (TEM) for each installed server manager instance. Replace *configuration-name* with the server manager configuration name. The name consists of the TEM configuration name and the server manager's pool ID. This script is run automatically by TEM but can be run manually if required.

SYNTAX**installmgr****ARGUMENTS**

None.

ENVIRONMENT

None.

FILES

None.

RESTRICTIONS

None.

EXAMPLES

To manually install a J2EE server manager named **PoolA** in a Teamcenter configuration named **MyConfig**, enter the following command on a single line:

```
D:\tc_root\pool_manager\confs\MyConfig_PoolA>installmgr
```

uninstallmgr.bat

Uninstalls the J2EE-based server manager for four-tier environments as a Windows service. A copy of this script is created in the directory *TC_ROOT\pool_manager\confs\configuration-name* by Teamcenter Environment Manager (TEM) for each installed server manager instance. Replace *configuration-name* with the server manager configuration name. The name consists of the TEM configuration name and the server manager's pool ID.

SYNTAX**uninstallmgr****ARGUMENTS**

None.

ENVIRONMENT

None.

FILES

None.

RESTRICTIONS

None.

EXAMPLES

To manually uninstall a J2EE server manager named **PoolA** in a Teamcenter configuration named **MyConfig**, enter the following command on a single line:

```
D:\tc_root\pool_manager\confs\MyConfig_PoolA>uninstallmgr
```

mgrstop

Shell script to cleanly stop the server manager for J2EE four-tier environments. A copy of this script is created in the directory *TC_ROOT/pool_manager/confs/configuration-name* by Teamcenter Environment Manager (TEM) for each installed server manager instance. Replace *configuration-name* with the server manager name. The name consists of the TEM configuration name and the server manager's pool ID. On UNIX/Linux systems, this script is launched by *rc.tc.mgr_config* when the stop action is requested.

Note This command stops a server manager instance; it does not mark a Windows service as stopped. If the server manager is running as a Windows service, the Windows Services Manager attempts to restart the manager if the service is configured for restarting. A server manager running as a Windows service must be stopped from the Windows Services Manager.

SYNTAX

mgrstop [-Dimmediate]

ARGUMENTS

-Dimmediate

Specifies whether the manager terminates all servers immediately rather than waiting for them to become idle (which is the default behavior).

ENVIRONMENT

None.

FILES

Files in the *TC_ROOT/pool_manager/confs/config* directory.

File	Description
mgr.tmp	Temporary file written by the manager at startup and deleted when it exits. The file contains the values of the POOL_ID and the JMX_HTTP_ADAPTOR_PORT environment variables, which identify the server manager pool and the port on which it listens.
password.txt (optional)	Contains the user name and the encrypted password for the server manager administration interface. This file exists only if the user name or password has been changed using the server manager administration interface.

RESTRICTIONS

This script must be executed on the machine where the server manager is running.

EXAMPLES

To immediately shut down a server manager named **PoolA** in a server manager configuration named **MyConfig**, enter the following command on a single line:

```
/tc_root/pool_manager/confs/MyConfig_PoolA>mgrstop -Dimmediate
```

rc.tc.mgr_<config>

Shell script to run the J2EE-based server manager for four-tier environments as a UNIX/Linux daemon. A uniquely-named copy of this script is created in the directory *TC_ROOT/pool_manager/confs/config* by Teamcenter Environment Manager (TEM) for each installed server manager instance. Replace *config* with the server manager name and pool ID. For example, if the server manager name is **MyConfig** and the pool ID is **PoolA**, the script has the following name and location:

TC_ROOT/pool_manager/confs/MyConfig_PoolA/rc.tc.mgr_MyConfig_PoolA

To install the daemon, this script must be deployed by the system administrator in the appropriate UNIX/Linux directory as a postinstallation step. The script is usually launched by the UNIX/Linux operating system in response to system administration commands.

SYNTAX

rc.tc.mgr_config *action*

ARGUMENTS

action

Specifies the daemon control action to be performed:

start

Starts the server manager daemon.

start_msg

Displays a message that the daemon is starting.

stop

Stops the server manager daemon without waiting for servers to become idle.

stop_msg

Displays a message that the daemon is stopping.

status

Displays a message indicating whether the server manager daemon is running.

log

Displays the full output of the server manager.

ENVIRONMENT

None.

FILES

None.

RESTRICTIONS

None.

EXAMPLES

To start a server manager named **PoolA** in a Teamcenter configuration named **MyConfig**, enter the following command on a single line:

```
/tc_root/pool_manager/confs/MyConfig_PoolA>rc.tc.mgr_MyConfig_PoolA start
```

Subscription Manager

purge_invalid_subscriptions

Provides the capability to delete invalid and expired subscriptions. For security reasons, only a system administrator can run this program. The user is also able to get the numbers of invalid and expired subscriptions without deleting them.

A subscription references a target object as an external reference. If the target gets deleted, the subscription becomes invalid. The user can interactively delete invalid subscriptions in the rich client interface. Finding a few invalid subscriptions among a large number of subscriptions in the table sometimes is not easy. In addition, subscriptions expire after their expiration dates.

SYNTAX

```
purge_invalid_subscriptions [-u=user-id] {-p=password |  
-pf=password-file} -g=group]  
[-report] [-e] [-h]
```

ARGUMENTS

-u

Specifies the user ID.

This is generally **infodba** or another user with administration privileges. If this argument is used without a value, the operating system user name is used.

Note

If Security Services single sign-on (SSO) is enabled for your server, the **-u** and **-p** arguments are authenticated externally through SSO rather than being authenticated against the Teamcenter database. If you do not supply these arguments, the utility attempts to join an existing SSO session. If no session is found, you are prompted to enter a user ID and password.

-p

Specifies the password.

If used without a value, the system assumes a null value.

If this argument is not used, the system assumes the *user-ID* value to be the password.

This argument is mutually exclusive with the **-pf** argument.

-pf

Specifies the password file. If used without a value, the system assumes a null value. If this argument is not used, the system assumes the *user-ID* value to be the password.

For more information about managing password files, see [Manage password files](#).

This argument is mutually exclusive with the **-p** argument.

-g

Specifies the group associated with the user.

If used without a value, the user's default group is assumed.

-report

Reports the numbers of invalid and expired subscriptions without deleting them.

-e

Deletes expired subscriptions in addition to deleting invalid subscriptions.

-h

Displays help for this utility.

ENVIRONMENT

As specified in [Manually configuring your environment for Teamcenter utilities](#).

FILES

As specified in [Log files produced by Teamcenter](#).

RESTRICTIONS

None.

EXAMPLES

- To delete invalid subscriptions, enter the following command on a single line:

```
purge_invalid_subscriptions -u=infodba -p=infodba -g=dba
```

- To delete invalid subscriptions and expired subscriptions, enter the following command on a single line:

```
purge_invalid_subscriptions -u=infodba -p=infodba -g=dba -e
```

- To report the numbers of invalid and expired subscriptions without deleting them, enter the following command on a single line:

```
purge_invalid_subscriptions -u=infodba -p=infodba -g=dba -report
```

- To display the help message, enter the following command on a single line:

```
purge_invalid_subscriptions -u=infodba -p=infodba -g=dba -h
```

System maintenance

clearlocks

Clears dead process locks from the database. Dead process locks typically occur when a Teamcenter session terminates abnormally. Process locks are set on an object when it is being modified or deleted. If a Teamcenter session does not terminate gracefully (by logging out), these locks can remain in place.

Dead process locks (locks held by dead sessions) can cause diverse problems that are often difficult to diagnose, and Teamcenter applications make every effort to eliminate or otherwise avoid them. Nevertheless, there are occasions when such dead process locks must be explicitly removed from the database, and the **clearlocks** utility is used for this purpose.

Note To use the **-assert_dead** or **-assert_all_dead** options, you must specify the administrator's user name, password, and group.

The **clearlocks** utility can only obtain general information about the processes in the lock table. Normally, the PID is pulled from the table and a kill is sent to the operating system. If the PID exists, the **Alive** count is incremented. If the PID does not exist, the **Dead** count is incremented. A **Remote PID** count indicates the process was started from a node other than the one that was used to run clearlocks. On some platforms, for example, Solaris, the kill returns a security violation and no specific information about the PID. If this occurs, the **Other** count is incremented.

SYNTAX

```
clearlocks [-verbose] [-node_names]
[[-assert_dead -u=user-id {-p=password | -pf=password-file} -g=group]
|
[-assert_all_dead -u=user-id {-p=password | -pf=password-file} -g=group]]
[-h]
```

Caution The **clearlocks** utility can be run with active Teamcenter sessions, provided that the **-assert_dead** or **-assert_all_dead** arguments are not used. By default, the **clearlocks** utility discriminates between valid and dead process locks; the **-assert_dead** and **-assert_all_dead** arguments defeat this feature.

ARGUMENTS

-u

Specifies the user ID.

This is generally **infodba** or another user with administration privileges. If this argument is used without a value, the operating system user name is used.

Note If Security Services single sign-on (SSO) is enabled for your server, the **-u** and **-p** arguments are authenticated externally through SSO rather than being authenticated against the Teamcenter database. If you do not supply these arguments, the utility attempts to join an existing SSO session. If no session is found, you are prompted to enter a user ID and password.

-p

Specifies the password.

If used without a value, the system assumes a null value.

If this argument is not used, the system assumes the *user-ID* value to be the password.

This argument is mutually exclusive with the **-pf** argument.

-pf

Specifies the password file.

For more information about managing password files, see [Manage password files](#).

This argument is mutually exclusive with the **-p** argument. One of the two mutually exclusive password elements is required.

-g

Specifies the group associated with the user.

If used without a value, the user's default group is assumed.

-verbose

Displays a summary of processes and states (dead, alive, and unknown). Locks associated with dead processes are cleared by the **clearlocks** utility, live processes are not cleared, and the unknown processes are all other processes.

-node_names

Lists nodes upon which the known processes exist.

-assert_dead

Asserts that all processes on a particular node are dead and clears all process locks held by sessions running on that node with the exception of Multi-Site Collaboration transfer locks. If any of those sessions are alive and in use, the locks held by those sessions are compromised.

Note To use this argument, you must enter the node name and the administrator's user name, password, and group. To clear Multi-Site Collaboration transfer locks, use the [export_recovery](#) utility.

-assert_all_dead

Asserts that all processes in the database are dead and clears all process locks with the exception of Multi-Site Collaboration transfer locks. If any of those sessions are alive and in use, the locks held by those sessions will be compromised. Additionally, this option performs a complete cleanup of the database lock tables, the **POM_TIMESTAMP** table, and reports on the sessions that were asserted to be dead.

Note To use this argument, you must enter the administrator's user name, password, and group. To clear Multi-Site Collaboration transfer locks, use the [export_recovery](#) utility.

-h

Displays help for this utility.

ENVIRONMENT

As specified in [Manually configuring your environment for Teamcenter utilities](#).

FILES

As specified in [Log files produced by Teamcenter](#).

RESTRICTIONS

- Do not run the **clearlocks** utility with the **-assert_dead** or **-assert_all_dead** arguments if there are any active Teamcenter sessions running. Any locks held by active sessions will be lost and these sessions can then potentially modify data for which they no longer hold modify locks.
- The **-assert_dead** and **-assert_all_dead** arguments are powerful and potentially destructive. Therefore, these arguments should only be used to clear process locks that cannot be cleared otherwise. For this reason, you must enter the administrator's user name, password, and group when using these arguments.

- The **clearlocks** utility cannot clear the **Transfer** lock type, only the **Modify** lock. To clear **Transfer** locks, use the **export_recovery** utility. This behavior is intended to prevent cases where objects that are being transferred are forcibly unlocked and thereby exposing them to the possibility of being modified when their ownership is being transferred.
- When running Clearlocks with the **assert_all_dead** or **assert_dead** option, you may see the message:

```
Notice: There are transfer locks detected indicating active
Multi-Site transfer transactions. All transfers need to complete
before the upgrade can safely continue. Ensure that
ensure_site_consistency is successfully executed for any identified
objects before running Clearlocks.
```

This message also appears when upgrading a database to a new release if there are existing transfer locks in the database.

EXAMPLES

- To clear process locks for dead sessions, enter the following command from that node:

```
$TC_ROOT/bin/clearlocks
```

- To obtain a list of all network nodes which have process locks set on the database, enter the following command:

```
$TC_ROOT/bin/clearlocks -node_names
```

- To clear all process locks (active and dead) on a single network node, in this example **ntssun9**, enter the following command:

```
$TC_ROOT/bin/clearlocks -assert_dead infodba infodba dba ntssun9
```

In this example, **infodba** is the administrator's user name and password, and **dba** is the administrator's group.

- To clear all process locks (active and dead) on all nodes, enter the following command:

```
$TC_ROOT/bin/clearlocks -assert_all_dead infodba dba dba
```

In this example, **infodba** is the administrator's user name and password, and **dba** is the administrator's group.

- The following is an example of a line message (report) produced by **clearlocks -verbose**:

```
Processes: 7, Alive: 1, Dead: 6, Remote: 0, Other: 0
```

**CLEARING
PROCESS
LOCKS**

Perform the following steps to clear dead process locks using the **clearlocks** utility.

1. Ensure that all Teamcenter and Teamcenter Integration for NX users are logged out of the system.

When all users are logged out, all valid process locks are cleared.

2. Create a report of all remaining process locks by entering the following command:

```
$TC_ROOT/bin/clearlocks -node_names
```

The system displays a report listing network nodes that still have process locks set against the database. Because all users are logged off, these locks are dead and can be cleared.

3. Run the following command:

```
$TC_ROOT/bin/clearlocks
```

4. Create a report of all remaining process locks by entering the following command:

```
$TC_ROOT/bin/clearlocks -node_names
```

The system displays a report listing network nodes that still have process locks set against the database. Because all users are logged off, these locks are dead and can be cleared.

Any network nodes listed in this second report will require running the **clearlocks** utility with the **-assert_dead** argument to clear the difficult process locks.

5. Run the following command to clear locks held by the session of the specified nodes:

```
$TC_ROOT/bin/clearlocks -assert_dead node-name1 node-name2 node-name3...
```

node-name is a network node listed in the report.

6. Create a report of all remaining process locks by entering the following command:

```
$TC_ROOT/bin/clearlocks -node_names
```

This report should be clean (empty). If there are any nodes listed in this report, contact the Siemens PLM Software Global Technical Access Center (GTAC) for assistance.

clean_backpointer

Removes **relation_type** object references and **ImanRelation** primary and secondary object references from the backpointer table. As of Teamcenter 10.1, these objects are stored in the **ImanRelation** table to improve performance.

Run this utility after upgrading from a version previous to Teamcenter 10.1 if your previous deployment stored **relation_type** object references and **ImanRelation** primary and secondary object backpointer references the backpointer table, rather than the **ImanRelation** table.

SYNTAX

clean_backpointer **-u=admin-user-id** {**-p=password** | **-pf=password-file**}
[-g=dba-group] **[-s=chunk-size | ALL]** **[-m= INFO | DELETE]** **[-h]**

ARGUMENTS**-u**

Specifies the user ID.

This is generally **infodba** or another user with administration privileges. If this argument is used without a value, the operating system user name is used.

Note

If Security Services single sign-on (SSO) is enabled for your server, the user and password arguments are authenticated externally through SSO rather than being authenticated against the Teamcenter database. If you do not supply these arguments, the utility attempts to join an existing SSO session. If no session is found, you are prompted to enter a user ID and password.

-p

Specifies the user's password.

If used without a value, the system assumes a null value. If this argument is not used, the system assumes the *user-ID* value to be the password.

This argument is mutually exclusive with the **-pf** argument.

-pf

Specifies the password file.

For more information about managing password files, see [Manage password files](#).

This argument is mutually exclusive with the **-p** argument. One of the two mutually exclusive password elements is required.

-g

Specifies the administrative group associated with the user.

If used without a value, the user's default group is assumed.

-s

Specifies how many **ImanRelation** primary and secondary reference objects to delete from the backpointer table. Valid values are a positive integer (for example, 100000) or **ALL**.

-m

Specifies whether to provide information about the backpointer table or delete the specified number of objects. Valid values are **INFO** and **DELETE**.

-h

Displays help for this utility.

ENVIRONMENT

As specified in [Manually configuring your environment for Teamcenter utilities](#).

FILES

As specified in [Log files produced by Teamcenter](#).

RESTRICTIONS

None.

EXAMPLES

- To determine how many **ImanRelation** primary and secondary object references exist in your backpointer table, enter the following command on a single line:

```
clean_backpointer -u=infodba -p=password -g=dba -m=INFO
```

The utility returns the total number of backpointer entries, as well as the number of **ImanRelation** objects identified for deletion.

- To determine the amount of time it takes to delete **ImanRelation** primary and secondary object references from your backpointer table (as each deployment is unique in terms of database load, network performance, and server configuration), delete a small amount of objects from the backpointer table and time how long the operation takes to complete.

For example, enter the following command on a single line:

```
clean_backpointer -u=infodba -p=password -g=dba -s=10000 m=DELETE
```

- To delete all **ImanRelation** primary and secondary object references from your backpointer table, enter the following command on a single line:

```
clean_backpointer -u=infodba -p=password -g=dba -s=ALL m=DELETE
```

cleanup_recovery_table

Locates and removes all recipe objects resulting from time-out sessions and client crashes. Typically, recipe objects are deleted when users log off. But during client crashes or session time-outs, recipe objects may remain in the database if the session or objects are not recovered. Run this utility periodically to clear the database of recipe objects.

SYNTAX

cleanup_recovery_table **-u**=*user-id* {**-p**=*password* | **-pf**=*password-file*} [**-g**=*group*] [**-age**=*number-of-days*] [**-report**=*output-file*] [**-verbose**] [**-dryrun**] [**-h**]

ARGUMENTS**-u**

Specifies the user ID.

This is generally **infodba** or another user with administration privileges. If this argument is used without a value, the operating system user name is used.

Note

If Security Services single sign-on (SSO) is enabled for your server, the user and password arguments are authenticated externally through SSO rather than being authenticated against the Teamcenter database. If you do not supply these arguments, the utility attempts to join an existing SSO session. If no session is found, you are prompted to enter a user ID and password.

-p

Specifies the user's password.

If used without a value, the system assumes a null value. If this argument is not used, the system assumes the *user-ID* value to be the password.

This argument is mutually exclusive with the **-pf** argument.

-pf

Specifies the password file. If used without a value, the system assumes a null value. If this argument is not used, the system assumes the *user-ID* value to be the password.

For more information about managing password files, see [Manage password files](#).

This argument is mutually exclusive with the **-p** argument.

-g

Specifies the group associated with the user.

If used without a value, the user's default group is assumed.

-age

Specifies the age (in days) recipe objects must reach to be eligible for deletion. Recipe objects older than the specified age are deleted when the utility is run.

By default, the age is processed as **30** if you do not specify a value for this argument.

-report

Specifies the name of the output report file to which a summary of deleted recipe objects are logged. The summary states how many recipe objects were deleted, from how many dead sessions.

Accepts the relative or full path and file name.

-verbose

Logs a complete list of all deleted recipe objects (as opposed to a summary) to the file specified by the **-report** argument.

If the **-report** argument is not specified, the report is output to the console.

-dryrun

Generates a report stating how many recipe objects exist in the database for how many dead sessions. No recipe objects are deleted.

Siemens PLM Software recommends running this argument first, before actually deleting recipe objects from the database.

If this argument is run with the **-age** argument, the report states how many recipe objects of the specified age exist in the database, relative to the total number of recipe objects.

If the **-report** argument is not specified, the report is output to the console.

-h

Displays help for this utility.

ENVIRONMENT

As specified in [Manually configuring your environment for Teamcenter utilities](#).

FILES

As specified in [Log files produced by Teamcenter](#).

RESTRICTIONS

None.

EXAMPLES

- To delete all recipe objects older than 30 days and write a summarized report of the deleted objects to the **deletedrecipes.txt** file, enter the following command on a single line:

```
cleanup_recovery_table -age=30 -report=deletedrecipes.txt
```
- To delete all recipe objects older than 10 days and write a full report of all deleted objects to the **deletedrecipes.txt** file, enter the following command on a single line:

```
cleanup_recovery_table -age=10 -report=deletedrecipes.txt -verbose
```
- To test the removal of all recipe objects older than 30 days and write a summarized report of the deleted objects to the **deletedrecipes.txt** file, enter the following command on a single line:

```
cleanup_recovery_table -age=30 -report=deletedrecipes.txt -dryrun
```

convert_license_log

Converts a raw license log file into a space-delimited text file. The file can be read in a text reader or Microsoft Excel. The license log file includes timestamp, license daemon, license checkin/checkout, feature key, and user ID. For example:

```
6:02:45 (lmgrd) TIMESTAMP 5/20/2007
6:02:47 (ugslmd) OUT: "tol_cavity_milling" smeyer@svli6020
6:04:44 (ugslmd) IN: "tol_cavity_milling" smeyer@svli6020
6:04:51 (ugslmd) OUT: "tol_cavity_milling" smeyer@svli6020
6:11:09 (ugslmd) IN: "tol_cavity_milling" smeyer@svli6020
6:11:10 (ugslmd) OUT: "tol_cavity_milling" smeyer@svli6020
6:11:20 (ugslmd) IN: "tol_cavity_milling" smeyer@svli6020
12:02:45 (lmgrd) TIMESTAMP 5/20/2007
12:53:51 (ugslmd) OUT: "gateway" jdahlke@AHI6W022
12:54:02 (ugslmd) OUT: "ufunc_execute" jdahlke@AHI6W022
12:54:02 (ugslmd) IN: "ufunc_execute" jdahlke@AHI6W022
12:58:25 (ugslmd) OUT: "cam_base" jdahlke@AHI6W022
12:58:42 (ugslmd) OUT:
```

The Teamcenter license log file is stored in the path specified when you installed the Siemens PLM Software Common Licensing Server to distribute licenses.

For more information about installing the licensing server, see the [Installation on Windows Servers Guide](#) or [Installation on UNIX and Linux Servers Guide](#).

SYNTAX

convert_license_log -input=log-file -output=file-name -delimiter=character [-h]

ARGUMENTS**-u**

Specifies the user ID.

This is generally **infodba** or another user with administration privileges. If this argument is used without a value, the operating system user name is used.

Note

If Security Services single sign-on (SSO) is enabled for your server, the **-u** and **-p** arguments are authenticated externally through SSO rather than being authenticated against the Teamcenter database. If you do not supply these arguments, the utility attempts to join an existing SSO session. If no session is found, you are prompted to enter a user ID and password.

-p

Specifies the password.

If used without a value, the system assumes a null value.

If this argument is not used, the system assumes the *user-ID* value to be the password.

This argument is mutually exclusive with the **-pf** argument.

-pf

Specifies the password file. If used without a value, the system assumes a null value. If this argument is not used, the system assumes the *user-ID* value to be the password.

For more information about managing password files, see [Manage password files](#).

This argument is mutually exclusive with the **-p** argument.

-g

Specifies the group associated with the user.

If used without a value, the user's default group is assumed.

-input

Specifies the full path and file name of the log file you want to convert into a text file.

-output

Specifies the full path and file name of the output file.

-delimiter

Specifies the delimiter character used in the output file. By default, a comma is used.

Surround the delimiter character with single quotes. For example:

' , '

-h

Displays help for this utility.

ENVIRONMENT

As specified in [Manually configuring your environment for Teamcenter utilities](#).

FILES

As specified in [Log files produced by Teamcenter](#).

RESTRICTIONS

None.

generate_client_meta_cache

Generates the shared client metadata cache. This utility is run during the installation and upgrade. Up to three versions of the client meta cache are saved in the server.

Following are some of the situations when you should run this utility:

- Generate the full client cache (**all** argument) if the client cache folder has been deleted.
- Generate the style sheet cache (**stylesheets** argument) after a style sheet is edited. This keeps the style sheet feature in the client cache folder up-to-date.

For more information, see the [Client Customization Programmer's Guide](#).

- Generate the text server cache (**textservers** argument) after text server data has been updated.

For more information about working with text server files, see the [Localization Guide](#).

Metadata is cached on the client computer, eliminating server calls for the cached metadata. When a client logs on to the server, the client cache is placed at the following location on the client machine:

```
home-path\Teamcenter\RAC\session\metacache
```

For example on a Windows machine, the metadata cache is in the following location:

```
C:\Documents and Settings\user-name\Teamcenter\RAC\session\metacache
```

The following features provide additional client caching functionality:

- **Caching refresh notification**

Notifies you when the cache is refreshed. When you log on to the rich client, a check is made for new metadata on the server. If new metadata is found, the cache on the rich client is refreshed, and the following message is displayed:

```
Synchronizing the Rich Client install files with the Teamcenter Server.
```

For more information, see the [Rich Client Interface Guide](#).

- **ClientCache** folder

Stores cached metadata. If you are logged on to the rich client as a database administrator (DBA), you see the folder under the **Home** location.

For more information, see the [Rich Client Interface Guide](#).

Warning If the administrator moves, renames, or deletes this folder, all clients fall back to a nonclient-cached mode.

- **Generate Client Cache** check box

Runs the **generate_client_meta_cache** utility. The check box appears in Teamcenter Environment Manager (TEM) and the Business Modeler IDE.

For more information, see the [Teamcenter Environment Manager Help](#) and the [Business Modeler IDE Guide](#).

- **TC_SKIP_CLIENT_CACHE**

Causes the rich client to run in legacy mode and avoid downloading or using a local client meta cache.

For more information, see the [Preferences and Environment Variables Reference](#).

SYNTAX

```
generate_client_meta_cache [-u=infodba {-p=password |  
-pf=password-file} -g=group]  
command [generate | update | delete | report | tickets | context]  
cache [all | types | lovs | stylesheets | textservers | preferences]  
-log=log-file-path  
-delta=differences-file-path  
l1=language  
l2=language  
-err=error-file-name  
-t=timings  
-remote  
-h
```

ARGUMENTS

-u

Specifies the user ID.

This is generally **infodba** or another user with administration privileges. If this argument is used without a value, the operating system user name is used.

Note

If Security Services single sign-on (SSO) is enabled for your server, the user and password arguments are authenticated externally through SSO rather than being authenticated against the Teamcenter database. If you do not supply these arguments, the utility attempts to join an existing SSO session. If no session is found, you are prompted to enter a user ID and password.

-p

Specifies the user's password.

If used without a value, the system assumes a null value. If this argument is not used, the system assumes the *user-ID* value to be the password.

This argument is mutually exclusive with the **-pf** argument.

-pf

Specifies the password file. If used without a value, the system assumes a null value. If this argument is not used, the system assumes the *user-ID* value to be the password.

For more information about managing password files, see [Manage password files](#).

This argument is mutually exclusive with the **-p** argument.

-g

Specifies the group associated with the user.

If used without a value, the user's default group is assumed.

command

Specifies the command to run. (You must run the command with **-u=infodba**.)

generate

Generates the client cache.

update

Updates the client cache as needed. This is the default setting.

delete

Deletes the client cache.

report

Reports on the state of the client cache.

tickets

Generates tickets for the feature.

context

Checks context for LOVs.

cache

Specifies the client cache to work on.

all

Generates all the available client cache files. This is the default setting.

types

Generates the types cache. This includes the metadata of business objects, property descriptions, and lists of values.

lovs

Generates the list of values (LOVs) cache.

stylesheets

Generates the style sheets cache.

textservers

Generates the text server cache.

preferences

Generates the site preferences cache.

-log

Specifies file path and name of the log file that contains the results of this execution. The default log file is written to the **TEMP** directory.

This argument is optional.

-delta

Specifies the file path and name of the file into which data model differences are written.

-l1

Specifies the first language for which to generate the client cache, for example, **en_US**, **de_DE**, **es_ES**, **fr_FR**, **pt_BR**.

-l2

Specifies the second language for which to generate the client cache.

-err

Specifies the error file name. The default error file is written to the **TEMP** directory.

-t

Specifies time function calls.

-remote

Specifies remote start. This generates log files.

-h

Displays help for this utility.

ENVIRONMENT

As specified in [Manually configuring your environment for Teamcenter utilities](#).

FILES

As specified in [Log files produced by Teamcenter](#).

RESTRICTIONS

None.

EXAMPLES

- To generate or regenerate all the client meta cache, enter the following command on a single line.

```
generate_client_meta_cache generate all -u=infodba -p=password -g=dba
```

This command keeps the previous version of cache data in the server. Up to three versions for all the client meta cache are retained on the server.

- To generate style sheet client meta cache, enter the following command on a single line.

```
generate_client_meta_cache generate stylesheet -u=infodba  
-p=password -g=dba
```

- To generate text server files client meta cache, enter the following command on a single line.

```
generate_client_meta_cache generate textservers -u=infodba  
-p=password -g=dba
```

generate_metadata_cache

Generates the metadata shared server cache dataset. The cache is generated only if constant, type, or property metadata is updated since the last update. The **-force** argument forces regeneration of the cache.

If you receive the following message, an administrator must run this utility on the server:

```
The schema file is out of date. Please regenerate.
```

This indicates that when a template containing schema changes was installed to the server, the **Generate Server Cache** check box was not selected in Teamcenter Environment Manager (TEM) or the Business Modeler IDE. (It is also possible that the error message appears because the server instance the user is attempting to connect to does not have an up-to-date server cache. The user should log off and log on several times to connect to a fresh server instance. If the error message persists, the user should contact the server administrator to regenerate the server cache.)

For more information about managing shared server cache, see the [System Administration Guide](#).

SYNTAX

```
generate_metadata_cache -u=user-id {-p=password |  
-pf=password-file} [-g=group]  
[-force]  
[-h]
```

ARGUMENTS**-u**

Specifies the user ID.

This is generally **infodba** or another user with administration privileges. If this argument is used without a value, the operating system user name is used.

Note

If Security Services single sign-on (SSO) is enabled for your server, the user and password arguments are authenticated externally through SSO rather than being authenticated against the Teamcenter database. If you do not supply these arguments, the utility attempts to join an existing SSO session. If no session is found, you are prompted to enter a user ID and password.

-p

Specifies the user's password.

If used without a value, the system assumes a null value. If this argument is not used, the system assumes the *user-ID* value to be the password.

This argument is mutually exclusive with the **-pf** argument.

-pf

Specifies the password file. If used without a value, the system assumes a null value. If this argument is not used, the system assumes the *user-ID* value to be the password.

For more information about managing password files, see [Manage password files](#).

This argument is mutually exclusive with the **-p** argument.

-g

Specifies the group associated with the user.

If used without a value, the user's default group is assumed.

-force

Forces creation of the dataset even if a correct version already exists.

-h

Displays help for this utility.

ENVIRONMENT

As specified in [Manually configuring your environment for Teamcenter utilities](#).

FILES

As specified in [Log files produced by Teamcenter](#).

RESTRICTIONS

None.

install_event_types

Defines which event and object types can be subscribed to and/or audited. This utility also add event types to an object type.

Caution The **install_event_types** is deprecated.

Beginning in Teamcenter 10, the **create** parameter used with the **-f** argument is obsolete, and therefore you can no longer create event types using this utility. Now you must create custom event types using the Business Modeler IDE rather than this utility.

For more information, see the [Business Modeler IDE Guide](#).

SYNTAX

```
install_event_types [-u=user-id [-p=password | -pf=password-file] -g=group]
-f=function {install | add | remove | modify | listValidEvents
| listEventtypes | text-file-name} [-overwrite]
[-eventtype=eventTypeId] [-imantype=imanTypeName]
[-imanclass=imanClassName] [-remove] [-audit] [-noaudit]
[-subscribe] [-nosubscribe] [-h]
```

ARGUMENTS

-u

Specifies the user ID.

This is generally **infodba** or another user with administration privileges. If this argument is used without a value, the operating system user name is used.

Note If Security Services single sign-on (SSO) is enabled for your server, the **-u** and **-p** arguments are authenticated externally through SSO rather than being authenticated against the Teamcenter database. If you do not supply these arguments, the utility attempts to join an existing SSO session. If no session is found, you are prompted to enter a user ID and password.

-p

Specifies the password.

If used without a value, the system assumes a null value.

If this argument is not used, the system assumes the *user-ID* value to be the password.

This argument is mutually exclusive with the **-pf** argument.

-pf

Specifies the password file. If used without a value, the system assumes a null value. If this argument is not used, the system assumes the *user-ID* value to be the password.

For more information about managing password files, see [Manage password files](#).

This argument is mutually exclusive with the **-p** argument.

-g

Specifies the group associated with the user.

If used without a value, the user's default group is assumed.

-f

Specifies the mode in which the utility executes. The mode must be one of the following:

- **install**
Installs standard event types and event type mappings.
- **add**
Adds the event type to the specified imantype and imanclass.
- **remove**
Removes event types or event type mappings.
- **modify**
Modifies event type mapping.
- **listValidEvents**
Lists valid event types for the specified imanclass and imantype.
- **listEventtypes**
Lists all the event types in the database.
- *text-file-name*
Specifies the file name used to create event type mappings, modify event type mappings, or delete event types and event type mappings.

-overwrite

Overwrites existing definitions, if any, during installation. Valid only with the **-f=install** and **-f=input-file** arguments.

-eventtype

Specifies the event type ID when creating or deleting event types or event type mappings.

-imantype

Specifies the **imantype** name when creating or deleting event types or event type mappings or when listing valid event types for that particular **imantype**.

-imanclass

Specifies the **imanclass** name when creating event type mappings, deleting event type mappings, or listing valid event types for that particular **imanclass**.

-remove

Use with the **-f=file-name** option to perform the remove operation (remove event types and event type mapping) on the file data.

-audit

Specifies that the event type can be audited. Valid only with the **-f=modify** argument.

-noaudit

Specifies that the event type cannot be audited. Valid only with the **-f=modify** argument.

-subscribe

Specifies that the event type can be subscribed to. Valid only with the **-f=modify** argument.

-nosubscribe

Specifies that the event type cannot be subscribed to. Valid only with the **-f=modify** argument.

-h

Displays help for this utility.

ENVIRONMENT

As specified in [Manually configuring your environment for Teamcenter utilities](#).

FILES

As specified in [Log files produced by Teamcenter](#).

RESTRICTIONS

When deleting event types and event type mappings using a file, all event type mappings for the event type must be deleted prior to deleting event type itself. For more information, see the example for using the **-f=file-name -remove** option, below.

EXAMPLES

- To install the default event types and event type mapping definitions, enter the following command on a single line:

```
install_event_types -f=install -overwrite
```

The **-override** switch causes the install to overwrite any existing definitions.

- To add the **MyEventType** event type as a valid event type for the Teamcenter type **Item**, enter the following command on a single line:

```
install_event_types -f=add -imantype=Item -imaclass=Item
-eventtype=MyEventType
```

- To remove the **MyEventType** event type from the Teamcenter type **Item**, enter the following command on a single line:

```
install_event_types -f=remove -imantype=Item -imaclass=Item
-eventtype=MyEventType
```

Note

Users cannot remove Teamcenter internal event types.

- To set the **MyEventType** event type as subscribable but not auditable for the Teamcenter type **Item**, enter the following command on a single line:

```
install_event_types -f=modify -imantype=Item
-imaclass=Item -eventtype=MyEventType -audit -nosubscribe
```

- To list all valid event types for the Teamcenter type **Item**, enter the following command on a single line:

```
install_event_types -f=listValidEvents -imantype=Item -imaclass=Item
```

- To list all event types defined in the database, enter the following command on a single line:

```
install_event_types -f=listEventtypes
```

- To read the specified file, define new event types to install, (each line with an event type name) and defines event type mappings (each line with an event type mapping) in the following format:

```
iman_type_name, iman_class_name, event_type_name, subscribable_flag,
auditable_flag
install_event_types -f=C:\temp\my_event_types.txt -overwrite
```

In the **C:\temp\my_event_types.txt**:

```
my_event_type_1
my_event_type_2
EngChange,Item,my_event_type_1,true,true
EngChange,Item,my_event_type_2,true,false
EngChange Revision,ItemRevision,my_event_type_1,true,true
EngChange Revision,ItemRevision,my_event_type_2,true,false
```

- To read the specified file, delete event type mappings, (each line with an event type mapping) and delete event types (each line with an event type name) in the formats shown, enter the following command on a single line:

```
install_event_types -f=C:\temp\my_event_types.txt -remove
```

Format to delete event type mapping

```
iman_type_name, iman_class_name, event_type_name
```

Format to delete event type

```
event_type_name
```

The **my_event_types.txt** file contains the following:

```
EngChange,Item,my_event_type_1
EngChange,Item,my_event_type_2
EngChange Revision,ItemRevision,my_event_type_1
EngChange Revision,ItemRevision,my_event_type_2
my_event_type_1
my_event_type_2
```

- To delete the **MyEventType** event type, enter the following command on a single line:

```
install_event_types -f=remove -eventtype=MyEventType
```

To set display names for event types:

- Open the **user_property_names.xml** file. This file is located in the **TC_ROOT\lang\textserver\en_US** directory.
- Add the keys for the event names as follows:

```
<key id="k_et_event1">Event Name 1</key>
<key id="k_et_event2">Event Name 2</key>
```

- Stop the pool manager.
- Delete the Teamcenter shared memory files.

The shared memory files are typically located in the temporary files location. For example, in Windows, the shared files are located in the **C:\Temp\P_folder number** directory.

- Restart the pool manager.

list_types

Lists all types in the Teamcenter database. Use the output of this utility as input to the [database_verify](#) utility for the offline case.

SYNTAX

list_types [-u=*user-id* {-p=*password* | -pf=*password-file*} -g=*group*]
[-outfile=*output-file-name*] [-h]

ARGUMENTS

-u

Specifies the user ID.

This is generally **infodba** or another user with administration privileges. If this argument is used without a value, the operating system user name is used.

Note If Security Services single sign-on (SSO) is enabled for your server, the **-u** and **-p** arguments are authenticated externally through SSO rather than being authenticated against the Teamcenter database. If you do not supply these arguments, the utility attempts to join an existing SSO session. If no session is found, you are prompted to enter a user ID and password.

-p

Specifies the password.

If used without a value, the system assumes a null value.

If this argument is not used, the system assumes the *user-ID* value to be the password.

This argument is mutually exclusive with the **-pf** argument.

-pf

Specifies the password file. If used without a value, the system assumes a null value. If this argument is not used, the system assumes the *user-ID* value to be the password.

For more information about managing password files, see [Manage password files](#).

This argument is mutually exclusive with the **-p** argument.

-g

Specifies the group associated with the user.

If used without a value, the user's default group is assumed.

-outfile

Specifies the name of the file to contain the output. If this argument is not specified, all types information is displayed on the console.

-h

Displays help for this utility.

ENVIRONMENT

As specified in [Manually configuring your environment for Teamcenter utilities](#).

FILES

As specified in [Log files produced by Teamcenter](#).

RESTRICTIONS None.

EXAMPLES None.

list_users

Creates a list of users currently logged on to Teamcenter and the node they are using. This information is useful if database maintenance is necessary and all users currently logged on must be notified.

SYNTAX

list_users [-u=*user-id* {-p=*password* | -pf=*password-file*} -g=*group*] [-h]

ARGUMENTS**-u**

Specifies the user ID.

This is generally **infodba** or another user with administration privileges. If this argument is used without a value, the operating system user name is used.

Note

If Security Services single sign-on (SSO) is enabled for your server, the **-u** and **-p** arguments are authenticated externally through SSO rather than being authenticated against the Teamcenter database. If you do not supply these arguments, the utility attempts to join an existing SSO session. If no session is found, you are prompted to enter a user ID and password.

-p

Specifies the password.

If used without a value, the system assumes a null value.

If this argument is not used, the system assumes the *user-ID* value to be the password.

This argument is mutually exclusive with the **-pf** argument.

-pf

Specifies the password file. If used without a value, the system assumes a null value. If this argument is not used, the system assumes the *user-ID* value to be the password.

For more information about managing password files, see [Manage password files](#).

This argument is mutually exclusive with the **-p** argument.

-g

Specifies the group associated with the user.

If used without a value, the user's default group is assumed.

-h

Displays help for this utility.

ENVIRONMENT

As specified in [Manually configuring your environment for Teamcenter utilities](#).

FILES

As specified in [Log files produced by Teamcenter](#).

RESTRICTIONS

None.

EXAMPLES

None.

migrate_home_folder

Finds the instances of the home folder for all users in the Teamcenter database and modifies the **object_type** property value to **Fnd0HomeFolder**. The **Fnd0HomeFolder** business object represents the home folder and is a child of the **Folder** business object.

This utility is run automatically during upgrade. You only need to run it manually if the utility did not complete during upgrade.

By default, this utility finds and process only the unmigrated instances. You can force it to process the objects that are already migrated by specifying the **-force_rerun** argument. You can rerun the utility any number of times. You can process all the home folder instances in bulk using the **-objects** argument.

SYNTAX

```
migrate_home_folder -u=user-ID {-p=password | -pf=password-file} [-g=group]
[-objects=number-of-objects-per-iteration]
[-force-rerun]
[-log=log-file-path]
[-v]
[-h]
```

ARGUMENTS

-u

Specifies the user ID. The user must have administrative privileges.

If this argument is used without a value, the operating system user name is used.

-p

Specifies the user's password.

If used without a value, the system assumes a null value. If this argument is not used, the system assumes the *user-ID* value to be the password.

This argument is mutually exclusive with the **-pf** argument.

-pf

Specifies the password file. If used without a value, the system assumes a null value. If this argument is not used, the system assumes the *user-ID* value to be the password.

For more information about managing password files, see [Manage password files](#).

This argument is mutually exclusive with the **-p** argument.

-g

Specifies the group associated with the user.

-objects

Specifies the number objects to be processed per iteration. The default value is 1000 objects. The input value must be a positive integer. If the number of objects specified exceeds the maximum objects in the database, this utility processes all instances in the first iteration itself.

-force_rerun

Forces the utility to query and migrate the instances that are already migrated along with unmigrated instances.

-log

Specifies the path for the log file. The log file contains the information about the number objects successfully migrated, the number of objects not migrated, the time taken to complete the operation, and so on. If this argument is not specified, the log file is generated under the directory defined by the **TC_TMP_DIR** environment variable.

-h

Displays help for this utility.

ENVIRONMENT

As specified in [Manually configuring your environment for Teamcenter utilities](#).

FILES

Apart from the default log files specified in [Log files produced by Teamcenter](#), this utility generates an additional log file with the details of the number of instances migrated, the number of instances that failed to migrate, and the time taken to complete the operation. The additional log file has this format:

migrate_home_folderYYYY_MM_DD_HH_MM_SS.log

You can find it in the directory specified by the **TC_TMP_DIR** environment variable.

RESTRICTIONS

You must be a Teamcenter administrator to execute this utility.

EXAMPLES

- To migrate all home folder instances in the database with up to 5000 objects per iteration, enter the following command.

```
migrate_home_folder -u=infodba -p=password -g=dba -objects=5000
```

- To migrate all home folder instances in the database with up to 10000 objects per iteration and to force the utility to rerun the migration on already migrated instances along with unmigrated instances:

```
migrate_home_folder -u=infodba -p=password -g=dba -objects=5000 -force_rerun
```

purge_file_cache

Cleans up the file cache based on last access date of the files in the cache. It also deletes any file from the cache that no longer exists within the Teamcenter volume.

SYNTAX

purge_file_cache [-u=*user-id* {-p=*password* | -pf=*password-file*} -g=*group*] [-max=*megabytes-to-remain-in-cache*] [-h]

ARGUMENTS

-u

Specifies the user ID.

This is generally **infodba** or another user with administration privileges. If this argument is used without a value, the operating system user name is used.

Note

If Security Services single sign-on (SSO) is enabled for your server, the **-u** and **-p** arguments are authenticated externally through SSO rather than being authenticated against the Teamcenter database. If you do not supply these arguments, the utility attempts to join an existing SSO session. If no session is found, you are prompted to enter a user ID and password.

-p

Specifies the password.

If used without a value, the system assumes a null value.

If this argument is not used, the system assumes the *user-ID* value to be the password.

This argument is mutually exclusive with the **-pf** argument.

-pf

Specifies the password file. If used without a value, the system assumes a null value. If this argument is not used, the system assumes the *user-ID* value to be the password.

For more information about managing password files, see [Manage password files](#).

This argument is mutually exclusive with the **-p** argument.

-g

Specifies the group associated with the user.

If used without a value, the user's default group is assumed.

-max

Specifies the maximum size, in megabytes, of the file cache after the purge is performed. If the parameter is not used, only files that are no longer available in the Teamcenter volume are purged.

-h

Displays help for this utility.

ENVIRONMENT

As specified in [Manually configuring your environment for Teamcenter utilities](#).

FILES

As specified in *Log files produced by Teamcenter*.

RESTRICTIONS

The **purge_file_cache** utility must be run by a user with the privileges required to delete files from the cache directory at the operating system level.

EXAMPLES

- The **purge_file_cache** utility can be used to clean up the file cache so that at most only 10 megabytes of data remain after purging, as follows:

```
purge_file_cache -u=smith -p=password -g=design -max=10
```

- In this example, the files that are no longer available in the Teamcenter volume are purged from the cache.

```
purge_file_cache -u=smith -p=password -g=design
```

reset_user_home_folder

Repairs corruption that may occur when deleting a user from the database by redirecting the user's home folder, mailbox folder, and **Newstuff** folder to the home folder of the administrator running this utility.

If a home folder owned by the administrator is found, the utility points the deleted user's home folder back the administrator's home folder. If the administrator's home folder is not found, the utility creates a new home folder and redirects the deleted user's home folder to it.

SYNTAX

```
%TC_BIN%\reset_user_home_folder [-u=user-id {-p=password |  
-pf=password-file}  
-g=group] -id=user-id-whose-home-folder-is-to-be-reset [-h]
```

ARGUMENTS

-u

Specifies the user ID.

This is generally **infodba** or another user with administration privileges. If this argument is used without a value, the operating system user name is used.

Note

If Security Services single sign-on (SSO) is enabled for your server, the **-u** and **-p** arguments are authenticated externally through SSO rather than being authenticated against the Teamcenter database. If you do not supply these arguments, the utility attempts to join an existing SSO session. If no session is found, you are prompted to enter a user ID and password.

-p

Specifies the password.

If used without a value, the system assumes a null value.

If this argument is not used, the system assumes the *user-ID* value to be the password.

This argument is mutually exclusive with the **-pf** argument.

-pf

Specifies the password file. If used without a value, the system assumes a null value. If this argument is not used, the system assumes the *user-ID* value to be the password.

For more information about managing password files, see [Manage password files](#).

This argument is mutually exclusive with the **-p** argument.

-g

Specifies the group associated with the user.

If used without a value, the user's default group is assumed.

-id

Specifies the user ID of the user who owns the home folder to be reset.

ENVIRONMENT

As specified in [Manually configuring your environment for Teamcenter utilities](#).

FILES

As specified in *Log files produced by Teamcenter*.

RESTRICTIONS

None.

EXAMPLES

None.

site_util

Performs site-related maintenance, such as creating and deleting remote sites, setting ownership, and changing the ID of remote sites.

SYNTAX

```
site_util [-u=user-id {-p=password | -pf=password-file} -g=group]
-f=function [-site_id=site-id] [-site_name=site-name]
[-node_name=node-name] [-gms_url=gms-url]
[-ods= y | n] [-hub=y | n] [-http=y | n] [-tcxml=y | n] [-offline=y | n]
[-replicaDel=y | n] [-display_only] [-h]
```

ARGUMENTS

-u

Specifies the user ID.

This is generally **infodba** or another user with administration privileges. If this argument is used without a value, the operating system user name is used.

Note

If Security Services single sign-on (SSO) is enabled for your server, the **-u** and **-p** arguments are authenticated externally through SSO rather than being authenticated against the Teamcenter database. If you do not supply these arguments, the utility attempts to join an existing SSO session. If no session is found, you are prompted to enter a user ID and password.

-p

Specifies the password.

If used without a value, the system assumes a null value.

If this argument is not used, the system assumes the *user-ID* value to be the password.

This argument is mutually exclusive with the **-pf** argument.

-pf

Specifies the password file. If used without a value, the system assumes a null value. If this argument is not used, the system assumes the *user-ID* value to be the password.

For more information about managing password files, see [Manage password files](#).

This argument is mutually exclusive with the **-p** argument.

-g

Specifies the group associated with the user.

If used without a value, the user's default group is assumed.

-f

Specifies one of the following functions:

create

Defines a remote site in the local database.

set_id

Changes the current ID of a remote site. This function corrects errors made while defining sites and must be used with extreme caution.

Caution Modifying a site ID can result in serious data sharing problems. This function is not allowed to be run on the local site, because it will corrupt the database.

The **-site_name** argument is required when using the **set_id** function.

modify

Changes attributes of a site other than the site ID and may be used for local or remote sites. The **-site_id** argument is required to identify the site being modified. Only the given attributes are modified.

delete

Deletes a remote site from the local database. Only those sites not referenced by any object in the local database can be deleted. This function cannot be used for the local site. The **-site_id** argument is required when using the **delete** function.

list

Lists all defined sites in the database and their attributes. No attribute switch is required.

fix_site_ownership

Sets the ownership of all defined sites in the database as being owned by the local site. Use this when you encounter an error stating that you cannot export a **POM_imc** object because it is owned by another site. This can be run while users are logged on to the database. No attribute switch is required.

-site_id

Specifies the ID of the site to which the specified function applies.

-site_name

Specifies the name of the site when creating or modifying sites.

-node_name

Specifies the node name when creating a new site. This argument can also be used when modifying site properties. If the value of the **-http** argument is set to **y**, the value of **-site_name** can be a URL of an SOA.

-gms_url

Specifies the URL for Global Services.

-ods

Specifies whether the site being created or modified is an Object Directory Services (ODS) site. For more information about Object Directory Services, see *Multi-Site Collaboration Guide*.

-hub

Specifies whether the site being created or modified is a hub. For more information about hub configurations, see *Multi-Site Collaboration Guide*.

-http

Specifies whether to use the HTTP protocol or remote procedure call (RPC) protocol.

-tcxml

Specifies the utility uses TC XML payload instead of an object manager.

-offline

Specifies that the site has no network connection to the local site. This argument is intended to avoid sending unnecessary messages to a site when a replica is deleted.

-replicaDel

Switch that indicates objects replicated to a given site can be deleted as long as the object is not replicated to other sites that do not have this property.

-display_only

Determines whether it is necessary to fix site ownership. This argument must be used in conjunction with the **fix_site_ownership** function. A returned count greater than zero indicates that site ownership must be fixed by running the utility using the **fix_site_ownership** function.

-h

Displays help for this utility.

ENVIRONMENT

As specified in [Manually configuring your environment for Teamcenter utilities](#).

FILES

As specified in [Log files produced by Teamcenter](#).

RESTRICTIONS

None.

EXAMPLES

- To define a new remote site in the local database, enter the following command on a single line:

```
site_util -f=create -site_id=123456789 -site_name=Site1
-node_name=node1 -ods=y
```

- To change the node name of a site, enter the following command on a single line:

```
site_util -f=modify -site_id=123456789 -node_name=node2
```

- To indicate that a site is a multisite hub, enter the following command on a single line:

```
site_util -f=modify -site_id=123456789 -hub=y
```

A returned count greater than zero indicates that site ownership must be fixed by running the utility using the **fix_site_ownership** function.

- To determine if it is necessary to run the **fix_site_ownership** function, enter the following command on a single line:

```
site_util -f=fix_site_ownership -display_only
```

- To define a GMS site, enter the following command on a single line:

```
site_util -f=create -http=y -tcxml=y
-node_name=http://url_of_the_gs_instance
```

runBatch

Script that allows a site to execute a group of translations together. The translations are contained in either the default **testxml\TranslationPerf.xml** file or any other XML file and the path of the XML file must be passed as argument.

SYNTAX

```
-u=user-id {-p=password | -pf=password-file} [-g=group]  
TranslationManagementRootDir/AdminClient/bin/runBatch  
[-help] input-batch-xml-file-path
```

ARGUMENTS

-u

Specifies the user ID.

This is generally **infodba** or another user with administration privileges. If this argument is used without a value, the operating system user name is used.

Note

If Security Services single sign-on (SSO) is enabled for your server, the **-u** and **-p** arguments are authenticated externally through SSO rather than being authenticated against the Teamcenter database. If you do not supply these arguments, the utility attempts to join an existing SSO session. If no session is found, you are prompted to enter a user ID and password.

-p

Specifies the password.

If used without a value, the system assumes a null value.

If this argument is not used, the system assumes the *user-ID* value to be the password.

This argument is mutually exclusive with the **-pf** argument.

-pf

Specifies the password file. If used without a value, the system assumes a null value. If this argument is not used, the system assumes the *user-ID* value to be the password.

For more information about managing password files, see [Manage password files](#).

This argument is mutually exclusive with the **-p** argument.

-g

Specifies the group associated with the user.

If used without a value, the user's default group is assumed.

input-batch-xml-file-path

Input batch XML file path. This argument is optional. If the path is not specified, the system uses the default path as **..\testxml\TranslationPerf.xml**.

-help

Displays help for this utility.

ENVIRONMENT

None.

FILES

None.

RESTRICTIONS

None.

EXAMPLES

- The following is an example of a translation XML file:

```
<?xml version="1.0"?>

<!--
This software and related documentation are proprietary to UGS Corp.
COPYRIGHT 2007 UGS CORP. ALL RIGHTS RESERVED
-->

<TranslationTasks>
  <!-- RootDir is the common directory for all the Input files in this xml file.    >
  <    This can be absolute path or relative path from AdminClient/bin Directory. -->
  <RootDir value="../data"/>
  <!-- To add more tasks to the batch, copy and insert more TranslationTask elements -->
  <TranslationTask Submits="1" Provider="UGS" Service="tozipfile" context="Translation">
    <Priority value="2"/>
    <!-- For time based tasks uncomment Time tag. Time format is "MM/dd/yyyy HH:mm" -->
    <!--Time value="01/25/2007 17:33"/-->
    <Options>
      <Option key="Test_Key" value="Test_Value"/>
    </Options>
    <!-- If RootDir is specified input file path is value of "RootDir + Input". >
    <    If RootDir is not specified input file path is value of "Input" and    >
    <      has to be absolute path.                                           >
    <    This applies to the both Input and Dependant element attribute.      -->
    <Input value="ssw_idi0001.idi"/>
    <!-- Dependant values take * wildcard -->
    <!-- Dependant value="*.dep"/ -->
  </TranslationTask>
</TranslationTasks>
```

- The following are examples of command line entries:

```
runbatch test test
runbatch test test ..\testxml\TranslationPerf.xml
```

tc_mail_smtp

Sends SMTP (Simple Mail Transfer Protocol) e-mail. Use this platform-independent utility when sending e-mail on both UNIX and Windows platforms.

SYNTAX

```
tc_mail_smtp -u=user-id {-p=password | -pf=password-file} [-g=group]  
{-to=address | -to_list_file=email-list-file-name} [-cc=address] [-bcc=address]  
[-subject=subject-line] [-server=mail-server-name [:port]] [-port=]  
[-body=file-name [-body=alternate-file-name]] [-attachments=file-name]  
[-user=sender's-name] [-validation=validation-mode] [-testmode=file-name] [-h=]
```

ARGUMENTS**-u**

Specifies the user ID.

This is generally **infodba** or another user with administration privileges. If this argument is used without a value, the operating system user name is used.

Note

If Security Services single sign-on (SSO) is enabled for your server, the **-u** and **-p** arguments are authenticated externally through SSO rather than being authenticated against the Teamcenter database. If you do not supply these arguments, the utility attempts to join an existing SSO session. If no session is found, you are prompted to enter a user ID and password.

-p

Specifies the password.

If used without a value, the system assumes a null value.

If this argument is not used, the system assumes the *user-ID* value to be the password.

This argument is mutually exclusive with the **-pf** argument.

-pf

Specifies the password file. If used without a value, the system assumes a null value. If this argument is not used, the system assumes the *user-ID* value to be the password.

For more information about managing password files, see [Manage password files](#).

This argument is mutually exclusive with the **-p** argument.

-g

Specifies the group associated with the user.

If used without a value, the user's default group is assumed.

-to

Specifies the e-mail address of the recipient. If there are multiple recipients, use a **-to** argument for each recipient.

Either this argument or the **-to_list_file** argument is required.

-to_list_file

Specifies the full path and file name of the file containing the list of e-mail addresses. Each line of the file must contain a full e-mail address.

Either this argument or the **-to** argument is required.

-cc

Specifies the e-mail address of the carbon-copied recipient. If there are multiple recipients, use a **-cc** argument for each recipient.

-bcc

Specifies the e-mail address of the blind carbon-copied recipient. If there are multiple recipients, use a **-bcc** argument for each recipient.

-subject

Specifies the subject line of the e-mail. Accepts a string up to 100 characters. Enclose the string text in double quotes.

-server

Specifies the name of the mail server. Accepts a string up to 100 characters. Optionally, you can define the mail server port within this argument. For example:

```
-server=myserver:1234
```

Alternatively, you can define the mail server port using the **-port** argument. If you do not use either method to specify the mail server port, the SMTP mail port (25) of the local machine is used.

-port

Specifies the mail server port. If not defined, the SMTP mail port (25) of the local machine is used.

-body

Specifies the full path (limited to 200 characters) to the file containing the text of the e-mail body. Limited to 200 characters. You can also specify an alternate file name, in a different format. Files can be either text files or HTML files. (HTML files extensions can be only **.htm** or **.HTML**.) For example:

```
-body=C:\correspondence\body\filename1.txt  
-body=C:\correspondence\body\filename2.htm
```

You can use this argument to specify a total of one text file, or one HTML file, or one text file and one HTML file. You cannot specify, two or more text files, or two or more HTML files.

If you specify one text file and one HTML file, the e-mail always contains the text message first, and then the HTML message, regardless of the order in which you sent the argument values.

If you specify multiple values for the same body type, the last value entered in the command line for this argument is read by the system. For example, if you specify:

```
-body=C:\correspondence\body\filename1.txt  
-body=C:\correspondence\body\filename2.txt  
-body=C:\correspondence\body\filename3.txt
```

Only the contents of **filename3.txt** are added to the body of the message.

-attachments

Specifies the full path and file name of one or more attachment lists and the format of each attachment (**B**=binary text, **T**=text). The full path and file name cannot exceed 200 characters.

For example, you can specify the full path of the location of the **bin_file** file and indicate the file is a binary file:

```
C:\correspondence\attachments\attachment1\bin_file=B
```

To include graphics within the HTML body of an e-mail, specify the image source within the HTML (**img src=cid:myimage.jpg**), and then specify the same image source within the attachment file. For example:

```
C:\correspondence\attachments\attachment1\myimage.jpg=B
```

To include several graphic attachments with the same name, differentiate the files with IDs. In the HTML body of an e-mail, specify the various image sources within the HTML (**img src=cid:fred_image.jpg**) and (**img src=cid:bob_image.jpg**). Then specify the corresponding image sources within the attachment file. For example:

```
C:\correspondence\attachments\attachment1\myimage.jpg=B,html,id=fred_image.jpg
C:\correspondence\attachments\attachment2\myimage.jpg=B,html,id=bob_image.jpg
```

-user

Specifies the name of the user from whom the e-mail is sent. If not defined, the login name of the local machine is used.

-validation

Determines the behavior of e-mail delivery if an invalid e-mail address is entered.

0

E-mail delivery continues, even if incorrect e-mail addresses are encountered. No error is returned. This is the default setting.

1

E-mail delivery continues, even if incorrect e-mail addresses are encountered. An error is returned.

2

E-mail delivery is sent first to recipients on the **To** list, then the **CC** list, and then the **BCC** list. Delivery stops at the first invalid address. An error is returned.

-testmode

Allows you to print a test of the e-mail output. Specify either **stderr**, **stdout**, or the full path and file name to the **stderr** file.

ENVIRONMENT

As specified in *[Manually configuring your environment for Teamcenter utilities](#)*.

FILES

As specified in *[Log files produced by Teamcenter](#)*.

RESTRICTIONS

None.

EXAMPLES

- When defining the formatting of the attachment list, each line in the list must consist of an attachment file name and its file format.

Assume that the attachment list file is called **attachment_list.txt**, and the list's content is:

```
C:\temp\BAR_RED.JPG=B  
C:\temp\test.txt=T
```

This results in two files being included as attachments, one binary file named **BAR_RED.JPG** and one text file named **test.txt**. Both files are located in the **C:\temp** directory.

```
tc_mail_smtp -subject="my test"  
-to=person1@company1.com -to=person2@company2.com  
-cc=person3@company3.com  
-bcc=person4@company4.com  
-server=mail_server_1  
-body=C:\temp\test.txt -attachments=C:\temp\attachment_list.txt  
-user=person5
```

uih_to_xml

Converts existing UIH files to XML files that serve text and error messages.

SYNTAX

uih_to_xml **-u**=*user-id* { **-p**=*password* | **-pf**=*password-file* } [**-g**=*group*]

ARGUMENTS**-u**

Specifies the user ID.

This is generally **infodba** or another user with administration privileges. If this argument is used without a value, the operating system user name is used.

Note

If Security Services single sign-on (SSO) is enabled for your server, the **-u** and **-p** arguments are authenticated externally through SSO rather than being authenticated against the Teamcenter database. If you do not supply these arguments, the utility attempts to join an existing SSO session. If no session is found, you are prompted to enter a user ID and password.

-p

Specifies the password.

If used without a value, the system assumes a null value.

If this argument is not used, the system assumes the *user-ID* value to be the password.

-pf

Specifies the password file. If used without a value, the system assumes a null value. If this argument is not used, the system assumes the *user-ID* value to be the password.

For more information about managing password files, see [Manage password files](#).

This argument is mutually exclusive with the **-p** argument.

-g

Specifies the group associated with the user.

If used without a value, the user's default group is assumed.

file1 **.uih** *file2* **.uih...**

Lists UIH files to be converted to XML. If no file is specified, the current directories and subdirectories are searched for UIH files, which are then translated to XML.

-d= *my / directory*

Parses XML files in a given directory and related subdirectories.

-h

Displays help for this utility.

ENVIRONMENT

As specified in [Manually configuring your environment for Teamcenter utilities](#).

FILES

As specified in [Log files produced by Teamcenter](#).

RESTRICTIONS

None.

EXAMPLES

There are two basic examples for this utility.

- The following example traverses the current directory and all subdirectories searching for UIH files and translates them to XML files:

```
uih_to_xml
```

- The following example translates only the specified UIH files into their corresponding XML files:

```
uih_to_xml    ss_errors.uih    ae_errors.uih
```

Document management

pdfgenerator

Translates Microsoft Office (Word, Excel, PowerPoint, and Project) documents, Postscript, Encapsulated Postscript (**.eps**), Adobe Photoshop (**.psd**), WordPerfect, rich text, bitmap, GIF, JPEG, TIFF and multipage TIF to Adobe PDF file format.

You need the source authoring applications such as the MS Office documents (Word, Excel, PowerPoint or/and Project), Adobe Acrobat, and Adobe Live Cycle PDF Generator ES to run this program.

SYNTAX

```
-u=user-id {-p=password | -pf=password-file} [-g=group]  
DispatcherRootDirectory/Module/translators/docmgt_translators/  
pdfgenerator.bat -inputDir=source directory path -inputFile=full  
path of the source file to be translated-outputDir=output directory  
path-outputType=output format
```

ARGUMENTS**-u**

Specifies the user ID.

This is generally **infodba** or another user with administration privileges. If this argument is used without a value, the operating system user name is used.

Note

If Security Services single sign-on (SSO) is enabled for your server, the **-u** and **-p** arguments are authenticated externally through SSO rather than being authenticated against the Teamcenter database. If you do not supply these arguments, the utility attempts to join an existing SSO session. If no session is found, you are prompted to enter a user ID and password.

-p

Specifies the password.

If used without a value, the system assumes a null value.

If this argument is not used, the system assumes the *user-ID* value to be the password.

-pf

Specifies the password file. If used without a value, the system assumes a null value. If this argument is not used, the system assumes the *user-ID* value to be the password.

For more information about managing password files, see [Manage password files](#).

This argument is mutually exclusive with the **-p** argument.

-g

Specifies the group associated with the user.

If used without a value, the user's default group is assumed.

-inputDir

Specifies the complete path of the input directory.

-inputFile

Specifies the path of the input file and the input file name. This is the source file to be translated.

-outputDir

Specifies the path of the output file.

-outputType

Specifies the output format. The default is **PDF**.

ENVIRONMENT

None.

FILES

None.

RESTRICTIONS

Requires Dispatcher, Adobe PDF Generator ES, and source authoring applications like the MS Office.

EXAMPLES

- To translate a MS Word file named **testfile.doc** file to a PDF file named **testfile.pdf**, enter the following command on a single line:

```
Pdfgenerator.bat -inputDir c:\temp\ -inputFile c:\temp\testfile.doc -outputDir  
c:\temp\ -outputType PDF
```

Chapter

*14 Teamcenter Integration for
NX utilities*

export_attr_mappings

Exports attribute mappings from the Teamcenter database to a text file.

SYNTAX

```
export_attr_mappings -file=text-file [-test]  
[-u=user-id {-p=password | -pf=password-file} -g=group]  
[-h]
```

ARGUMENTS**-u**

Specifies the user ID.

This is generally **infodba** or another user with administration privileges. If this argument is used without a value, the operating system user name is used.

Note

If Security Services single sign-on (SSO) is enabled for your server, the **-u** and **-p** arguments are authenticated externally through SSO rather than being authenticated against the Teamcenter database. If you do not supply these arguments, the utility attempts to join an existing SSO session. If no session is found, you are prompted to enter a user ID and password.

-p

Specifies the password.

If used without a value, the system assumes a null value.

If this argument is not used, the system assumes the *user-ID* value to be the password.

This argument is mutually exclusive with the **-pf** argument.

-pf

Specifies the password file. If used without a value, the system assumes a null value. If this argument is not used, the system assumes the *user-ID* value to be the password.

For more information about managing password files, see [Manage password files](#).

This argument is mutually exclusive with the **-p** argument.

-g

Specifies the group associated with the user.

If used without a value, the user's default group is assumed.

-file

Specifies the name of the file to be written to.

-test

Exports the test mappings rather than the actual mappings.

-h

Displays help for this utility.

ENVIRONMENT

This utility must be run in the Teamcenter shell environment.

FILES

As specified in *Log files produced by Teamcenter*.

RESTRICTIONS

None.

import_attr_mappings

Imports attribute mappings into the Teamcenter database. Siemens PLM Software recommends that you initially import the mappings using the **-test** argument and verify the accuracy of the mappings before making them generally available.

SYNTAX

```
import_attr_mappings -file= text-file [-test] [-dryrun]  
[-u=user-id {-p=password | -pf=password-file} -g=group] [-h]
```

ARGUMENTS**-u**

Specifies the user ID.

This is generally **infodba** or another user with administration privileges. If this argument is used without a value, the operating system user name is used.

Note

If Security Services single sign-on (SSO) is enabled for your server, the **-u** and **-p** arguments are authenticated externally through SSO rather than being authenticated against the Teamcenter database. If you do not supply these arguments, the utility attempts to join an existing SSO session. If no session is found, you are prompted to enter a user ID and password.

-p

Specifies the password.

If used without a value, the system assumes a null value.

If this argument is not used, the system assumes the *user-ID* value to be the password.

This argument is mutually exclusive with the **-pf** argument.

-pf

Specifies the password file. If used without a value, the system assumes a null value. If this argument is not used, the system assumes the *user-ID* value to be the password.

For more information about managing password files, see [Manage password files](#).

This argument is mutually exclusive with the **-p** argument.

-g

Specifies the group associated with the user.

If used without a value, the user's default group is assumed.

-file= *text-file*

Specifies the name of the input file.

-test

Imports the file without overwriting the existing mapping file.

-dryrun

Parses the file but does not save the mappings.

-h

Displays help for this utility.

ENVIRONMENT

This utility should be run from a shell where the Teamcenter environment is set.

FILES

As specified in *Log files produced by Teamcenter*.

RESTRICTIONS

None.

refile_info

Generates a list of identifiers of every item revision in the database. The identifiers include keys. This utility is used in conjunction with the NX **ug_refile** utility to refile all **UGMASTER** and/or **UGPART** datasets in the database. Refer to NX online help for additional information.

SYNTAX

refile_info [-u=user-id {-p=password | -pf=password-file} -g=group]
-o=file-name

ARGUMENTS**-u**

Specifies the user ID.

This is generally **infodba** or another user with administration privileges. If this argument is used without a value, the operating system user name is used.

Note

If Security Services single sign-on (SSO) is enabled for your server, the **-u** and **-p** arguments are authenticated externally through SSO rather than being authenticated against the Teamcenter database. If you do not supply these arguments, the utility attempts to join an existing SSO session. If no session is found, you are prompted to enter a user ID and password.

-p

Specifies the password.

If used without a value, the system assumes a null value.

If this argument is not used, the system assumes the *user-ID* value to be the password.

This argument is mutually exclusive with the **-pf** argument.

-pf

Specifies the password file. If used without a value, the system assumes a null value. If this argument is not used, the system assumes the *user-ID* value to be the password.

For more information about managing password files, see [Manage password files](#).

This argument is mutually exclusive with the **-p** argument.

-g

Specifies the group associated with the user.

If used without a value, the user's default group is assumed.

-o

Specifies the name of the file to which the list of item revision identifiers is written. If no output file name is specified, the default file name **item_revision_list** is used.

ENVIRONMENT

As specified in [Manually configuring your environment for Teamcenter utilities](#).

FILES

As specified in [Log files produced by Teamcenter](#).

RESTRICTIONS

None.

EXAMPLES

To generate a list of identifiers of all item revisions in the database and write it to a file called **item_revision_list**, enter the following command on a single line:

```
$TC_ROOT/bin/refile_info -u=infodba -p=password -g=dba -o=item_revision_list
```

tess_server

Starts the tessellation server which translates **UGMASTER/UGALTREP** datasets to JT datasets.

SYNTAX

tess_server {-s | -c} [-h] [-u=user-id {-p=password | -pf=password-file} -g=group]
{-s | -c} [-h]

ARGUMENTS**-u**

Specifies the user ID.

This is generally **infodba** or another user with administration privileges. If this argument is used without a value, the operating system user name is used.

Note

If Security Services single sign-on (SSO) is enabled for your server, the **-u** and **-p** arguments are authenticated externally through SSO rather than being authenticated against the Teamcenter database. If you do not supply these arguments, the utility attempts to join an existing SSO session. If no session is found, you are prompted to enter a user ID and password.

-p

Specifies the password.

If used without a value, the system assumes a null value.

If this argument is not used, the system assumes the *user-ID* value to be the password.

This argument is mutually exclusive with the **-pf** argument.

-pf

Specifies the password file. If used without a value, the system assumes a null value. If this argument is not used, the system assumes the *user-ID* value to be the password.

For more information about managing password files, see [Manage password files](#).

This argument is mutually exclusive with the **-p** argument.

-g

Specifies the group associated with the user.

If used without a value, the user's default group is assumed.

-s

Starts the tessellation server.

-c

Stops the tessellation server.

-h

Displays help for this utility.

ENVIRONMENT

As specified in [Manually configuring your environment for Teamcenter utilities](#).

FILES

As specified in *Log files produced by Teamcenter*.

RESTRICTIONS

None.

EXAMPLES

- Enter the following command to start the tessellation server:

```
tess_server -s
```

- Enter the following command to stop the tessellation server:

```
tess_server -c
```

- Enter the following command to display help for the **tess_server** utility:

```
tess_server -h
```

nxmgr_upgrade_bvrsyncform

Upgrades assemblies created in a Teamcenter Engineering version earlier than 8.0 to use the **BVRSyncInfo Form**. The form is created only if the last synchronization date exists in the **PDI reprev** cache.

SYNTAX

```
nxmgr_upgrade_bvrsyncform [-u=user-id {-p=password |  
-pf=password-file} -g=group]  
[-input_list=file-name] [-folder=folder-name] [-item=item-id] [-rev=revision-id]  
[-output_file=file-name] [-log_file=file-name]  
[-bypass=boolean] [-resume_from=line-number] [-update_mod_props=boolean]  
[-upgrade_released=boolean] [-h]
```

ARGUMENTS**-u**

Specifies the user ID.

This is generally **infodba** or another user with administration privileges. If this argument is used without a value, the operating system user name is used.

Note

If Security Services single sign-on (SSO) is enabled for your server, the **-u** and **-p** arguments are authenticated externally through SSO rather than being authenticated against the Teamcenter database. If you do not supply these arguments, the utility attempts to join an existing SSO session. If no session is found, you are prompted to enter a user ID and password.

-p

Specifies the password.

If used without a value, the system assumes a null value.

If this argument is not used, the system assumes the *user-ID* value to be the password.

This argument is mutually exclusive with the **-pf** argument.

-pf

Specifies the password file. If used without a value, the system assumes a null value. If this argument is not used, the system assumes the *user-ID* value to be the password.

For more information about managing password files, see [Manage password files](#).

This argument is mutually exclusive with the **-p** argument.

-g

Specifies the group associated with the user.

If used without a value, the user's default group is assumed.

-input_list

Specifies the name of the file containing a list of specifications (either as handles or in **@DB/item/rev** format), of items to upgrade. This argument is optional, but you must specify one of the following arguments: **-item**, **-input_list**, or **-folder**.

-folder

Specifies the name of the folder listing items to upgrade. This argument is optional, but you must specify one of the following arguments: **-item**, **-input_list**, or **-folder**.

-item

Specifies the ID of the item to upgrade (upgrades all the revisions). This argument is optional, but you must specify one of the following arguments: **-item**, **-input_list**, or **-folder**.

-rev

Specifies the revision ID of the item to upgrade. Use with the **-item** argument.

-output_file

Specifies the name of the file to write failure information to.

-log_file

Specifies the name of the file to write log information to.

-bypass

Specifies whether to use bypass privilege if necessary. Valid values are **yes** and **no**.

-resume_from

Specifies the line number of the input list to resume processing from.

The **-resume_from** argument is applicable only if the **-item** argument is used.

-update_mod_props

Specifies whether to update the last modifying user and date on objects. Valid values are **yes** and **no**.

-upgrade_released

Specifies whether to upgrade item revisions with release status. Valid values are **yes** and **no**.

-h

Displays help for the utility.

ENVIRONMENT

Requires the standard Teamcenter environment for running Integration Toolkit programs.

FILES

As specified in [Log files produced by Teamcenter](#).

RESTRICTIONS

This utility is supported for Hewlett-Packard HP-UX, Sun Solaris, and Microsoft Windows systems only.

EXAMPLES

- The following example displays the usage message:

```
nxmgr_upgrade_bvrsyncform -h
```

- The following example upgrades the **UGMASTER** dataset attached under the **parent A** item with the new **BVRSYNCINFO** named reference:

```
nxmgr_upgrade_bvrsyncform -item=parent -rev=A
```

- The following example upgrades all parts in the **ForUpgrade** folder as **infodba**. The last modified dates are not updated during this upgrade:

```
nxmgr_upgrade_bvrsyncform -u=infodba -p=password -g=dba  
-folder=ForUpgrade -bypass=yes -update_mod_props=no
```

- The following example upgrades all revisions of the **top** item:

```
nxmgr_upgrade_bvrsyncform -item=top -bypass=yes
```

- The following example upgrades the items contained in the **list.txt** file (either as handles or in **@DBitem/rev** format).

```
nxmgr_upgrade_bvrsyncform -i=list.txt
```

nxmlgr_upgrade_transforms

Upgrades transforms.

SYNTAX

```
nxmlgr_upgrade_transforms [-u=user-id {-p=password |  
-pf=password-file} -g=group]  
[-i=input-list-file-name] [-f=folder-name] [-item=item-id] [-rev=revision-id]  
[-out=output-file-name] [-log=log-file-name] [-bypass=boolean] [-resume=line-number]  
[-update_mod=boolean] [-upgrade_release=boolean]  
[-force_units=measurement-unit]  
[-h]
```

ARGUMENTS

-u

Specifies the user ID.

This is generally **infodba** or another user with administration privileges. If this argument is used without a value, the operating system user name is used.

Note

If Security Services single sign-on (SSO) is enabled for your server, the **-u** and **-p** arguments are authenticated externally through SSO rather than being authenticated against the Teamcenter database. If you do not supply these arguments, the utility attempts to join an existing SSO session. If no session is found, you are prompted to enter a user ID and password.

-p

Specifies the password.

If used without a value, the system assumes a null value.

If this argument is not used, the system assumes the *user-ID* value to be the password.

This argument is mutually exclusive with the **-pf** argument.

-pf

Specifies the password file. If used without a value, the system assumes a null value. If this argument is not used, the system assumes the *user-ID* value to be the password.

For more information about managing password files, see [Manage password files](#).

This argument is mutually exclusive with the **-p** argument.

-g

Specifies the group associated with the user.

If used without a value, the user's default group is assumed.

-i

Specifies the name of the file containing the input list of specifications (either as handles or in **@DB/item/rev** format), of items to upgrade. This argument is optional, but you must specify one of the following arguments: **-item**, **-input_list**, or **-folder**.

-f

Specifies the name of the folder listing items to upgrade. This argument is optional, but you must specify one of the following arguments: **-item**, **-input_list**, or **-folder**.

-item

Specifies the ID of the item to upgrade (upgrades all the revisions). This argument is optional, but you must specify one of the following arguments: **-item**, **-input_list**, or **-folder**.

-rev

Specifies the revision ID of the item to upgrade. Use with the **-item** argument.

-out

Specifies the name of the output file to which failure information is written.

-log

Specifies the name of the log file to which log information is written.

-bypass

Specifies whether to use bypass privilege if necessary. Valid values are **yes** and **no**.

-resume

Specifies the line number of the input list from which to resume upgrade processing.

This argument is applicable only if the **-item** argument is used.

-update_mod

Specifies whether to update the last modifying user and date on objects. Valid values are **yes** and **no**.

-upgrade_release

Specifies whether to upgrade item revisions with release status. Valid values are **yes** and **no**.

-force_units

Specifies measurement unit of revisions being upgraded. Valid values are **inches** and **millimeters**.

Note

Any BOMView Revision containing a **legacy_transform_factors** setting must be set to either **0.0254** (representing inches) or **0.001** (representing millimeters). These are the only valid values for this setting.

Caution

Be extremely cautious when specifying the **-force_units** option. Use this option only if you are absolutely certain of the units of the assembly part being upgraded.

-h

Displays help for the utility.

ENVIRONMENT

Requires the standard Teamcenter environment for running Integration Toolkit programs.

FILES

As specified in *Log files produced by Teamcenter*.

RESTRICTIONS

This utility is supported for Hewlett-Packard HP-UX, Sun Solaris, and Microsoft Windows systems only.

EXAMPLES

- The following example displays the usage message:

```
nxmgr_upgrade_transforms -h
```

- The following example upgrades the transforms for the **parent A** part. This part must be created by NX 18 or later and have a **UGPART-ATTRIBUTES** named reference form attached to the **UGMASTER** dataset.

```
nxmgr_upgrade_transforms -item=parent -rev=A
```

- The following example upgrades all parts in the **ForUpgrade** folder as **infodba**. The last modified dates are not updated during this upgrade:

```
nxmgr_upgrade_transforms -u=infodba -p=password -g=dba -folder=ForUpgrade  
-bypass=yes -update_mod_props=no
```

- The following example upgrades all revisions of the **top** item asserting that this part was modelled in inches:

```
nxmgr_upgrade_transforms -item=top -bypass=yes -force_units=inches
```

- The following example upgrades all revisions having a release status:

```
nxmgr_upgrade_transforms -u=infodba -p=password -g=dba  
input_list=file-name  
output_file=file-name log_file=file-name  
-upgrade_released=yes -bypass=yes
```

Chapter

15 Integration utilities

Teamcenter/Community Collaboration

tcc_context_upload

Uploads fully or partially configured assemblies to Teamcenter Community after downloading them from Teamcenter. This utility does the following:

1. Sets up the staging directory to store the temporary files.
2. Executes a search in RDV setup and downloads the resulting data into a flat file (AJT or PLM XML) by calling the **rdv_context_download** utility.
3. Calls the **asciitojt** utility to convert the AJT file downloaded in Step 2 into the JT format.
4. Uploads the JT assembly files into Teamcenter Community.
5. Deletes the staging directory.

SYNTAX

```
tcc_context_upload [-u=user-id {-p=password | -pf=password-file} -g=group]
[-item_id=item-ID] [-rev_id=revision-ID]
[-variant_rule_name=variant-rule-name] [-revision_rule_name=revision-rule-name]
[-engg_change_id=engineering-change-ID]
[-sco_name=structure-context-object name]
[-folder_name=folder-name] [-process_name=process_name]
[-zone_name=zone-name] [-zone_type=BOX | PLANE] [-title=JT-file-base-name]
[-stage=staging-area] [-d] [-verbose] [-tccuser=TeamcenterCommunity-user-name]
[-tccpass=TeamcenterCommunity-password] [-tccanon]
[-tcccreds=TeamcenterCommunity-credential-file]
[-tccdestination=url] [-keep] [-h]
```

ARGUMENTS

-u

Specifies the user ID.

This is generally **infodba** or another user with administration privileges. If this argument is used without a value, the operating system user name is used.

Note

If Security Services single sign-on (SSO) is enabled for your server, the **-u** and **-p** arguments are authenticated externally through SSO rather than being authenticated against the Teamcenter database. If you do not supply these arguments, the utility attempts to join an existing SSO session. If no session is found, you are prompted to enter a user ID and password.

-p

Specifies the password.

If used without a value, the system assumes a null value.

If this argument is not used, the system assumes the *user-ID* value to be the password.

This argument is mutually exclusive with the **-pf** argument.

-pf

Specifies the password file. If used without a value, the system assumes a null value. If this argument is not used, the system assumes the *user-ID* value to be the password.

For more information about managing password files, see [Manage password files](#).

This argument is mutually exclusive with the **-p** argument.

-g

Specifies the group associated with the user.

If used without a value, the user's default group is assumed.

-sco_name

Specifies the name of the structure context object.

-item_id

Specifies the item ID.

-rev_id

Specifies the revision ID.

-key

Specifies the key of the item. Use the following format:

```
[keyAttr1=keyVal1] [,keyAttr2=keyVal2]...[,keyAttrN=keyValN]
```

To find the key of an object, use the **get_key_string** utility.

For more information, see the [Business Modeler IDE Guide](#).

-engg_change_id

Specifies the engineering change item ID. The utility configures an RDV context based on change attachments and the latest engineering change revision.

-process_name

Specifies the name of the Teamcenter workflow processes that have not yet completed.

-zone_name

Specifies the name of the zone. When both the **-zone_name** and **-zone_type** arguments are specified, the utility performs an appearance or QPL search according to the preference settings.

-zone_type

Specifies the type of the zone, which must be either **BOX** or **PLANE**. The **-zone_name** argument must be used in conjunction with the **-zone_type** argument.

-folder_name

Configures a context based on attachments of the *folder/envelope/engineering-change-revision-name*.

The attachments include the following:

- One product item revision
- One or more component item IDs

- Optional revision rule, overwrites the **-r** argument
- Optional variant rule, overwrites the **-v** argument

Note Search results are affected by the following Teamcenter preferences:

- **TC_config_rule_name**
- **WebDesignContextDefaultSearchDistance**
- **PortalDesignContextMaxMatchingObjects**
- **PortalDesignContextMaxMatchingBOMLines**

For more information, see the *Preferences and Environment Variables Reference*.

-variant_rule_name

Specifies the name of the variant rule.

-revision_rule_name

Specifies the revision rule, which defaults according to the **TC_config_rule_name** preference.

-title

Specifies the base name of the JT file. If not specified, it defaults to **-I**.

-stage

Specifies the staging area (for example, **/tmp**). If not specified, it defaults to **\$TC_TMP_DIR**.

-d

Turns on debugging for the **Create bookmark (rdv_context_download)** utility.

-verbose

Turns on debugging for this utility.

-tccuser

Specifies the Teamcenter Community user name if it is different than the user name in the **-u** argument.

-tccpass

Specifies the Teamcenter Community password if it is different than the password in the **-p** argument.

-tccanon

Sets the Teamcenter Community login to anonymous, assuming the destination permits anonymous access.

-tcccreds

Specifies the Teamcenter Community externally created credential file for logging in.

-tccdestination

Specifies the URL for Teamcenter Community.

-keep

Do not delete the staging area after uploading the JT assembly files.

-h

Displays help for this utility.

ENVIRONMENT

- The **TcCUploaderS.jar** file must be present in **CLASSPATH**.
- The Teamcenter environment must be set to run this utility.
- As mentioned in the [Dependencies](#) section below, the user must have a Teamcenter Community login and a Teamcenter Community environment set up.
- If the **IMAN_QPL_PROX_FILTER_INCL_COMPS** environment variable is set, proximity searches not only return nearby parts for an instance, but also for all of its children and grandchildren.
- If the **IMAN_RDV_VALID_OVERLAYS_ONLY** environment variable is set, background searches (the **-c** or **-n** argument) return only background component instances overlaying valid combinations of variants. This may cause long processing times.
- If the environment variable **RDVContextDownloadDebug=1** is set, the Teamcenter **syslog** file contains information about the **rdv_context_download** utility.
- If the environment variable **RDV_debug=Init+QPL+SearchCriteria+Variants** is set, the Teamcenter **syslog** file contains additional RDV debugging information.

DEPENDENCIES

- This utility depends on the **TcCUploaderS.jar** file. Before using this utility, you must get this file from the Teamcenter Community kit and place it in the **CLASSPATH** on each client host.
- You must have access to the Teamcenter Community environment to run this utility.

RESTRICTIONS

- Zone searches (**-z**) require an NX-based QPL search index with zones or zone filters explicitly created in a product structure for appearance caches.
- Context searches (**-c** and **-n**) require a QPL search index.

The attachments include the following:

- o One product item revision
- o One or more component item IDs
- o Optional revision rule, overwrites the **-r** argument
- o Optional variant rule, overwrites the **-v** argument

Note Search results are affected by the following Teamcenter preferences:

- o **TC_config_rule_name**
- o **WebDesignContextDefaultSearchDistance**
- o **PortalDesignContextMaxMatchingObjects**
- o **PortalDesignContextMaxMatchingBOLMLines**

For more information, see the [Preferences and Environment Variables Reference](#).

EXAMPLES

- The following example uploads the antenna assembly to Teamcenter Community, overlaying all applicable variants (the RDV search index is not used or needed). It uses the **TC_config_rule_name** user preference to determine the revision rule.

```
$TC_ROOT/bin/tcc_context_upload -item_id TL109375 -rev_id 004
-ttitle Antenna -stage /tmp -tccuser subrata
-tccdestination http://usamseveh001/mydocs
```

- The following example uploads the antenna assembly to Teamcenter Community, overlaying all applicable variants (the RDV search index is not used or needed). It enforces a revision configuration using the **Beta or less w/pdi** revision rule.

```
$TC_ROOT/bin/tcc_context_upload -item_id TL109375 -rev_id 004
-ttitle Antenna -revision_rule_name "Beta or less w/pdi" -stage /tmp
-tccuser subrata -tccdestination http://usamseveh001/mydocs
```

- The following example uploads a subset of the RDV00190 product assembly that contains all components in the ENGINE zone to Teamcenter Community. It also:
 - o Requires an NX ENGINE box zone in the top-level NX part file
 - o Requires an NX-based QPL search index
 - o Overlays all applicable variants
 - o May require a higher value for the **PortalDesignContextMaxMatchingBOLMLines** preference if the ENGINE zone has many components.

```
$TC_ROOT/bin/tcc_context_upload -item_id RDV00190 -rev_id 008
-ttitle EngineCompartment - revision_rule_name "Beta or less w/pdi"
-zone_name ENGINE -stage /tmp -tccuser subrata
-tccdestination http://usamseveh001/mydocs
```

- The following example uploads a subset of the product assembly referenced in the latest revision of the 000042RDV Engineering Change Item, including affected components and their nearby parts within the distance specified in the **WebDesignContextDefaultSearchDistance** preference. Please note:
 - o 000042RDV is expected to reference the following:
 - The affected parts or part revisions (for example, 15759576/003-RADIATOR ASM-(W/ A/C CNDSR)

15006864/015-REINF-RAD UPPER INR SUPT LH
15068174/002-SUPPORT_ASM_RADIATOR).

- The product item revision (for example, RDV00190/008).
- The revision rule (for example, **Beta or less w/pdi**)
- o The subset contains all components inside the ENGINE zone.
- o It requires a QPL search index.
- o It overlays all applicable variants.

```
$TC_ROOT/bin/tcc_context_upload -engg_change_id 000042RDV  
-title RadiatorSupport -stage /tmp -tccuser subrata  
-tccdestination http://usamseveh001/mydocs
```

Teamcenter/System Engineering and Requirements Management

proxy_sync

Synchronizes Teamcenter objects that are linked to remote Teamcenter systems engineering and requirements management applications, using information stored in the **ExportedProxyLink** class to determine which objects must be synchronized. The utility can query and select a subset of records based on a specified foreign application, date, and status. You can also determine whether to synchronize objects that were modified, objects that were deleted, or objects for which the links (proxies) have been deleted. This utility also provides records of links from the local application.

In addition, you can use the **-diagnostic** parameter to test the setup environment for linking with any type of Teamcenter application.

SYNTAX

```
proxy_sync [-u=user-id {-p=password | -pf=password-file} -g=group]  
-obj_uid=obj_uid -app_guide=app_guide -force -sync -report -delete -proxy_report  
-diagnostic -remote_app_guide -remote_app_type tcprj | tcreq | tceng -h
```

ARGUMENTS**-u**

Specifies the user ID.

This is generally **infodba** or another user with administration privileges. If this argument is used without a value, the operating system user name is used.

Note

If Security Services single sign-on (SSO) is enabled for your server, the **-u** and **-p** arguments are authenticated externally through SSO rather than being authenticated against the Teamcenter database. If you do not supply these arguments, the utility attempts to join an existing SSO session. If no session is found, you are prompted to enter a user ID and password.

-p

Specifies the password.

If used without a value, the system assumes a null value.

If this argument is not used, the system assumes the *user-ID* value to be the password.

This argument is mutually exclusive with the **-pf** argument.

-pf

Specifies the password file. If used without a value, the system assumes a null value. If this argument is not used, the system assumes the *user-ID* value to be the password.

For more information about managing password files, see [Manage password files](#).

This argument is mutually exclusive with the **-p** argument.

-g

Specifies the group associated with the user.

If used without a value, the user's default group is assumed.

-obj_uid= *obj_uid*

Selects records of the Teamcenter objects specified by the object UID.

Note The **-app_guid** and **-obj_uid** arguments form a condition based on the selected records. However, there is also an implicit condition which specifies that only objects that have been modified after the last synchronization are selected.

-app_guid= *app_guid*

Selects records that were exported to the foreign application identified by the specified application GUID.

-force

Selects and synchronizes objects regardless of the last modification date.

-sync

Note Either the **-sync**, **-delete**, or **-report** operation must be specified when running this utility.

Performs the synchronization.

-report

Displays the queried objects.

-delete

Deletes the queried objects.

-proxy_report

Lists all proxies in the local database. No other parameter is taken into consideration when this argument is specified.

-diagnostic

Performs diagnostics that check the values of various preferences and records, communication with the application registry and communication with a remote application, as indicated by the **-remote_app_guid** parameter.

-remote_app_guid

Specifies the application that the utility attempts to contact. This argument is mandatory when the **-diagnostic** argument is used.

-remote_app_type

Specifies one of the following application types:

- **tcprij**
- **tcreq**
- **tceng**

This argument is mandatory when the **-diagnostic** argument is used.

-debug

Prints information to the standard output.

-h

Displays help for this utility.

ENVIRONMENT

As specified in *Manually configuring your environment for Teamcenter utilities*.

FILES

As specified in *Log files produced by Teamcenter*.

EXAMPLES

None.

Chapter

*16 Teamcenter Automotive
Edition-GM Overlay utilities*

find_released_datasets

Locates the item revisions and item IDs for the **UGMASTER** and **UGALTREP** datasets released after a certain date and that do not have a DirectModel dataset. The output is written to a file, which is a user argument for the utility. If this argument is not provided, output is written to **query_output.txt** in the current location. If no release date is provided, the system date is used.

SYNTAX

```
find_released_datasets [-u=user-id {-p=password | -pf=password-file} -g=group]
-released_date=date [-bypass= TRUE | FALSE]
[-out_file=output-file-name] [include_remote] [-h]
```

ARGUMENTS

-u

Specifies the user ID.

This is generally **infodba** or another user with administration privileges. If this argument is used without a value, the operating system user name is used.

Note

If Security Services single sign-on (SSO) is enabled for your server, the **-u** and **-p** arguments are authenticated externally through SSO rather than being authenticated against the Teamcenter database. If you do not supply these arguments, the utility attempts to join an existing SSO session. If no session is found, you are prompted to enter a user ID and password.

-p

Specifies the password.

If used without a value, the system assumes a null value.

If this argument is not used, the system assumes the *user-ID* value to be the password.

This argument is mutually exclusive with the **-pf** argument.

-pf

Specifies the password file. If used without a value, the system assumes a null value. If this argument is not used, the system assumes the *user-ID* value to be the password.

For more information about managing password files, see [Manage password files](#).

This argument is mutually exclusive with the **-p** argument.

-g

Specifies the group associated with the user.

If used without a value, the user's default group is assumed.

-released_date

Date on which the **UGMASTER/UGALTREP** datasets are released to which we are going to create the tessellated datasets. The format is *DD-MMM-YYYY*. This argument is optional. If not specified, system date is used as the released date.

-bypass

Indicates a bypass switch for having bypass privileges. The default value is **FALSE**.

-out_file

Specifies the file to which output is written. This argument is optional. If this is not specified, the file is written to the current location.

-include_remote

If the value given is not null, remote objects are included.

-h

Displays help for this utility.

EXAMPLES

```
find_released_datasets -u=infodba -p=infodba -g=dba
  -released_date=22-Mar-2005
  -out_file=c:\temp\find_released_datasets_test.txt
```

get_bvr_structure

Retrieves the assembly structure of a BVR of the specified item revision, and retrieves the child item IDs associated with the assembly structure, then writes these results to the output file along with the IR.

SYNTAX

get_bvr_structure [-u=*user-id* {-p=*password* | -pf=*password-file*} -g=*group*]
[-f=*input-file-name* | -itemRevisionKeyFile=*file-name*] [-h]

ARGUMENTS**-u**

Specifies the user ID.

This is generally **infodba** or another user with administration privileges. If this argument is used without a value, the operating system user name is used.

Note

If Security Services single sign-on (SSO) is enabled for your server, the **-u** and **-p** arguments are authenticated externally through SSO rather than being authenticated against the Teamcenter database. If you do not supply these arguments, the utility attempts to join an existing SSO session. If no session is found, you are prompted to enter a user ID and password.

-p

Specifies the password.

If used without a value, the system assumes a null value.

If this argument is not used, the system assumes the *user-ID* value to be the password.

This argument is mutually exclusive with the **-pf** argument.

-pf

Specifies the password file. If used without a value, the system assumes a null value. If this argument is not used, the system assumes the *user-ID* value to be the password.

For more information about managing password files, see [Manage password files](#).

This argument is mutually exclusive with the **-p** argument.

-pf

Specifies the password file. If used without a value, the system assumes a null value. If this argument is not used, the system assumes the *user-ID* value to be the password.

For more information about managing password files, see [Manage password files](#).

This argument is mutually exclusive with the **-p** argument.

-g

Specifies the group associated with the user.

If used without a value, the user's default group is assumed.

-f

Specifies the input file name containing a list of the specified item revisions.

-itemRevisionKeyFile

Specifies the input file name containing the keys of the specified item revisions.
The file format is:

```
-key = [keyAttr1=keyVal1] [keyAttr2=keyVal2]...  
-key = [keyAttr1=keyVal1] [keyAttr2=keyVal2]...  
-key = [keyAttr1=keyVal1] [keyAttr2='keyVal2']...
```

To find the key of an object, use the **get_key_string** utility.

For more information, see the [Business Modeler IDE Guide](#).

-h

Displays help for this utility.

Note This utility creates a log file in the **/tmp** directory and writes the error messages to this file. This utility also creates the output file in the current working directory and writes the NX component names with the item revision.

EXAMPLES

```
get_bvr_structure -u=infodba -p=infodba -g=dba -f=input_file
```

list_ir_with_bvr

Lists the item revisions with BVRs in the database to the **list_ir_with_bvr_XXXXXXX** output file.

SYNTAX

list_ir_with_bvr [-u=*user-id* {-p=*password* | -pf=*password-file*} -g=*group*]
-type=*CORP_Part Revision* -query=*defined-query-name*
-create_before=*date-time* -create_after=*date* [-h]

ARGUMENTS**-u**

Specifies the user ID.

This is generally **infodba** or another user with administration privileges. If this argument is used without a value, the operating system user name is used.

Note

If Security Services single sign-on (SSO) is enabled for your server, the **-u** and **-p** arguments are authenticated externally through SSO rather than being authenticated against the Teamcenter database. If you do not supply these arguments, the utility attempts to join an existing SSO session. If no session is found, you are prompted to enter a user ID and password.

-p

Specifies the password.

If used without a value, the system assumes a null value.

If this argument is not used, the system assumes the *user-ID* value to be the password.

This argument is mutually exclusive with the **-pf** argument.

-pf

Specifies the password file. If used without a value, the system assumes a null value. If this argument is not used, the system assumes the *user-ID* value to be the password.

For more information about managing password files, see [Manage password files](#).

This argument is mutually exclusive with the **-p** argument.

-g

Specifies the group associated with the user.

If used without a value, the user's default group is assumed.

-type

Specifies only the item revision type listed in the **list_ir_with_bvr_timeStamp.output** file.

-query

Specifies that the specified query is executed. This query must be defined in the database.

-create_before

Specifies the date to retrieve the list of item revisions created before this date. The format is *dd-mo-yyyy hh:mm*.

-create_after

Specifies the date to retrieve the list of item revisions created after this date. The format is *dd-mo-yyyy hh:mm*.

-h

Displays help for this utility.

migrate_gmo_to_gcn_events

Migrates all subscription events created in the GM Overlay using **CNonIR** project to create the GCN events. Migration is done by modifying attributes such as **event_type**, **attribute_names**, **attribute_values**, **logic_operators**, and **math_operators** on the **ImanSubscription** class.

SYNTAX

```
migrate_gmo_to_gcn_events [-u=user-id {-p=password |  
-pf=password-file} -g=group]  
[-list]  
-split=number-of-subscriptions-in-a-file] [-input_file=input-file]  
[-report=report-file] [-h]
```

ARGUMENTS

-u

Specifies the user ID.

This is generally **infodba** or another user with administration privileges. If this argument is used without a value, the operating system user name is used.

Note If Security Services single sign-on (SSO) is enabled for your server, the **-u** and **-p** arguments are authenticated externally through SSO rather than being authenticated against the Teamcenter database. If you do not supply these arguments, the utility attempts to join an existing SSO session. If no session is found, you are prompted to enter a user ID and password.

-p

Specifies the password.

If used without a value, the system assumes a null value.

If this argument is not used, the system assumes the *user-ID* value to be the password.

This argument is mutually exclusive with the **-pf** argument.

-pf

Specifies the password file. If used without a value, the system assumes a null value. If this argument is not used, the system assumes the *user-ID* value to be the password.

For more information about managing password files, see [Manage password files](#).

This argument is mutually exclusive with the **-p** argument.

-g

Specifies the group associated with the user.

If used without a value, the user's default group is assumed.

-list

Outputs all UIDs of instances of the **ImanSubscription** class in the database.

-split

Limits the number of subscriptions output to each file.

-input_file

Specifies the file containing the UIDs of the subscriptions. Only the valid GM Overlay subscription events specified in the input file are migrated to GCN events.

-report

Writes all messages and errors to the file specified by this argument.

-h

Displays help for this utility.

RESTRICTIONS

None.

EXAMPLES

- To list the UIDs of all the subscriptions in the database in an output file, enter the following command on a single line:

```
migrate_gmo_to_gcn_events -user=test-user -p=test-password -g=dba -list
```

- To output all UIDs of the subscriptions in the database to output files containing a maximum of 100 subscriptions per file, enter the following command on a single line:

```
migrate_gmo_to_gcn_events -user=test-user -p=test-password -g=dba  
-list -split=100
```

- To migrate all subscriptions in a given input file and write all messages and errors to a file specified by the report option, enter the following command on a single line:

```
migrate_gmo_to_gcn_events -user=test-user -p=test-password -g=dba  
-input_file=/tmp/migrate_input.txt -report=/tmp/migrate_report.txt
```

If the report option is not specified, the utility writes to the **migrate_gmo_to_gcn_events.txt** default file.

- To migrate all subscriptions in the database and report all messages and errors to the default report file using the default user login, enter the following command on a single line:

```
migrate_gmo_to_gcn_events
```

gmo_assoc_items_to_project

Converts special GM logic in the object description field to the project level security feature, as follows:

1. Queries all projects in the database.
2. For each project, searches the description field of all item revisions in the database for the following string:

`|project-id |`

If a match is found, the utility associates the corresponding item with the project.

The utility assumes the following:

- The pipe symbol is the delimiter used in the object description field.
- An item may be assigned to two different projects.
- Valid projects exist in the database.

SYNTAX

gmo_assoc_items_to_project [-u=*user-id* {-p=*password* | -pf=*password-file*} -g=*group*] [-h]

ARGUMENTS**-u**

Specifies the user ID.

This is generally **infodba** or another user with administration privileges. If this argument is used without a value, the operating system user name is used.

Note

If Security Services single sign-on (SSO) is enabled for your server, the **-u** and **-p** arguments are authenticated externally through SSO rather than being authenticated against the Teamcenter database. If you do not supply these arguments, the utility attempts to join an existing SSO session. If no session is found, you are prompted to enter a user ID and password.

-p

Specifies the password.

If used without a value, the system assumes a null value.

If this argument is not used, the system assumes the *user-ID* value to be the password.

This argument is mutually exclusive with the **-pf** argument.

-pf

Specifies the password file. If used without a value, the system assumes a null value. If this argument is not used, the system assumes the *user-ID* value to be the password.

For more information about managing password files, see [Manage password files](#).

This argument is mutually exclusive with the **-p** argument.

-g

Specifies the group associated with the user.

If used without a value, the user's default group is assumed.

-h

Displays help for this utility.

ENVIRONMENT

As specified in [*Manually configuring your environment for Teamcenter utilities.*](#)

FILES

None.

RESTRICTIONS

None.

EXAMPLES

- To obtain help for this utility, enter the following command on a single line:

```
gmo_assoc_items_to_project -h
```

- To associate all items in the database to appropriate projects based on the object description field, enter the following command on a single line:

```
gmo_assoc_items_to_project -u=infodba -p=infodba -g=dba
```

gmo_change_itemid_naming_rule

Modifies the item ID naming rule property from a given value to the customer specified value.

SYNTAX

gmo_change_itemid_naming_rule [-u=*user-id* {-p=*password* | -pf=*password-file*} -g=*group*] -option=*option* -prefix=*prefix* -nr_name=*naming-rule* -init_value=*initial-value* -max_value=*maximum-value* [-h]

ARGUMENTS**-u**

Specifies the user ID.

This is generally **infodba** or another user with administration privileges. If this argument is used without a value, the operating system user name is used.

Note

If Security Services single sign-on (SSO) is enabled for your server, the **-u** and **-p** arguments are authenticated externally through SSO rather than being authenticated against the Teamcenter database. If you do not supply these arguments, the utility attempts to join an existing SSO session. If no session is found, you are prompted to enter a user ID and password.

-p

Specifies the password.

If used without a value, the system assumes a null value.

If this argument is not used, the system assumes the *user-ID* value to be the password.

This argument is mutually exclusive with the **-pf** argument.

-pf

Specifies the password file. If used without a value, the system assumes a null value. If this argument is not used, the system assumes the *user-ID* value to be the password.

For more information about managing password files, see [Manage password files](#).

This argument is mutually exclusive with the **-p** argument.

-g

Specifies the group associated with the user.

If used without a value, the user's default group is assumed.

-option

Specifies the option for this utility, that is, **change_naming_rule**.

-prefix

Specifies the prefix of the GM ItemNumeric naming rule.

-nr_name

Specifies the naming rule to change the prefix. You can specify more than one naming rule separated by commas.

-init_value

Specifies the starting value of the counter.

-max_value

Specifies the maximum value of the counter.

-h

Displays help for this utility.

RESTRICTIONS

None.

EXAMPLES

- The following example uses the **GM ItemRule** naming rule to change the prefix:

```
gmo_change_itemid_naming_rule -u=user -p=password -g=group
-option=change_naming_rule -nr_name="GM ItemRule "
-prefix=GMO -init_value=00000 -max_value=99999
```

- The following example uses the **GM ItemRule** and **CORP_Tool Rule** naming rules to change the prefix:

```
gmo_change_itemid_naming_rule -u=user -p=password -g=group
-option=change_naming_rule -nr_name="GM ItemRule,CORP_Tool Rule"
-prefix=GMO -init_value=00000 -max_value=99999
```

gmo_change_owner

Sets user and group ownership of objects contained in a folder, item, or item revision. All processing procedures are logged to the log file, if specified.

SYNTAX

```
gmo_change_owner [-u=user-id {-p=password | -pf=password-file} -g=group]  
-folder=folder-name -item=item-id -rev=revision-id  
[-key=[keyAttr1=keyVal1] [,keyAttr2=keyVal2]...[,keyAttrN=keyValN]]  
[-r] -owner=new-user-id -own_grp=new-group  
-log=log-file-name -bypass [-h]
```

ARGUMENTS**-u**

Specifies the user ID.

This is generally **infodba** or another user with administration privileges. If this argument is used without a value, the operating system user name is used.

Note

If Security Services single sign-on (SSO) is enabled for your server, the **-u** and **-p** arguments are authenticated externally through SSO rather than being authenticated against the Teamcenter database. If you do not supply these arguments, the utility attempts to join an existing SSO session. If no session is found, you are prompted to enter a user ID and password.

-p

Specifies the password.

If used without a value, the system assumes a null value.

If this argument is not used, the system assumes the *user-ID* value to be the password.

This argument is mutually exclusive with the **-pf** argument.

-pf

Specifies the password file. If used without a value, the system assumes a null value. If this argument is not used, the system assumes the *user-ID* value to be the password.

For more information about managing password files, see [Manage password files](#).

This argument is mutually exclusive with the **-p** argument.

-g

Specifies the group associated with the user.

If used without a value, the user's default group is assumed.

-folder

Specifies the folder name. Either **folder** or **item** and/or **rev** should be supplied.

-item

Specifies the item ID.

-rev

Specifies the revision ID.

-key

Specifies the key of the item to be checked. The **-key** argument can be specified instead of the **-item** argument. Use the following format:

`[keyAttr1=keyVal1] [,keyAttr2=keyVal2]...[,keyAttrN=keyValN]`

To find the key of an object, use the [get_key_string](#) utility.

For more information, see the [Business Modeler IDE Guide](#).

-r

If specified, the ownership of the contents is changed.

-owner

Specifies the user ID to which the ownership is changed. Both **owner** and **own_grp** should be supplied.

-own_grp

Specifies the new group to which the ownership is changed.

-log

Specifies verbose messages are logged to this file.

-bypass

Bypass access checks. This argument is available only to the system administrator.

-h

Displays help for this utility.

EXAMPLES

None.

gmo_check_comp_names

Truncates the component names greater than 25 characters in length. This program locates the components of an assembly with component names greater than 25 characters long. The component name is the occurrence note of type name **UG NAME**.

SYNTAX

```
gmo_check_comp_names [-u=user-id {-p=password | -pf=password-file} -g=group]
-all | -f=folder-name | -i=item-id
| [-key=[keyAttr1=keyVal1] [,keyAttr2=keyVal2]...[,keyAttrN=keyValN]]
[-r=itemrev-id] [-report | update] [-out=output-file-name]
[-v] [-h]
```

ARGUMENTS**-u**

Specifies the user ID.

This is generally **infodba** or another user with administration privileges. If this argument is used without a value, the operating system user name is used.

Note

If Security Services single sign-on (SSO) is enabled for your server, the **-u** and **-p** arguments are authenticated externally through SSO rather than being authenticated against the Teamcenter database. If you do not supply these arguments, the utility attempts to join an existing SSO session. If no session is found, you are prompted to enter a user ID and password.

-p

Specifies the password.

If used without a value, the system assumes a null value.

If this argument is not used, the system assumes the *user-ID* value to be the password.

This argument is mutually exclusive with the **-pf** argument.

-pf

Specifies the password file. If used without a value, the system assumes a null value. If this argument is not used, the system assumes the *user-ID* value to be the password.

For more information about managing password files, see [Manage password files](#).

This argument is mutually exclusive with the **-p** argument.

-g

Specifies the group associated with the user.

If used without a value, the user's default group is assumed.

-all

Runs on all the items in database.

-f

Specifies the name of the folder containing list of items to be checked for component names.

-i

Specifies the item ID to be checked.

-key

Specifies the key of the item to be checked. Use the following format:

`[keyAttr1=keyVal1] [,keyAttr2=keyVal2]...[,keyAttrN=keyValN]`

To find the key of an object, use the [get_key_string](#) utility.

For more information, see the [Business Modeler IDE Guide](#).

-r

Specifies the revision ID to be checked.

-report

Generates the report of items with component names greater than 25 characters.

-update

Truncates all the component names greater than 25 characters.

-out

Specifies the name of the file to which output is written. This argument is optional. Default file name is **gmpdm_comp_name_report.txt**.

-v

Specifies verbose mode.

-h

Displays help for this utility.

RESTRICTIONS

This utility must be run by a user in the **dba** group with the **-update** option.

EXAMPLES

```
gmo_check_comp_names -u=infodba -p=infodba -g=dba
    -all -report -v
```

gmo_clone

Serves as a wrapper over the Teamcenter Integration for NX **ug_clone** utility to provide Teamcenter Automotive Edition–GM Overlay-specific clone import/export functionality from a command shell.

Use this utility to import and export NX data in the GM Overlay environment to ensure the clone conforms with GM Overlay naming conventions.

SYNTAX

```
gmo_clone [-pim=yes] [-u=user-id {-p=password | -pf=password-file} -g=group]
[-corba_ior_file=ior-file] [-http_url=4-tier-server-url]
[-o=operation] [-fam=lose | strip_status | error] [-asse=assembly] [-par=part]
[-dir=directory-name] [-fol=folder] [-default_checkin=default-check-in]
[-default_checkout=default-check-out] [-default_a=default-action]
[-default_n=default-naming] [-default_t=default-item-type]
[-asso=associated-directory] [-copy_a=copy-associated-files]
[-copy_n=copy-non-master-type] [-default_o=default-owner] [-l=load-log-file]
[-s=save-log-file] [-auto=translate_mode] [-propagate=yes/no]
[-export_dfa_kf=dfa_only | dva_in_part] [-export_dfa_list=] [-rev_up=]
[-attach_log_file=] [-copy_n=copy-non-master-type] [-dr=dryrun] [-h=help]
```

ARGUMENTS

-plm

Set **-plm** to **Yes** to initialize Teamcenter Integration for NX only, instead of native NX.

-u

Specifies the user ID.

This is generally **infodba** or another user with administration privileges. If this argument is used without a value, the operating system user name is used.

Note

If Security Services single sign-on (SSO) is enabled for your server, the **-u** and **-p** arguments are authenticated externally through SSO rather than being authenticated against the Teamcenter database. If you do not supply these arguments, the utility attempts to join an existing SSO session. If no session is found, you are prompted to enter a user ID and password.

-p

Specifies the password.

If used without a value, the system assumes a null value.

If this argument is not used, the system assumes the *user-ID* value to be the password.

This argument is mutually exclusive with the **-pf** argument.

-pf

Specifies the password file. If used without a value, the system assumes a null value. If this argument is not used, the system assumes the *user-ID* value to be the password.

For more information about managing password files, see [Manage password files](#).

This argument is mutually exclusive with the **-p** argument.

-g

Specifies the group associated with the user.

If used without a value, the user's default group is assumed.

-corba_ior_file

Specifies the Teamcenter password server IOR file.

-http_url

Specifies the HTTP URL for four-tier configuration.

-h

Displays help for this utility.

RESTRICTIONS

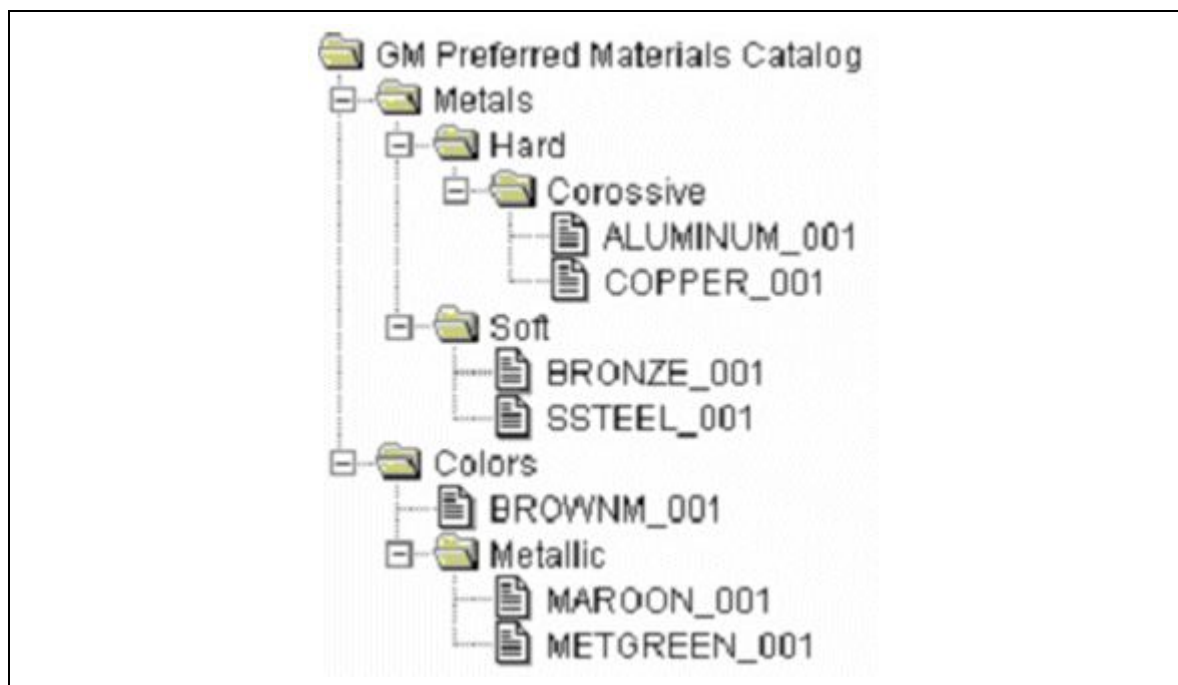
None.

gmo_create_material_form_templates

Creates new material form templates by reading the input from an ASCII text file. Input must be supplied in a defined format, as specified below. Material form templates are of the form type **Material** and are stored in folders or subfolders of the folder type **Material Template**. These folders and forms are stored in the **GM Preferred Materials Catalog** that is displayed in the **infodba** user's **Home** folder.

INPUT FILE FORMAT

This section describes the input file format using the sample folder structure shown in the following figure.



Sample directory structure

To achieve the structure shown in the sample directory structure, the input file must be in the following format:

```

<Folder1>[:<Folder2>:<Folder3>:<Folder4>:...]<Form Name>#value of p_mat
# value of p_pcoat# value of p_perf# value of p_pcperf# value of p_appear#
\value of p_fin# value of p_svc# value of p_addreq# value of p_mateng
# value of p_appeng# value of p_pnteng
  
```

Folders are delimited by a colon (:), folders and forms are delimited by a dollar sign (\$), and the form and form values are delimited by the (#) symbol.

Note The delimiting symbols described in the previous paragraph assume that these symbols are not used as values in any of the materials form fields.

The following is an example of the input file format.

```

Metals:Hard:Corossive$ALUMINUM_001#p_mat#p_pcoat#p_perf#p_pcperf
#p_appear#p_fin#p_svc#p_addreq#p_mateng#p_appeng#p_pnteng
Metals:Hard:Corossive$COPPER_001#p_mat#p_pcoat#p_perf#p_pcperf#p_appear
#p_fin#p_svc#p_addreq#p_mateng#p_appeng#p_pnteng
  
```

```

Metals:Soft$BRONZE_001#p_mat#p_pcoat#p_perf#p_pcperf#p_appear
#p_fin#p_svc#p_addreq#p_mateng#p_appeng#p_pnteng
Metals:Soft$SSTEEL_001#p_mat#p_pcoat#p_perf#p_pcperf#p_appear
#p_fin#p_svc#p_addreq#p_mateng#p_appeng#p_pnteng
Colors$BROWNM_001#p_mat#p_pcoat#p_perf#p_pcperf#p_appear
#p_fin#p_svc#p_addreq#p_mateng#p_appeng#p_pnteng
Colors:Metallic$MAROON_001#p_mat#p_pcoat#p_perf#p_pcperf#p_appear
#p_fin#p_svc#p_addreq#p_mateng#p_appeng#p_pnteng
Colors:Metallic$METGREEN_001#metallic#green#high#corrosive#greenish
#smooth#requires applying greese#none#user#user#user

```

SYNTAX

gmo_create_material_form_templates [-u=*user-id* {-p=*password* |
-pf=*password-file*}
-g=*group*]
[-infile=*full-path-to-input-file*] [-h]

ARGUMENTS**-u**

Specifies the user ID.

This is generally **infodba** or another user with administration privileges. If this argument is used without a value, the operating system user name is used.

Note

If Security Services single sign-on (SSO) is enabled for your server, the **-u** and **-p** arguments are authenticated externally through SSO rather than being authenticated against the Teamcenter database. If you do not supply these arguments, the utility attempts to join an existing SSO session. If no session is found, you are prompted to enter a user ID and password.

-p

Specifies the password.

If used without a value, the system assumes a null value.

If this argument is not used, the system assumes the *user-ID* value to be the password.

This argument is mutually exclusive with the **-pf** argument.

-pf

Specifies the password file. If used without a value, the system assumes a null value. If this argument is not used, the system assumes the *user-ID* value to be the password.

For more information about managing password files, see [Manage password files](#).

This argument is mutually exclusive with the **-p** argument.

-g

Specifies the group associated with the user.

If used without a value, the user's default group is assumed.

-infile

Specifies the full path to the input file.

-h

Displays help for this utility.

ENVIRONMENT

As specified in *Manually configuring your environment for Teamcenter utilities*.

FILES

As specified in *Log files produced by Teamcenter*. In addition, the **gmo_create_material_form_templates.log** file is created in the directory from which the utility is run.

RESTRICTIONS

None.

EXAMPLES

- To obtain help for this utility, enter the following command on a single line:

```
gmo_create_material_form_templates -h
```

- To read the supplied input file and create folders and forms that are inserted in the **GM Preferred Materials Catalog** folder, enter the following command on a single line:

```
$GMPDM_ROOT/bin/gmo_create_material_form_templates -u=infodba -p=infodba  
-g=dba -infile=/tmp/MATERIAL.txt
```


gmo_find_changed_install_assem

Locates the installation assemblies that have changed since the specified date and that are configured with the specified revision rules.

SYNTAX

```
gmo_find_changed_install_assem [-u=user-id {-p=password | -pf=password-file}  
-g=group]-time=date-time -revision_rule=rule1  
[-revision_rule=rule2 ...-revision_rule=rulen] -rev_rule_file=file-name  
[-obj_type=object-name] [-out_file=output-file-name] [-h]
```

ARGUMENTS

-u

Specifies the user ID.

This is generally **infodba** or another user with administration privileges. If this argument is used without a value, the operating system user name is used.

Note

If Security Services single sign-on (SSO) is enabled for your server, the **-u** and **-p** arguments are authenticated externally through SSO rather than being authenticated against the Teamcenter database. If you do not supply these arguments, the utility attempts to join an existing SSO session. If no session is found, you are prompted to enter a user ID and password.

-p

Specifies the password.

If used without a value, the system assumes a null value.

If this argument is not used, the system assumes the *user-ID* value to be the password.

This argument is mutually exclusive with the **-pf** argument.

-pf

Specifies the password file. If used without a value, the system assumes a null value. If this argument is not used, the system assumes the *user-ID* value to be the password.

For more information about managing password files, see [Manage password files](#).

This argument is mutually exclusive with the **-p** argument.

-g

Specifies the group associated with the user.

If used without a value, the user's default group is assumed.

-time

Specifies the date and time from which the **Released CORP_Install** item revisions are to be searched. This should be provided in the Teamcenter-specified format. An operating system file name can also be given whose last modification time will be taken as the time.

-revision_rule

Specifies revision rules for which the search is to be done. You can specify this argument multiple times. If this is given, the **rev_rule_file** argument cannot be used.

-rev_rule_file

Specifies a text file listing all the revision rules on separate lines. If this argument is given, **revision_rule** option cannot be used.

-obj_type

Specifies the type of object which is to be searched for the change in release status since the specified date. This argument is optional and defaults to objects of type **CORP_Install Revision**.

-out_file

Specifies file to which output is written. This argument is optional. If it is not specified, the output is sent to **stdout**.

-h

Displays help for this utility.

EXAMPLES

None.

gmo_get_partspec

Retrieves the part specification of the specified dataset.

SYNTAX

```
gmo_get_partspec [-u=user-id {-p=password | -pf=password-file} -g=group]  
-dataset_tag=dataset [-h]
```

ARGUMENTS

-u

Specifies the user ID.

This is generally **infodba** or another user with administration privileges. If this argument is used without a value, the operating system user name is used.

Note

If Security Services single sign-on (SSO) is enabled for your server, the **-u** and **-p** arguments are authenticated externally through SSO rather than being authenticated against the Teamcenter database. If you do not supply these arguments, the utility attempts to join an existing SSO session. If no session is found, you are prompted to enter a user ID and password.

-p

Specifies the password.

If used without a value, the system assumes a null value.

If this argument is not used, the system assumes the *user-ID* value to be the password.

This argument is mutually exclusive with the **-pf** argument.

-pf

Specifies the password file. If used without a value, the system assumes a null value. If this argument is not used, the system assumes the *user-ID* value to be the password.

For more information about managing password files, see [Manage password files](#).

This argument is mutually exclusive with the **-p** argument.

-g

Specifies the group associated with the user.

If used without a value, the user's default group is assumed.

-dataset_tag

Specifies the dataset tag in a string.

-h

Displays help for this utility.

EXAMPLES

None.

gmo_get_pds_info

Retrieves the **pds** attributes for the given part numbers. The output files are generated at the location specified by `%TC_TMP_DIR %` with the names **sitename_timestamp_parts_notfnd.txt** and **sitename_timestamp_parts_fnd.txt**.

SYNTAX

```
gmo_get_pds_info [-u=user-id {-p=password | -pf=password-file} -g=group]  
[-input_file=input-file-name | -itemKeyFile=file-name] [-h]
```

ARGUMENTS**-u**

Specifies the user ID.

This is generally **infodba** or another user with administration privileges. If this argument is used without a value, the operating system user name is used.

Note

If Security Services single sign-on (SSO) is enabled for your server, the **-u** and **-p** arguments are authenticated externally through SSO rather than being authenticated against the Teamcenter database. If you do not supply these arguments, the utility attempts to join an existing SSO session. If no session is found, you are prompted to enter a user ID and password.

-p

Specifies the password.

If used without a value, the system assumes a null value.

If this argument is not used, the system assumes the *user-ID* value to be the password.

This argument is mutually exclusive with the **-pf** argument.

-pf

Specifies the password file. If used without a value, the system assumes a null value. If this argument is not used, the system assumes the *user-ID* value to be the password.

For more information about managing password files, see [Manage password files](#).

This argument is mutually exclusive with the **-p** argument.

-g

Specifies the group associated with the user.

If used without a value, the user's default group is assumed.

-input_file

Specifies the input file with part numbers and revision IDs separated by the ~ (tilde) character.

-itemKeyFile

Specifies the input file name containing the keys of the desired part numbers. The file format is:

```
-key = [keyAttr1=keyVal1][keyAttr2=keyVal2]...  
-key = [keyAttr1=keyVal1][keyAttr2=keyVal2]...
```

`-key =[keyAttr1=keyVal1] [keyAttr2='keyVal2']...`

-h

Displays help for this utility.

gmo_install_usage_queries

Installs usage queries.

SYNTAX

```
gmo_install_usage_queries [-u=user-id {-p=password |  
-pf=password-file} -g=group]  
[-recreate] [-h]
```

ARGUMENTS**-u**

Specifies the user ID.

This is generally **infodba** or another user with administration privileges. If this argument is used without a value, the operating system user name is used.

Note

If Security Services single sign-on (SSO) is enabled for your server, the **-u** and **-p** arguments are authenticated externally through SSO rather than being authenticated against the Teamcenter database. If you do not supply these arguments, the utility attempts to join an existing SSO session. If no session is found, you are prompted to enter a user ID and password.

-p

Specifies the password.

If used without a value, the system assumes a null value.

If this argument is not used, the system assumes the *user-ID* value to be the password.

This argument is mutually exclusive with the **-pf** argument.

-pf

Specifies the password file. If used without a value, the system assumes a null value. If this argument is not used, the system assumes the *user-ID* value to be the password.

For more information about managing password files, see [Manage password files](#).

This argument is mutually exclusive with the **-p** argument.

-g

Specifies the group associated with the user.

If used without a value, the user's default group is assumed.

-recreate

Optional parameter. If specified, currently installed usage queries are deleted prior to installation of corresponding GMO-specific usage queries. As default, usage queries are normally installed. Siemens PLM Software recommends you use this parameter to avoid installation error messages when attempting to install over existing queries. This utility can be executed repeatedly using this option.

-h

Displays help for this utility.

ENVIRONMENT

As specified in *Manually configuring your environment for Teamcenter utilities*.

FILES

As specified in *Log files produced by Teamcenter*.

RESTRICTIONS

None.

gmo_ipvbom_import

Imports build intent data from a PLM XML file and creates a change object of type **GM Build Intent**.

SYNTAX

gmo_ipvbom_import [-u=*user-id* {-p=*password* | -pf=*password-file*} -g=*group*]
-xml_file=*full-path-to-PLM XML-file* [-h]

ARGUMENTS**-u**

Specifies the user ID.

This is generally **infodba** or another user with administration privileges. If this argument is used without a value, the operating system user name is used.

Note

If Security Services single sign-on (SSO) is enabled for your server, the **-u** and **-p** arguments are authenticated externally through SSO rather than being authenticated against the Teamcenter database. If you do not supply these arguments, the utility attempts to join an existing SSO session. If no session is found, you are prompted to enter a user ID and password.

-p

Specifies the password.

If used without a value, the system assumes a null value.

If this argument is not used, the system assumes the *user-ID* value to be the password.

This argument is mutually exclusive with the **-pf** argument.

-pf

Specifies the password file. If used without a value, the system assumes a null value. If this argument is not used, the system assumes the *user-ID* value to be the password.

For more information about managing password files, see [Manage password files](#).

This argument is mutually exclusive with the **-p** argument.

-g

Specifies the group associated with the user.

If used without a value, the user's default group is assumed.

-xml_file

Specifies the path to the PLM XML input file containing the build intents.

-h

Displays help for this utility.

ENVIRONMENT

As specified in [Manually configuring your environment for Teamcenter utilities](#).

FILES

As specified in [Log files produced by Teamcenter](#).

RESTRICTIONS

This utility must be used only to import build intent data. It is not intended to import other data contained in PLM XML files.

EXAMPLES

- To obtain help for this utility, enter the following command on a single line:

```
gmo_ipvbom_import -h
```

- To import build intent data contained in the **/tmp/IMPORT.xml** file, enter the following command on a single line:

```
gmo_ipvbom_import -u=infodba -p=infodba -g=dba -xml_file=/tmp/IMPORT.xml
```

gmo_ipvbom_export

Exports specific build intent information, all build intent information, specific build intents along with partial build intents, full BOM or incremental BOM data.

SYNTAX

```
gmo_ipvbom_export [-u=user-id {-p=password | -pf=password-file} -g=group]  
[-build_id=build-intent-id-number] [-fullbom=yes | no]  
[-inputfile=full-path-to-inputfile] -xml_path=full-path-to-PLM XML-file [-h]
```

ARGUMENTS**-u**

Specifies the user ID.

This is generally **infodba** or another user with administration privileges. If this argument is used without a value, the operating system user name is used.

Note

If Security Services single sign-on (SSO) is enabled for your server, the **-u** and **-p** arguments are authenticated externally through SSO rather than being authenticated against the Teamcenter database. If you do not supply these arguments, the utility attempts to join an existing SSO session. If no session is found, you are prompted to enter a user ID and password.

-p

Specifies the password.

If used without a value, the system assumes a null value.

If this argument is not used, the system assumes the *user-ID* value to be the password.

This argument is mutually exclusive with the **-pf** argument.

-pf

Specifies the password file. If used without a value, the system assumes a null value. If this argument is not used, the system assumes the *user-ID* value to be the password.

For more information about managing password files, see [Manage password files](#).

This argument is mutually exclusive with the **-p** argument.

-g

Specifies the group associated with the user.

If used without a value, the user's default group is assumed.

-build_id

Specifies the ID number of the build intent to be exported.

-fullbom

Specifies whether or not to export the full BOM. Valid values are **yes** or **no**. If **yes**, the utility exports the full BOM; otherwise, the delta BOM is exported.

The default value is **yes**.

-inputfile

Full path to file.

-xml_path

Specifies the path to the output directory containing the XML file. If not specified, the path defined in the preference file is used.

-h

Displays help for this utility.

ENVIRONMENT

As specified in *Manually configuring your environment for Teamcenter utilities*. In addition, the **IPVBOM_build_intent_status** preference, which specifies the release status used to obsolete older revisions of the build intent changes, must be set.

FILES

As specified in *Log files produced by Teamcenter*. In addition, the values of the **IPVBOM_compare_mode_var_level_RevisionCompare_ItemTypes** and **IPVBOM_compare_mode_var_level_Occurrence_Notes** preferences affect the behavior of the **gmiman_export** utility. For more information about these preferences, see the *Preferences and Environment Variables Reference*.

RESTRICTIONS

This utility must be used only to export build intent data. It is not intended to export other data to PLM XML files.

EXAMPLES

- To obtain help for this utility, enter the following command on a single line:

```
gmo_ipvbom_export -h
```

- To export build BOMs listed in the **/tmp/EXPORTS.txt** file from the Teamcenter database to a PLM XML file, enter the following command on a single line:

```
gmo_ipvbom_export -u=infodba -p=infodba -g=dba -fullbom=yes  
-xml_path=/tmp -inputfile=/tmp/EXPORT.txt
```

gmo_ipvbom_pulldate

Updates the pull date information for each build intent that is defined in a PLM XML file.

SYNTAX

gmo_ipvbom_pulldate [-u=*user-id* {-p=*password* | -pf=*password-file*} -g=*group*] [-xml_file=*absolute-path-of-xml-file*] [-h]

ARGUMENTS**-u**

Specifies the user ID.

This is generally **infodba** or another user with administration privileges. If this argument is used without a value, the operating system user name is used.

Note

If Security Services single sign-on (SSO) is enabled for your server, the **-u** and **-p** arguments are authenticated externally through SSO rather than being authenticated against the Teamcenter database. If you do not supply these arguments, the utility attempts to join an existing SSO session. If no session is found, you are prompted to enter a user ID and password.

-p

Specifies the password.

If used without a value, the system assumes a null value.

If this argument is not used, the system assumes the *user-ID* value to be the password.

This argument is mutually exclusive with the **-pf** argument.

-pf

Specifies the password file. If used without a value, the system assumes a null value. If this argument is not used, the system assumes the *user-ID* value to be the password.

For more information about managing password files, see [Manage password files](#).

This argument is mutually exclusive with the **-p** argument.

-g

Specifies the group associated with the user.

If used without a value, the user's default group is assumed.

-xml_file

Specifies the full path of the PLM XML file.

-h

Displays help for this utility.

ENVIRONMENT

As specified in [Manually configuring your environment for Teamcenter utilities](#).

FILES

As specified in [Log files produced by Teamcenter](#).

RESTRICTIONS

None.

EXAMPLES

- To obtain help for this utility, enter the following command on a single line:

```
gmo_ipvbom_pulldate -h
```

- To update pulldate information for each build intent contained in the **/tmp/PULLDATE.xml** file, enter the following command on a single line:

```
gmo_ipvbom_pulldate -u=infodba -g=dba -xml_file=c:/tmp/PULLDATE.xml
```

gmo_migrate_ulink_to_rdvauto

Migrates ULink occurrences in a VAS to GRDVA occurrences.

SYNTAX

```
gmo_migrate_ulink_to_rdvauto [-u=user-id {-p=password | -pf=password-file}  
-g=group]  
[-product=product-item-id | -key=[keyAttr1=keyVal1] [,keyAttr2=keyVal2]...  
[,keyAttrN=keyValN]]  
-rev=product-revision-id -ia_list=IA-item-id-file  
-process_path=y | -process_all=y | -process_rules=y  
-logfile=logfile-name [-h]
```

ARGUMENTS**-u**

Specifies the user ID.

This is generally **infodba** or another user with administration privileges. If this argument is used without a value, the operating system user name is used.

Note

If Security Services single sign-on (SSO) is enabled for your server, the **-u** and **-p** arguments are authenticated externally through SSO rather than being authenticated against the Teamcenter database. If you do not supply these arguments, the utility attempts to join an existing SSO session. If no session is found, you are prompted to enter a user ID and password.

-p

Specifies the password.

If used without a value, the system assumes a null value.

If this argument is not used, the system assumes the *user-ID* value to be the password.

This argument is mutually exclusive with the **-pf** argument.

-pf

Specifies the password file. If used without a value, the system assumes a null value. If this argument is not used, the system assumes the *user-ID* value to be the password.

For more information about managing password files, see [Manage password files](#).

This argument is mutually exclusive with the **-p** argument.

-g

Specifies the group associated with the user.

If used without a value, the user's default group is assumed.

-product

Specifies VAS item ID.

-key

Specifies the key of the VAS item. Use the following format:

```
[keyAttr1=keyVal1] [,keyAttr2=keyVal2]...[,keyAttrN=keyValN]
```

To find the key of an object, use the **get_key_string** utility.

For more information, see the *Business Modeler IDE Guide*.

-rev

Specifies VAS revision ID.

-logfile

Specifies the name of the logging file.

-h

Displays help for this utility.

ENVIRONMENT

As specified in *Manually configuring your environment for Teamcenter utilities*.

FILES

As specified in *Log files produced by Teamcenter*.

RESTRICTIONS

None.

gmo_migrate_usage_nves

Migrates existing architecture breakdowns to the newer, simplified format of named variant expression (NVE) comprising usage, year, and production usage. You should only run this utility if you created NVEs in Teamcenter 2007.1 MP6 or earlier by running the **rdv_import_usage** utility. If you created NVEs in later releases, it is not necessary to run this utility. Otherwise, you should run this migration utility only once on each architecture breakdown.

The utility outputs the usage NVEs in the architecture breakdown and the assembly structure in the new format of NVEs. It also generates a list of the older format NVEs that are replaced and you can use this list to identify unused NVEs for deletion.

For example, it takes an older format NVE coded as `D9_2009-09_AA5M_1PD69_PS` and generate the following NVE code strings:

```
D9_AA5M_1PD69_000
Year_2009-09
ProductionUsage
```

SYNTAX

gmo_migrate_usage_nves [-u=user-id {-p=password | -pf=password-file} -g=group]
-revision_rule=rev-rule-string -mode=migration-mode
-top_level_AB_id=vehicle-architecture-id | -louholder_item_id=item-id
-archId_list_file=file-name
-h

ARGUMENTS**-u**

Specifies the user ID.

This is generally **infodba** or another user with administration privileges. If this argument is used without a value, the operating system user name is used.

Note

If Security Services single sign-on (SSO) is enabled for your server, the **-u** and **-p** arguments are authenticated externally through SSO rather than being authenticated against the Teamcenter database. If you do not supply these arguments, the utility attempts to join an existing SSO session. If no session is found, you are prompted to enter a user ID and password.

-p

Specifies the password.

If used without a value, the system assumes a null value.

If this argument is not used, the system assumes the *user-ID* value to be the password.

This argument is mutually exclusive with the **-pf** argument.

-pf

Specifies the password file. If used without a value, the system assumes a null value. If this argument is not used, the system assumes the *user-ID* value to be the password.

For more information about managing password files, see [Manage password files](#).

This argument is mutually exclusive with the **-p** argument.

-g

Specifies the group associated with the user.

If used without a value, the user's default group is assumed.

-revision_rule

Specifies a valid revision rule string to configure the structure window.

-mode

Specifies the running mode of the utility, either **report** to generate a report of NVEs to migrate or **migrate** to initiate migration of affected NVEs.

-top_level_AB_id

Specifies the item ID of a top level architecture. All LOUs under LOU holders in the specified architecture are processed. If you specify this argument, do not specify a **-louholder_item_id** argument.

-louholder_item_id

Specifies the item ID of a LOU holder. If you specify this argument, do not specify a **-top_level_AB_id** argument.

-archId_list_file

Specifies a flat file containing architecture IDs. The utility uses these architecture IDs to retrieve the NVEs of affected LOUs.

-h

Displays help for this utility.

ENVIRONMENT

As specified in [Manually configuring your environment for Teamcenter utilities](#).

RESTRICTIONS

None.

FILES

As specified in [Log files produced by Teamcenter](#).

EXAMPLES

To migrate all the NVEs for the **Model_2009_AB** architecture breakdown, enter the following command on a single line:

```
gmo_migrate_usage_nves  
-u=user-name -p=password -g=group -revision_rule=production -mode=migrate  
-top_level_AB_id=Model_2009_AB -archId_list_file=AB.txt
```

gmo_set_rel_status

Sets the release status for the item revisions and related datasets, revision masters, and BOMview revisions with the specified statuses. It searches for the specified folder in home folder of the user **infodba** and find all the items and item revisions in that folder. The utility then sets the specified *product-release-status* for all the non-**PDE** item revisions, datasets, BOMview revisions, revision master forms and the specified *pdi-release-status* for all the **PDI** item revisions, datasets, BOMview revisions, and revision master forms. It sets the given release status only if no release status is set before for item revision, dataset, bomview revision, and revision master form.

SYNTAX

gmo_set_rel_status [-u=*user-id* {-p=*password* | -pf=*password-file*} -g=*group*]
folder_name prod_rel_status pdi_rel_status

ARGUMENTS**-u**

Specifies the user ID.

This is generally **infodba** or another user with administration privileges. If this argument is used without a value, the operating system user name is used.

Note

If Security Services single sign-on (SSO) is enabled for your server, the **-u** and **-p** arguments are authenticated externally through SSO rather than being authenticated against the Teamcenter database. If you do not supply these arguments, the utility attempts to join an existing SSO session. If no session is found, you are prompted to enter a user ID and password.

-p

Specifies the password.

If used without a value, the system assumes a null value.

If this argument is not used, the system assumes the *user-ID* value to be the password.

This argument is mutually exclusive with the **-pf** argument.

-pf

Specifies the password file. If used without a value, the system assumes a null value. If this argument is not used, the system assumes the *user-ID* value to be the password.

For more information about managing password files, see [Manage password files](#).

This argument is mutually exclusive with the **-p** argument.

-g

Specifies the group associated with the user.

If used without a value, the user's default group is assumed.

-h

Displays help for this utility.

RESTRICTIONS

The *folder_name* must be present in the home folder of the **infodba** user.

EXAMPLES

None.

gmo_split_usage

Divides GM Corporate Dictionary (Architecture Breakdown Structure) and Line Of Usage Data (GPDS XML files) into predefined blocks of data (GPDS XML files). For each block of data, this utility creates script/batch file that is run to upload the Corporate Dictionary/Usage Data.

SYNTAX

```
gmo_split_usage [-u=user-id {-p=password | -pf=password-file} -g=group]
=input-file {[-log=name-of-logfile -max=maximum-usage |
[-enable_lock_grabbing] -arch=y
{-archtop_item_id=arch-top-item-id | -archtop_item_name=arch-item-name]]}
-import_usage_log=logfile-name -generate_delta_xml [
-enable_lock_grabbing] [-h]
```

ARGUMENTS

-u

Specifies the user ID.

This is generally **infodba** or another user with administration privileges. If this argument is used without a value, the operating system user name is used.

Note

If Security Services single sign-on (SSO) is enabled for your server, the **-u** and **-p** arguments are authenticated externally through SSO rather than being authenticated against the Teamcenter database. If you do not supply these arguments, the utility attempts to join an existing SSO session. If no session is found, you are prompted to enter a user ID and password.

-p

Specifies the password.

If used without a value, the system assumes a null value.

If this argument is not used, the system assumes the *user-ID* value to be the password.

-pf

Specifies the password file. If used without a value, the system assumes a null value. If this argument is not used, the system assumes the *user-ID* value to be the password.

For more information about managing password files, see [Manage password files](#).

This argument is mutually exclusive with the **-p** argument.

-g

Specifies the group associated with the user.

If used without a value, the user's default group is assumed.

-input

Specifies the name of the input file.

-log

Specifies the name of the log file.

-max

Specifies the maximum usage.

-arch

Boolean flag to upload Corporate Dictionary.

-archtop_item_id

Specifies item ID of the Architecture Breakdown.

-archtop_item_name

Specifies the name of the architecture top item.

-import_usage_log

Specifies for each block of data, a separate logfile is generated with increments indicating each data block.

-generate_delta_xml

Specifies the utility is to generate the Delta XML file.

-h

Displays help for this utility.

ENVIRONMENT

As specified in [Manually configuring your environment for Teamcenter utilities](#).

FILES

As specified in [Log files produced by Teamcenter](#).

RESTRICTIONS

None.

gmo_update_vas_data

Connects to the GPDS system and updates the VAS registration.

SYNTAX

gmo_update_vas_data [-u=*user-id* {-p=*password* | -pf=*password-file*} -g=*group*]
-datasource=*external-datasource-name* [-hostname=*external-datasource-hostname*]
[-user=*external-proxy-user*] [-passwd=*external-proxy-password*] [-h]

ARGUMENTS**-u**

Specifies the user ID.

This is generally **infodba** or another user with administration privileges. If this argument is used without a value, the operating system user name is used.

Note

If Security Services single sign-on (SSO) is enabled for your server, the **-u** and **-p** arguments are authenticated externally through SSO rather than being authenticated against the Teamcenter database. If you do not supply these arguments, the utility attempts to join an existing SSO session. If no session is found, you are prompted to enter a user ID and password.

-p

Specifies the password.

If used without a value, the system assumes a null value.

If this argument is not used, the system assumes the *user-ID* value to be the password.

-pf

Specifies the password file. If used without a value, the system assumes a null value. If this argument is not used, the system assumes the *user-ID* value to be the password.

For more information about managing password files, see [Manage password files](#).

This argument is mutually exclusive with the **-p** argument.

-g

Specifies the group associated with the user.

If used without a value, the user's default group is assumed.

-datasource

Specifies the name of the external datasource.

-hostname

Specifies the hostname of the external datasource.

-user

Specifies the user of the external datasource.

-passwd

Specifies the password of the external datasource.

-h

Displays help for this utility.

ENVIRONMENT

As specified in [*Manually configuring your environment for Teamcenter utilities.*](#)

FILES

As specified in [*Log files produced by Teamcenter.*](#)

RESTRICTIONS

None.

gmo_upgrade_dlist_objects

Migrates the distribution list objects created as a dataset using the TcAEV8.1/V9.1 functionality. It also searches for all distribution list objects existing in the database and transforms them into **EPMAssignmentList** objects.

**DESCRIPTION
SYNTAX**

gmo_upgrade_dlist_objects [-u=*user-id* {-p=*password* |
-pf=*password-file*} -g=*group*]
[-delete_old] [-h]

ARGUMENTS**-u**

Specifies the user ID.

This is generally **infodba** or another user with administration privileges. If this argument is used without a value, the operating system user name is used.

Note

If Security Services single sign-on (SSO) is enabled for your server, the **-u** and **-p** arguments are authenticated externally through SSO rather than being authenticated against the Teamcenter database. If you do not supply these arguments, the utility attempts to join an existing SSO session. If no session is found, you are prompted to enter a user ID and password.

-p

Specifies the password.

If used without a value, the system assumes a null value.

If this argument is not used, the system assumes the *user-ID* value to be the password.

-pf

Specifies the password file. If used without a value, the system assumes a null value. If this argument is not used, the system assumes the *user-ID* value to be the password.

For more information about managing password files, see [Manage password files](#).

This argument is mutually exclusive with the **-p** argument.

-g

Specifies the group associated with the user.

If used without a value, the user's default group is assumed.

-delete_old

Indicates that the utility is to delete the old assignment list objects after they are upgraded.

-h

Displays help for this utility.

ENVIRONMENT

As specified in [Manually configuring your environment for Teamcenter utilities](#).

FILES

As specified in *Log files produced by Teamcenter*.

RESTRICTIONS

None.

gmo_validate_xml

Validates the GPDS generated XML file. It ensures that all of the model option statements and model designators contain the correct VDS records and the corporate dictionary is correctly defined.

SYNTAX

gmo_validate_xml [-u=*user-id* {-p=*password* | -pf=*password-file*} -g=*group*]
-input=*input-xml-file-name* [-h]

ARGUMENTS**-u**

Specifies the user ID.

This is generally **infodba** or another user with administration privileges. If this argument is used without a value, the operating system user name is used.

Note

If Security Services single sign-on (SSO) is enabled for your server, the **-u** and **-p** arguments are authenticated externally through SSO rather than being authenticated against the Teamcenter database. If you do not supply these arguments, the utility attempts to join an existing SSO session. If no session is found, you are prompted to enter a user ID and password.

-p

Specifies the password.

If used without a value, the system assumes a null value.

If this argument is not used, the system assumes the *user-ID* value to be the password.

-pf

Specifies the password file. If used without a value, the system assumes a null value. If this argument is not used, the system assumes the *user-ID* value to be the password.

For more information about managing password files, see [Manage password files](#).

This argument is mutually exclusive with the **-p** argument.

-g

Specifies the group associated with the user.

If used without a value, the user's default group is assumed.

-input

Specifies the name of the XML file.

-h

Displays help for this utility.

ENVIRONMENT

As specified in [Manually configuring your environment for Teamcenter utilities](#).

FILES

As specified in [Log files produced by Teamcenter](#).

RESTRICTIONS

None.

gmo_vds_util

Synchronizes the variant data stored with the item revision between the base and multiple target item revisions.

SYNTAX

```
gmo_vds_util [-u=user-id {-p=password | -pf=password-file} -g=group]
[-i=item-id
| -key=[keyAttr1=keyVal1] [,keyAttr2=keyVal2]...[,keyAttrN=keyValN]
-b=base-item-rev [-t=target-item-rev1 ... ] [-h]
```

ARGUMENTS**-u**

Specifies the user ID.

This is generally **infodba** or another user with administration privileges. If this argument is used without a value, the operating system user name is used.

Note

If Security Services single sign-on (SSO) is enabled for your server, the **-u** and **-p** arguments are authenticated externally through SSO rather than being authenticated against the Teamcenter database. If you do not supply these arguments, the utility attempts to join an existing SSO session. If no session is found, you are prompted to enter a user ID and password.

-p

Specifies the password.

If used without a value, the system assumes a null value.

If this argument is not used, the system assumes the *user-ID* value to be the password.

-pf

Specifies the password file. If used without a value, the system assumes a null value. If this argument is not used, the system assumes the *user-ID* value to be the password.

For more information about managing password files, see [Manage password files](#).

This argument is mutually exclusive with the **-p** argument.

-g

Specifies the group associated with the user.

If used without a value, the user's default group is assumed.

-i

Specifies the item ID.

-key

Specifies the key of the item. Use the following format:

```
[keyAttr1=keyVal1] [,keyAttr2=keyVal2]...[,keyAttrN=keyValN]
```

To find the key of an object, use the **get_key_string** utility.

For more information, see the [Business Modeler IDE Guide](#).

-b

Specifies the base item revision.

-t

Specifies the first target item revision. You can specify this argument multiple times.

-h

Displays help for this utility.

rdv_import_usage

Imports usages from an XML file to Teamcenter usage representations. The utility applies all model NVEs available during the pre-usage stage to the top-level architecture node. The model NVEs are not applied to the children of the top-level architecture node (the architecture breakdown elements). However, the stored model NVEs on the top-level architecture element are available for manual application to any children in the architecture breakdown. When working in the Replace Design in Product wizard in DesignContext or Structure Manager, the user can select the model NVEs from the top-level architecture breakdown, rather than from the preselected architecture breakdown element (ABE).

Unlike in earlier versions of Teamcenter, model NVEs are not associated with the ABEs under the top-level architecture breakdown. Rather, model NVEs are stored as absolute occurrence data on the top-level architecture node. The utility creates the necessary absolute occurrence data at the top-level during the pre-usage import step.

Platform Designer allows the user to view model and manual NVEs associated with the top-level. You can use the **Add** command to add associated manual NVEs from the variant expression block at the top level to the absolute occurrence data.

Optionally, you can configure the utility to retry if it fails to complete a usage load for any reason on the first attempt. Failure to complete the load impacts downstream processes, as users cannot align their CAD solutions to the most recent PLM system changes. The utility retries obtaining a lock on the objects needing modification, typically, the product revision or top-level architecture node. Once a lock is obtained, the utility completes usage load operations. The optional ability for the utility to obtain the necessary locks is set with the **RDV_enable_product_lock** preference.

Caution Data loss may occur if the utility removes the lock while a user is actively editing data associated with the top-level product.

In the event of a failure while importing LOUs, it reports error codes and descriptions in the system log. These reports can be interpreted by users or scripts to take the necessary corrective action.

Note The format of the NVEs created by this utility changed with effect from Teamcenter 2007.1 MP6. Newer versions of the utility automatically include options and values so that you can use those options for the manual creation of NVEs and saved variant rules (SVRs) in Platform Designer. While running the utility, Teamcenter checks to see if the item ID of the product item is present in the **PSM_global_option_item_ids** preference. If so, it automatically creates the variability on the top level architecture breakdown. The import utility then applies variability for every option-value on the architecture breakdown (VAB). During execution of the **rdv_import_usage** utility, Teamcenter does not enforce hierarchical variability on the architecture breakdown (VAB).

If you used an earlier version of this utility to create NVEs, you should run the **gmo_migrate_usage_nves** utility to migrate them to the new format. If you created NVEs in Teamcenter 2007.1 MP6 or later, it is not necessary to migrate them. Examples of the old and new formats follow:

Old NVE format

D1_2008-UP_8619_2WC69_P
 D1_2007-07_8619_2WR69_P
 D1_2007-07_8619_2WP69_P

New NVE format

D1_8619_2WC69_000Year_2008-UPProductionUsage
 D1_8619_2WR69_000Year_2007-07ProductionUsage
 D1_8619_2WP69_000Year_2007-07ProductionUsage

A sequence number is appended to the authorized NVE so that true availability changes in model codes across years may be recorded in the NVE content. In the previous examples, the sequence number is **000**.

SYNTAX

rdv_import_usage [-u=*user-id* {-p=*password* | -pf=*password-file*} -g=*group*]
-input_file=*name-of-xml-file* **-h**

ARGUMENTS**-u**

Specifies the user ID.

This is generally **infodba** or another user with administration privileges. If this argument is used without a value, the operating system user name is used.

Note

If Security Services single sign-on (SSO) is enabled for your server, the **-u** and **-p** arguments are authenticated externally through SSO rather than being authenticated against the Teamcenter database. If you do not supply these arguments, the utility attempts to join an existing SSO session. If no session is found, you are prompted to enter a user ID and password.

-p

Specifies the password.

If used without a value, the system assumes a null value.

If this argument is not used, the system assumes the *user-ID* value to be the password.

-pf

Specifies the password file. If used without a value, the system assumes a null value. If this argument is not used, the system assumes the *user-ID* value to be the password.

For more information about managing password files, see [Manage password files](#).

This argument is mutually exclusive with the **-p** argument.

-g

Specifies the group associated with the user.

If used without a value, the user's default group is assumed.

-input_file

Specifies the XML input file from which the utility imports usages.

-h

Displays help for this utility.

ENVIRONMENT

- As specified in *Manually configuring your environment for Teamcenter utilities*.
- The **TC_retry_time** preference determines the time interval at which the utility tries to obtain a lock on objects to import, if it is not initially successful. You can set this preference as an environment variable so that the default value can be overridden by scripts during usage loading.
- The **TC_max_number_of_retries** preference determines how many attempts the utility makes to obtain a lock on objects to import, if it is not initially successful. You can set this preference as an environment variable so that the default value can be overridden by scripts during usage loading.

RESTRICTIONS

None.

FILES

As specified in *Log files produced by Teamcenter*.

EXAMPLES

To import usages, enter the following command on a single line:

```
rdv_import_usage -u=user-name -p=password -g=group -input_file=pdis101.x
```

The utility uses the **RDV_IMPORT_USAGE_TM** transfer mode to convert the XML file to PLM XML format, via a style sheet.

Chapter

17 Aerospace and Defense utilities

default_adsfoundation_queries

Installs the saved queries in the ADS Foundation template with names and descriptions either in the locale specified for the system or in all supported locales. The following queries are installed by this utility:

- **Find ADSTechDocument**
- **Find ADSDrawing**
- **Find ADSPart**
- **Find ADSDesign**

Note This utility runs automatically when the Aerospace and Defense solution is installed or upgraded.

SYNTAX

default_adsfoundation_queries [-u=*user-id* {-p=*password* | -pf=*password-file*} -g=*group*]-locales=*locale-code* | **ALL** [-h]

ARGUMENTS**-u**

Specifies the user ID.

This is generally **infodba** or another user with administration privileges. If this argument is used without a value, the operating system user name is used.

Note If Security Services single sign-on (SSO) is enabled for your server, the **-u** and **-p** arguments are authenticated externally through SSO rather than being authenticated against the Teamcenter database. If you do not supply these arguments, the utility attempts to join an existing SSO session. If no session is found, you are prompted to enter a user ID and password.

-p

Specifies the password.

If used without a value, the system assumes a null value.

If this argument is not used, the system assumes the *user-ID* value to be the password.

This argument is mutually exclusive with the **-pf** argument.

-pf

Specifies the password file. If used without a value, the system assumes a null value. If this argument is not used, the system assumes the *user-ID* value to be the password.

For more information about managing password files, see [Manage password files](#).

This argument is mutually exclusive with the **-p** argument.

-g

Specifies the group associated with the user.

If used without a value, the user's default group is assumed.

-locales

Specifies the locale, using locale codes or **ALL**, for which translated query names and descriptions are installed. You can specify a single locale, or you can specify multiple locales in a comma-separated list, for example, **en_US,de_DE,fr_FR**. Using the **ALL** value installs all locales supported by your Teamcenter system.

For a list of locale codes, see the [Localization Guide](#).

-h

Displays help for this utility.

ENVIRONMENT

As specified in [Manually configuring your environment for Teamcenter utilities](#).

FILES

As specified in [Log files produced by Teamcenter](#).

RESTRICTIONS

None.

EXAMPLES

- To install the ADS Foundation saved queries with names and descriptions in all locales supported for the system, enter the following command on a single line:

```
default_adsfoundation_queries -u=infodba -p=password -g=dba -locales=ALL
```

- To install the ADS Foundation saved queries with names and descriptions in English, German, and Czech, enter the following command on a single line:

```
default_adsfoundation_queries -u=infodba -p=password -g=dba  
-locales=en_US,de_DE,cs_CZ
```

default_adschangemanagement_queries

Installs the ADS Change Management saved queries with names and descriptions in either the locale specified for the system or in all supported locales. The following queries are installed by this utility:

- **Find All Change Notice Revisions**

Note This utility runs automatically when the Aerospace and Defense Change Management solution is installed or upgraded.

SYNTAX

```
default_adschangemanagement_queries [-u=user-id {-p=password |  
-pf=password-file} -g=group]  
-locales=locale-code | ALL [-h]
```

ARGUMENTS**-u**

Specifies the user ID.

This is generally **infodba** or another user with administration privileges. If this argument is used without a value, the operating system user name is used.

Note If Security Services single sign-on (SSO) is enabled for your server, the **-u** and **-p** arguments are authenticated externally through SSO rather than being authenticated against the Teamcenter database. If you do not supply these arguments, the utility attempts to join an existing SSO session. If no session is found, you are prompted to enter a user ID and password.

-p

Specifies the password.

If used without a value, the system assumes a null value.

If this argument is not used, the system assumes the *user-ID* value to be the password.

This argument is mutually exclusive with the **-pf** argument.

-pf

Specifies the password file. If used without a value, the system assumes a null value. If this argument is not used, the system assumes the *user-ID* value to be the password.

For more information about managing password files, see [Manage password files](#).

This argument is mutually exclusive with the **-p** argument.

-g

Specifies the group associated with the user.

If used without a value, the user's default group is assumed.

-locales

Specifies the locale, using locale codes or **ALL**, for which translated query names and descriptions are installed. You can specify a single locale or you can specify multiple

locales in a comma-separated list, for example **en_US,de_DE,fr_FR**. Using the **ALL** value installs all locales supported by your Teamcenter system.

For a list of locale codes, see the [Localization Guide](#).

-h

Displays help for this utility.

ENVIRONMENT

As specified in [Manually configuring your environment for Teamcenter utilities](#).

FILES

As specified in [Log files produced by Teamcenter](#).

RESTRICTIONS

None.

EXAMPLES

- To install the ADS Change Management saved queries with names and descriptions in all locales supported for the system, enter the following command on a single line:

```
default_adschangemanagement_queries -u=infodba -p=password  
-g=dba -locales=ALL
```

- To install the ADS Change Management saved queries with names and descriptions in English, German, and Czech, enter the following command on a single line:

```
default_adschangemanagement_queries -u=infodba -p=password -g=dba  
-locales=en_US,de_DE,cs_CZ
```

update_locationcode_from_owningorg

Updates the **Original Location Code** of items and **Current Location Code** of item revisions based on the **Organization ID** of the organization associated with the item or item revision.

This utility also creates a relation between groups and company locations based on the defined organization structure.

SYNTAX

```
update_locationcode_from_owningorg [-u=user-id {-p=password |  
-pf=password-file}  
-g=group] [-h]
```

ARGUMENTS**-u**

Specifies the user ID.

This is generally **infodba** or another user with administration privileges. If this argument is used without a value, the operating system user name is used.

Note

If Security Services single sign-on (SSO) is enabled for your server, the **-u** and **-p** arguments are authenticated externally through SSO rather than being authenticated against the Teamcenter database. If you do not supply these arguments, the utility attempts to join an existing SSO session. If no session is found, you are prompted to enter a user ID and password.

-p

Specifies the password.

If used without a value, the system assumes a null value.

If this argument is not used, the system assumes the *user-ID* value to be the password.

This argument is mutually exclusive with the **-pf** argument.

-pf

Specifies the password file. If used without a value, the system assumes a null value. If this argument is not used, the system assumes the *user-ID* value to be the password.

For more information about managing password files, see [Manage password files](#).

This argument is mutually exclusive with the **-p** argument.

-g

Specifies the group associated with the user.

If used without a value, the user's default group is assumed.

-h

Displays help for this utility.

ENVIRONMENT

As specified in [Manually configuring your environment for Teamcenter utilities](#).

FILES

As specified in *Log files produced by Teamcenter*.

RESTRICTIONS

None.

EXAMPLES

- To update the **Original Location Code** of all items and **Current Location Code** of all item revisions based on the **Organization ID** attribute, and create relations between groups and company locations:

```
update_locationcode_from_owningorg -u=infodba -p=password -g=dba
```

Chapter

*18 Materials Management and
Substance Compliance*

material_export

Exports the material and associated substances information in MatML format.

SYNTAX

material_export
[-u=*user-id*]
{-p=*password* | -pf=*password-file*}
[-g=*group*]
[-f=*output-xml-file*]
[-item=*item-id*]
[-rev=*item-revision-id*]
[-optionset=*name*]
[-transfermode=*name*]
[-h]

ARGUMENTS**-u**

Specifies the user ID.

This is generally **infodba** or another user with administration privileges. If this argument is used without a value, the operating system user name is used.

Note

If Security Services single sign-on (SSO) is enabled for your server, the **-u** and **-p** arguments are authenticated externally through SSO rather than being authenticated against the Teamcenter database. If you do not supply these arguments, the utility attempts to join an existing SSO session. If no session is found, you are prompted to enter a user ID and password.

-p

Specifies the password.

If used without a value, the system assumes a null value.

If this argument is not used, the system assumes the *user-ID* value to be the password.

This argument is mutually exclusive with the **-pf** argument.

-pf

Specifies the password file. If used without a value, the system assumes a null value. If this argument is not used, the system assumes the *user-ID* value to be the password.

For more information about managing password files, see [Manage password files](#).

This argument is mutually exclusive with the **-p** argument.

-g

Specifies the group associated with the user.

If used without a value, the user's default group is assumed.

-file

Specifies the MatML file that contains the information about the material objects that are exported.

-item

Specifies the item ID. This argument is mutually exclusive with the **-item_key** argument.

-item_key

Specifies the a string identifier containing attributes that identify the item object to export. This argument is mutually exclusive with the **-item** argument.

-rev

Specifies the item revision ID of the item that must be exported.

-optionset

Specifies the option set name of the transfer option set that must be used for this export.

-transfermode

Specifies the transfer mode name that is to be used during export.

-h

Displays help for this utility.

ENVIRONMENT

As specified in [Manually configuring your environment for Teamcenter utilities](#).

FILES

As specified in [Log files produced by Teamcenter](#).

RESTRICTIONS

None.

EXAMPLES

```
material_export -file=steel.xml -item=000016 -rev=A -xsl=tcxml_to_matml.xls  
optionset=TIEConfiguredExportDefault
```

material_import

Imports MatML file containing the materials and substances information into Teamcenter.

SYNTAX

```
material_import  
[-u=user-id]  
{-p=password | -pf=password-file}  
[-g=group]  
[-file=input-xml-file]  
[-dir=input-dir-path]  
[-optionset=name]  
[-transfermode=name]  
[-errorcontinue=yes/no]  
[-xls=xls-file]  
[-h]
```

ARGUMENTS**-u**

Specifies the user ID.

This is generally **infodba** or another user with administration privileges. If this argument is used without a value, the operating system user name is used.

Note

If Security Services single sign-on (SSO) is enabled for your server, the **-u** and **-p** arguments are authenticated externally through SSO rather than being authenticated against the Teamcenter database. If you do not supply these arguments, the utility attempts to join an existing SSO session. If no session is found, you are prompted to enter a user ID and password.

-p

Specifies the password.

If used without a value, the system assumes a null value.

If this argument is not used, the system assumes the *user-ID* value to be the password.

This argument is mutually exclusive with the **-pf** argument.

-pf

Specifies the password file. If used without a value, the system assumes a null value. If this argument is not used, the system assumes the *user-ID* value to be the password.

For more information about managing password files, see [Manage password files](#).

This argument is mutually exclusive with the **-p** argument.

-g

Specifies the group associated with the user.

If used without a value, the user's default group is assumed.

-file

Specifies the MatML file that contains the information about the material objects that must be imported.

-dir

Specifies the directory path of the MatML file.

-optionset

Specifies the name of the transfer option set that must be used for this import.

-transfermode

Specifies the transfer mode name that is to be used during import.

-errorcontinue

Specifies the option to control continuation of import when an error is encountered. The default value is no.

-xls

Specifies the XLS file that needs to be applied on input material file before the import is performed. If not provided, the default XLS file is located at **TC_DATA** is used.

-h

Displays help for this utility.

ENVIRONMENT

As specified in [Manually configuring your environment for Teamcenter utilities](#).

FILES

As specified in [Log files produced by Teamcenter](#).

RESTRICTIONS

None.

EXAMPLES

```
material_import -dir=d:\matML -xls=mattotcxml.xls material_import
-file=mat1.xml -optionset=MaterialImportDefaultOptionSet
-optionset=MaterialImportDefaultOptionSet

material_import -file=mat1.xml -dir=d:\matML -site=100001
-optionset=MaterialImportDefaultOptionSet
```

subscmpl_import_template

Imports the default user templates to work with Microsoft Office components that are required in Substance Compliance.

SYNTAX**material_import**

[-u=user-id
{-p=password | -pf=password-file}
[-g=group]
[-f=path of the folder]
[-i=path of the template file]
[-t=type of template being imported]
[-h]

ARGUMENTS**-u**

Specifies the user ID.

This is generally **infodba** or another user with administration privileges. If this argument is used without a value, the operating system user name is used.

Note

If Security Services single sign-on (SSO) is enabled for your server, the **-u** and **-p** arguments are authenticated externally through SSO rather than being authenticated against the Teamcenter database. If you do not supply these arguments, the utility attempts to join an existing SSO session. If no session is found, you are prompted to enter a user ID and password.

-p

Specifies the password.

If used without a value, the system assumes a null value.

If this argument is not used, the system assumes the *user-ID* value to be the password.

This argument is mutually exclusive with the **-pf** argument.

-pf

Specifies the password file. If used without a value, the system assumes a null value. If this argument is not used, the system assumes the *user-ID* value to be the password.

For more information about managing password files, see [Manage password files](#).

This argument is mutually exclusive with the **-p** argument.

-g

Specifies the group associated with the user.

If used without a value, the user's default group is assumed.

-f

Specifies the path of the folder from which templates are imported.

-i

Specifies the path of the template file to be imported.

-t

Specifies the type of the template being imported using **-I** or **-f** option, for example, **ExcelTemplate**.

-h

Displays help for this utility.

ENVIRONMENT

As specified in *Manually configuring your environment for Teamcenter utilities*.

FILES

As specified in *Log files produced by Teamcenter*.

RESTRICTIONS

None.

subscmpl_msd_import

Imports the material substance declaration data available in the IPC 1752 XML files into Teamcenter.

SYNTAX

```
subscmpl_msd_import  
-u=user-ID  
{-p=password | -pf=password-file}  
[-g=group]  
{-file=input-xml-file | -dir=directory_path}  
[-optionset=TransferOptionSet_name]  
[-transfermode=TransferMode_name]  
[-xsl=XML_style_sheet_xsl_file]  
[-h]
```

ARGUMENTS**-u**

Specifies the user ID. The user must have administrative privileges.

If this argument is used without a value, the operating system user name is used.

Note If your Teamcenter server uses Security Services single sign-on, see [Before you begin](#) for additional information.

-p

Specifies the user's password.

If used without a value, the system assumes a null value. If this argument is not used, the system assumes the *user-ID* value to be the password.

This argument is mutually exclusive with the **-pf** argument.

-pf

Specifies the password file. If used without a value, the system assumes a null value. If this argument is not used, the system assumes the *user-ID* value to be the password.

For more information about managing password files, see [Manage password files](#).

This argument is mutually exclusive with the **-p** argument.

-g

Specifies the group associated with the user.

-file

Specifies the material substance declaration data in XML file to be related to the vendor part.

Note This argument is mutually exclusive with the **-dir** argument.

-dir

Specifies the directory containing material substance declaration data in the XML file to be related to the vendor parts. The system searches for the matching XML file name as specified by the **SUBSCMPL_msd_file_naming_prefix** preference.

The **unique_file_name.xml** file is the input XML file containing material substance declaration data. The **unique_file_name.zip** file contains supporting documents for the same vendor part.

Note This argument is mutually exclusive with the **-file** argument.

-optionset

Specifies the name of the transfer option set to be used for the Teamcenter XML import.

Note This argument is optional.

By default, **TIEImportOptionSetDefault** is used as the transfer option set.

-transfermode

Specifies the name of the transfer mode to be used for the Teamcenter XML import.

Note This argument is optional.

By default, **TIEImportDefault** is used as the transfer mode.

-xsl

XML style sheet file to be applied to the input material substance declaration XML file before the Teamcenter XML import.

Note This argument is optional.

By default, the XML style sheet **msd_to_tcxml.xsl** file from the **TC_DATA** directory is used to convert the input XML file to the Teamcenter XML file format. For import, this file must specify the correct master site ID.

-h

Displays help for this utility.

ENVIRONMENT

As specified in [Manually configuring your environment for Teamcenter utilities](#).

FILES

As specified in [Log files produced by Teamcenter](#).

RESTRICTIONS

None.

EXAMPLES

To import material substance declaration data, enter one of the following commands on a single line:

```
subscmpl_msd_import
-u=user
-p=password
-g=group
-file=D:\IPC_Data\IPC_1.xml

subscmpl_msd_import
-u=user
-p=password
-g=group
-dir=/data/IPC_Data

subscmpl_msd_import
```

```
-u=user
-p=password
-g=group
-dir=/data/IPC_Data
-xsl=/data/Stylesheet/IPC_new.xsl

subscmpl_msd_import
-u=user
-p=password
-g=group
-dir=/data/IPC_Data
-xsl=/data/Stylesheet/IPC_new.xsl
-optionset=TIEImportOptionSetDefault

subscmpl_msd_import
-u=user
-p=password
-g=group
-dir=/data/IPC_Data
-xsl=/data/Stylesheet/IPC_new.xsl
-optionset=TIEImportOptionSetDefault
-transfermode=TIEImportDefault
```

subscmpl_validate_compliance_results

Validates the compliance results in Teamcenter based on the expiry dates.

SYNTAX**material_import**

[-u=user-id]
{-p=password | -pf=password-file}
[-g=group]
[-locales=language codes]
[-h]

ARGUMENTS**-u**

Specifies the user ID.

This is generally **infodba** or another user with administration privileges. If this argument is used without a value, the operating system user name is used.

Note

If Security Services single sign-on (SSO) is enabled for your server, the **-u** and **-p** arguments are authenticated externally through SSO rather than being authenticated against the Teamcenter database. If you do not supply these arguments, the utility attempts to join an existing SSO session. If no session is found, you are prompted to enter a user ID and password.

-p

Specifies the password.

If used without a value, the system assumes a null value.

If this argument is not used, the system assumes the *user-ID* value to be the password.

This argument is mutually exclusive with the **-pf** argument.

-pf

Specifies the password file. If used without a value, the system assumes a null value. If this argument is not used, the system assumes the *user-ID* value to be the password.

For more information about managing password files, see [Manage password files](#).

This argument is mutually exclusive with the **-p** argument.

-g

Specifies the group associated with the user.

If used without a value, the user's default group is assumed.

-locales

Specifies the language codes separated by a comma.

-h

Displays help for this utility.

ENVIRONMENT

As specified in [Manually configuring your environment for Teamcenter utilities](#).

FILES

As specified in *Log files produced by Teamcenter*.

RESTRICTIONS

None.

EXAMPLES

Translations for query name and descriptions will be added for the specified languages. User can specify single or multiple language codes separated with comma (.). Specifying ALL will add translations in all supported languages.

```
[-locales=language codes/ALL]
```

Chapter

*19 Consumer Packaged Goods
utilities*

add_specmgr_templates

Installs Specification Manager templates in Teamcenter. These templates are used to view information about a specification and its section in the Specification Manager application in the rich client.

The following templates are installed by this utility:

- **SPECMGR_default_spec_template**
- **SPECMGR_default_object_template**
- **SPECMGR_default_CPSpecSection_object_template**
- **SPECMGR_default_CPFreeText_object_template**
- **SPECMGR_default_CPRefText_object_template**
- **SPECMGR_default_CPPropertyList_object_template**
- **SPECMGR_default_CPComponentList_object_template**
- **SPECMGR_default_CPBOM_object_template**
- **SPECMGR_default_CPRefSpec_object_template**
- **SPECMGR_default_CPProcLine_object_template**
- **SPECMGR_default_CPRefObject_object_template**
- **SPECMGR_default_CPProcStage_object_template**

Note This utility runs automatically when you install or upgrade Specification Manager.

SYNTAX

add_specmgr_templates [-u=*user-id* {-p=*password* | -pf=*password-file*} -g=*group*] [-h]

ARGUMENTS**-u**

Specifies the user ID.

This is generally **infodba** or another user with administration privileges. If this argument is used without a value, the operating system user name is used.

Note If Security Services single sign-on (SSO) is enabled for your server, the **-u** and **-p** arguments are authenticated externally through SSO rather than being authenticated against the Teamcenter database. If you do not supply these arguments, the utility attempts to join an existing SSO session. If no session is found, you are prompted to enter a user ID and password.

-p

Specifies the password.

If used without a value, the system assumes a null value.

If this argument is not used, the system assumes the *user-ID* value to be the password.

This argument is mutually exclusive with the **-pf** argument.

-pf

Specifies the password file. If used without a value, the system assumes a null value. If this argument is not used, the system assumes the *user-ID* value to be the password.

For more information about managing password files, see [Manage password files](#).

This argument is mutually exclusive with the **-p** argument.

-g

Specifies the group associated with the user.

If used without a value, the user's default group is assumed.

-h

Displays help for this utility.

ENVIRONMENT

As specified in [Manually configuring your environment for Teamcenter utilities](#).

FILES

As specified in [Log files produced by Teamcenter](#).

RESTRICTIONS

None.

EXAMPLES

- To install Specification Manager templates:

```
add_specmgr_templates -u=infodba -p=password -g=dba
```

Chapter

*20 Computer-aided engineering
(CAE) utilities*

cae_migrate_atl_preferences

Migrates legacy tool configuration settings to a new configuration managed by the Teamcenter vaulted dataset.

SYNTAX

```
cae_migrate_atl_preferences  
[-u=user-id  
{-p=password  
-pf=password-file}  
-g=group]  
[-file=file-name (including file path)]  
[-overwrite]  
[-h]
```

ARGUMENTS**-u**

Specifies the user ID.

This is generally **infodba** or another user with administration privileges. If this argument is used without a value, the operating system user name is used.

Note

If Security Services single sign-on (SSO) is enabled for your server, the **-u** and **-p** arguments are authenticated externally through SSO rather than being authenticated against the Teamcenter database. If you do not supply these arguments, the utility attempts to join an existing SSO session. If no session is found, you are prompted to enter a user ID and password.

-p

Specifies the password.

If used without a value, the system assumes a null value.

If this argument is not used, the system assumes the *user-ID* value to be the password.

This argument is mutually exclusive with the **-pf** argument.

-pf

Specifies the password file. If used without a value, the system assumes a null value. If this argument is not used, the system assumes the *user-ID* value to be the password.

For more information about managing password files, see [Manage password files](#).

This argument is mutually exclusive with the **-p** argument.

-g

Specifies the group associated with the user.

If used without a value, the user's default group is assumed.

-file

Specifies the path and file name of the tool preferences file containing tool preferences data to be migrated/append to dataset-managed preferences.

If you do not specify this option, the utility looks for authoring tool launch preferences (used in Teamcenter 2007.1.x) in the database and simulation tool configuration (used in Teamcenter 8.0.x) in the **espf_configuration.xml** file in the **TC_DATA** folder and migrates them (if they are available) to a new configuration managed by the Teamcenter vaulted dataset.

Note You must specify the file name and the complete file path for the **-file** option.

-overwrite

Overwrites the existing tool preferences with legacy tool preferences.

If you do not specify this option, the utility appends legacy tool preferences to the existing tool preferences.

-h

Displays help for this utility.

ENVIRONMENT

As specified in [Manually configuring your environment for Teamcenter utilities](#).

FILES

As specified in [Log files produced by Teamcenter](#).

RESTRICTIONS

None.

EXAMPLES

- Migrate the Teamcenter 2007.1.x preferences to the Teamcenter 8.1 configuration file in a dataset.

Note This example assumes there is no **espf_configuration.xml** file in the active **TC_DATA** folder and no **CAESolution** dataset with the name indicated by **CAE_simulation_tool_config_dsname** exists in the database.

```
cae_migrate_atl_preferences
-u=infodba
-p=*****
-g=dba
```

The system reads the configuration definitions from the **CAE_pre_processor_***, **CAE_solver_***, and **CAE_post_processor_*** preferences and creates an appropriate **espf_configuration.xml** file. The system creates a new **CAESolution** dataset named according to the value in **CAE_simulation_tool_config_dsname**, and imports the file as an **XML-File** reference.

- Migrate the Teamcenter 8.0.x **espf_configuration.xml** file to the Teamcenter 8.1 configuration file in a dataset.

Note This example assumes a **simadmin** user is assigned to a **Simulation-Administrator** role in the **Simulation-Administration** group. This example assumes that there is an **espf_configuration.xml** file in the active **TC_DATA** folder, but no **CAESolution** dataset with the name indicated by **CAE_simulation_tool_config_dsname** exists in the database.

```
cae_migrate_atl_preferences
-u=simadmin
```

```
-p=*****  
-g=Simulation-Administration
```

The system locates the **espf_configuration.xml** file in the active **TC_DATA** folder. The system creates a new **CAESolution** dataset named according to the value in **CAE_simulation_tool_config_dsname** and imports the file as an **XML-File** reference.

- Migrate the specified Teamcenter 8.0.x **espf_configuration.xml** file to the Teamcenter 8.1 configuration file in a dataset.

Note This example assumes that there exists a **simadmin** user assigned to a **Simulation-Administrator** role in the **Simulation-Administration** group.

```
cae_migrate_atl_preferences  
-u=simadmin  
-p=*****  
-g=Simulation-Administration  
-file=D:\MyFiles\espf_configuration.xml  
-overwrite
```

The system locates the **espf_configuration.xml** file using the provided path. The system attempts to locate a **CAESolution** dataset named according to the value in **CAE_simulation_tool_config_dsname**. If an appropriate dataset is found, the system imports the file as an XML file reference, overwriting any existing reference. If an appropriate dataset is not found, the system creates a new **CAESolution** dataset named according to the value in **CAE_simulation_tool_config_dsname** and imports the provided file as an **XML-File** reference.

cae_save_result_data

Saves the output of an analysis run (results) to Teamcenter when called by an analysis application.

SYNTAX

```
cae_save_result_data
[-u=user-id]
{-p=password
-pf=password-file}
-g=group]
-name=result-name
[-type=type-name]
[-desc=result-description]
{[-item=item-id] | [-key=[keyAttr1=keyVal1][,keyAttr2=keyVal2]...[,keyAttrN=keyValN]]}
-rev=revision-id
-xml_file=xml-file-name
-result_dir=result-directory
[-external=true | false]
[-overwrite=true | false]
[-h]
```

ARGUMENTS

-u

Specifies the user ID.

This is generally **infodba** or another user with administration privileges. If this argument is used without a value, the operating system user name is used.

Note

If Security Services single sign-on (SSO) is enabled for your server, the **-u** and **-p** arguments are authenticated externally through SSO rather than being authenticated against the Teamcenter database. If you do not supply these arguments, the utility attempts to join an existing SSO session. If no session is found, you are prompted to enter a user ID and password.

-p

Specifies the password.

If used without a value, the system assumes a null value.

If this argument is not used, the system assumes the *user-ID* value to be the password.

This argument is mutually exclusive with the **-pf** argument.

-pf

Specifies the password file. If used without a value, the system assumes a null value. If this argument is not used, the system assumes the *user-ID* value to be the password.

For more information about managing password files, see [Manage password files](#).

This argument is mutually exclusive with the **-p** argument.

-g

Specifies the group associated with the user.

If used without a value, the user's default group is assumed.

-name

Specifies the name of the result object to be created in Teamcenter. This argument is required, and the name must be no longer than 32 characters in length. In addition, there must be no other result inside the results dataset that resides in the specified item revision with the same name.

-type

Specifies the value used for the type attribute of the new result in Teamcenter. This argument is optional, and if supplied it must be no longer than 32 characters long.

-desc

Specifies the value to be used for the description attribute of the new result in Teamcenter. This argument is optional, and if supplied it must be no longer than 240 characters long.

-item

Specifies the item ID of the item which contains the item revision containing the results dataset in which the result is created. This argument is mutually exclusive with the **-key** argument; one of these two arguments must be specified.

-key

Specifies the key of the item which contains the item revision containing the results dataset in which the result is created. This argument is mutually exclusive with the **-item** argument; one of these two arguments must be specified. Use the following format:

```
[keyAttr1=keyVal1] [,keyAttr2=keyVal2]...[,keyAttrN=keyValN]
```

To find the key of an object, use the [get_key_string](#) utility.

For more information, see the [Business Modeler IDE Guide](#).

-rev

Specifies the ID of the item revision containing the results dataset in which the result will be created. If a results dataset does not already exist in this item revision one is created to hold the new result. This argument is required.

-xml_file

Specifies the full path to the PLM XML metadata file used to generate the result. This argument is optional.

-result_dir

Specifies a path to a directory that is assumed to contain the data files for the result. All files found in this directory are associated with the result. This argument is optional.

-external

Indicates whether the files associated with the result are stored externally to the Teamcenter volume. Valid values for this argument are **true** and **false** and are not case sensitive. This argument is optional; if not provided the default value is false.

-overwrite

Indicates whether a pre-existing result with the same name should be overwritten. Valid values for this argument are **true** and **false** and are not case sensitive. This argument is optional. If no value is given, the default value is **false**. If the value of

this argument is **false** and a result with the input name already exists, the system returns an error.

-h

Displays help for this utility.

ENVIRONMENT

As specified in [Manually configuring your environment for Teamcenter utilities](#).

FILES

As specified in [Log files produced by Teamcenter](#).

RESTRICTIONS

- This utility must be called only from an integrated CAE analysis application.
- The type of the specified item must be the type defined as the CAE default analysis item type.

EXAMPLES

- To save a new result named **Result1** into a results dataset under **item 000001**, **revision A**, enter the following command on a single line:

```
cae_save_result_data
-name=Result1
-type=Analysis
-desc=Test first run
-item=000001
-rev=A
-xml_file=c:\temp\test.xml
-result_dir=c:\temp\result1
```

- To overwrite the existing result from the previous example with a new result of the same name, this time with externally stored files, enter the following command on a single line:

```
cae_save_result_data
-name=Result1
-type=Analysis
-desc=Overwrite first run
-item=000001
-rev=A
-xml_file=c:\temp\test2.xml
-result_dir=c:\temp\result2
-external=true
-overwrite=true
```

epm_import_batch_meshing_results

Imports batch meshing results into the Teamcenter database.

SYNTAX

```
epm_import_batch_meshing_results  
[-u=user-id  
{-p=password  
-pf=password-file}  
-g=group]  
-workdir=working-directory  
{[-itemid=item-id] | [-key=[keyAttr1=keyVal1][keyAttr2=keyVal2]...[keyAttrN=keyValN]]}  
-revid=revision-id  
-dsname=dataset-name  
-nrname=named-reference-name  
-size=mesh-size  
-ext=extension  
[-h]
```

ARGUMENTS**-u**

Specifies the user ID.

This is generally **infodba** or another user with administration privileges. If this argument is used without a value, the operating system user name is used.

Note

If Security Services single sign-on (SSO) is enabled for your server, the **-u** and **-p** arguments are authenticated externally through SSO rather than being authenticated against the Teamcenter database. If you do not supply these arguments, the utility attempts to join an existing SSO session. If no session is found, you are prompted to enter a user ID and password.

-p

Specifies the password.

If used without a value, the system assumes a null value.

If this argument is not used, the system assumes the *user-ID* value to be the password.

This argument is mutually exclusive with the **-pf** argument.

-pf

Specifies the password file. If used without a value, the system assumes a null value. If this argument is not used, the system assumes the *user-ID* value to be the password.

For more information about managing password files, see [Manage password files](#).

This argument is mutually exclusive with the **-p** argument.

-g

Specifies the group associated with the user.

If used without a value, the user's default group is assumed.

-workdir

Specifies the full operating system path of the working directory into which all batch meshing results for a single job will be written.

After meshing completes, the batch meshing interface examines this directory to determine which mesh results files are imported in to the Teamcenter database. This argument is mandatory.

-itemid

Identifies the item under which the file will be imported.

The utility imports the file indicated by the **-file** argument value in to the **CAEMesh** dataset at the location specified by the **-itemid**, **-revid**, and **-dsname** arguments. This argument is mutually exclusive with the **-key** argument; one of the two arguments must be specified.

-key

Uses the key to identify the item under which the file will be imported.

The utility imports the file indicated by the **-file** argument value in to the **CAEMesh** dataset at the location specified by the **-key**, **-revid**, and **-dsname** arguments. This argument is mutually exclusive with the **-itemid** argument; one of the two arguments must be specified. Use the following format:

```
[keyAttr1=keyVal1] [,keyAttr2=keyVal2]...[,keyAttrN=keyValN]
```

To find the key of an object, use the [get_key_string](#) utility.

For more information, see the [Business Modeler IDE Guide](#).

-revid

Identifies the item revision under which the file will be imported.

The utility imports the file indicated by the **-file** argument value in to the **CAEMesh** dataset at the location specified by the **-itemid**, **-revid**, and **-dsname** arguments. This argument and value are mandatory.

-dsname

Specifies the name to be applied to the resulting **CAEMesh** dataset.

The utility imports the file indicated by the **-file** argument value in to the **CAEMesh** dataset at the location specified by the **-itemid**, **-revid**, and **-dsname** arguments. This argument and value are mandatory.

-nrname

Specifies the base name used when generating the resulting named reference in the **CAEMesh** dataset.

This name is used as input to the **USER_get_batch_meshing_nr_name()** user exit to determine the actual named reference file name to be imported. This argument and value are mandatory.

-size

Specifies the mesh size used when generating the mesh.

The mesh size is used as input to the **USER_get_batch_meshing_nr_name()** user exit to determine the actual named reference file name. The mesh size is encoded in the named reference file name to distinguish those of different mesh sized in the same dataset. This argument and value are mandatory.

-ext

Specifies the file name extension to apply to the named reference in the **CAEMesh** dataset.

The batch meshing interface uses this file name extension to find the batch meshing results files to import in to the resulting **CAEMesh** dataset in the Teamcenter database. This argument and value are mandatory.

-h

Displays help for this utility.

ENVIRONMENT

As specified in [Manually configuring your environment for Teamcenter utilities](#).

FILES

As specified in [Log files produced by Teamcenter](#).

RESTRICTIONS

This utility is intended to be called only from the batch meshing interface from within Teamcenter.

EXAMPLES

To import the **some_mesh.bdf** file in to a **CAEMesh** dataset with the name **Some_part**, under item **000001**, revision **A**, enter the following command on a single line:

```
epm_import_batch_meshing_results
-workdir=c:\temp\batch_meshing_dir
-itemid=000001
-revid=A
-dsname=Some_part
-nrname=some_mesh
-size=10
-ext=dbf
```

epm_notify_batch_meshing_results

Notifies the user of the results of a batch meshing job.

SYNTAX

epm_notify_batch_meshing_results

[-u=user-id
{-p=password
-pf=password-file}
-g=group]
-workdir=working-directory
-logfile=log-file-name
[-h]

ARGUMENTS

-u

Specifies the user ID.

This is generally **infodba** or another user with administration privileges. If this argument is used without a value, the operating system user name is used.

Note

If Security Services single sign-on (SSO) is enabled for your server, the **-u** and **-p** arguments are authenticated externally through SSO rather than being authenticated against the Teamcenter database. If you do not supply these arguments, the utility attempts to join an existing SSO session. If no session is found, you are prompted to enter a user ID and password.

-p

Specifies the password.

If used without a value, the system assumes a null value.

If this argument is not used, the system assumes the *user-ID* value to be the password.

This argument is mutually exclusive with the **-pf** argument.

-pf

Specifies the password file. If used without a value, the system assumes a null value. If this argument is not used, the system assumes the *user-ID* value to be the password.

For more information about managing password files, see [Manage password files](#).

This argument is mutually exclusive with the **-p** argument.

-g

Specifies the group associated with the user.

If used without a value, the user's default group is assumed.

-workdir

Specifies the full operating system path to the working directory in which all batch meshing results for a single job reside.

User notification includes a reference to this directory in the event that the user must examine the contents of the directory. This argument and value are mandatory.

-logfile

Specifies the full operating system path to the log file containing specific information about the batch meshing job for which this notification is generated.

The utility examines the contents of this log file name as input for generating the user notification message. This argument and value are mandatory.

-h

Displays help for this utility.

ENVIRONMENT

As specified in [Manually configuring your environment for Teamcenter utilities](#).

FILES

As specified in [Log files produced by Teamcenter](#).

RESTRICTIONS

This utility is intended to be called only from the batch meshing interface from within Teamcenter.

EXAMPLES

To send the results of a batch meshing job to the current user's Teamcenter mailbox (using autologon), enter the following command on a single line:

```
epm_notify_batch_meshing_results  
-workdir=c:\temp\batch_meshing_dir  
-logfile=c:\temp\batch_meshing_dir\batch_meshing_log
```

cae_execute_datamap

Applies data mapping rules to an input structure by providing the root item revision of the input structure and the configuration information of the structure along with the domain for the data mapping rules.

When a snapshot is produced, the **Snapshot** folder, rather than the root item of the resulting structure, is pasted on the invoking user's **Newstuff** folder.

SYNTAX

cae_execute_datamap

[-u=cae_analyst

{-p=password

-pf=password-file}

-g=CAE_designer]

-inputItemKey=key-to-the-root-item-of-the-input-structure

-revID=revision-ID-of-the-input-item-revision

-snapshotOutput=snapshot-folder

[-h]

ARGUMENTS

-u

Specifies the user ID.

This is generally **infodba** or another user with administration privileges. If this argument is used without a value, the operating system user name is used.

Note

If Security Services single sign-on (SSO) is enabled for your server, the **-u** and **-p** arguments are authenticated externally through SSO rather than being authenticated against the Teamcenter database. If you do not supply these arguments, the utility attempts to join an existing SSO session. If no session is found, you are prompted to enter a user ID and password.

-p

Specifies the password.

If used without a value, the system assumes a null value.

If this argument is not used, the system assumes the *user-ID* value to be the password.

This argument is mutually exclusive with the **-pf** argument.

-pf

Specifies the password file. If used without a value, the system assumes a null value. If this argument is not used, the system assumes the *user-ID* value to be the password.

For more information about managing password files, see [Manage password files](#).

This argument is mutually exclusive with the **-p** argument.

-g

Specifies the group associated with the user.

If used without a value, the user's default group is assumed.

-inputItemKey

Specifies the key to the root item of the input structure; for example, **000001**.

-revID

Specifies the revision ID of the input item revision.

-snapshot

Specifies the snapshot folder name.

-h

Displays help for this utility.

ENVIRONMENT

As specified in [Manually configuring your environment for Teamcenter utilities](#).

FILES

As specified in [Log files produced by Teamcenter](#).

RESTRICTIONS

None.

EXAMPLES

To execute a data map on a input structure, enter the following command on a single line:

Example 1:

```
cae_execute_datamap
-u=cae_analyst
-p=password
-g=CAE_designer
-inputItemKey=000001 (key-to-the-root-item-of-the-input-structure)
-revID=A (revision-ID-of-the-input-item-revision)
-snapshotOutput=snapshot-folder
```

Example 2:

```
cae_execute_datamap
-u=cae_analyst
-g=CAE_designer
-inputItemKey=000001 (key-to-the-root-item-of-the-input-structure)
-revID=A (revision-ID-of-the-input-item-revision)
```

cae_execute_structuremap

Applies structure map rules to an input structure by providing the root item revision of the input structure and the structure map item revision containing the rule along with the configuration information of the structure.

When a snapshot is produced, the **Snapshot** folder, rather than the root item of the resulting structure, is pasted on the invoking user's **Newstuff** folder.

SYNTAX

```
cae_execute_structuremap
[-u=user-id
{-p=password
-pf=password-file}
-g=group]
-inputItemkey= key-of-the-root-item
-revID= revision-ID-of-the-input-item-revision
-SMItemKey= structure-map-item-key
-SMRev= structure-map-item-revision-ID
-snapshotOutput= snapshot-folder
[-h]
```

ARGUMENTS

-u

Specifies the user ID.

This is generally **infodba** or another user with administration privileges. If this argument is used without a value, the operating system user name is used.

Note

If Security Services single sign-on (SSO) is enabled for your server, the **-u** and **-p** arguments are authenticated externally through SSO rather than being authenticated against the Teamcenter database. If you do not supply these arguments, the utility attempts to join an existing SSO session. If no session is found, you are prompted to enter a user ID and password.

-p

Specifies the password.

If used without a value, the system assumes a null value.

If this argument is not used, the system assumes the *user-ID* value to be the password.

This argument is mutually exclusive with the **-pf** argument.

-pf

Specifies the password file. If used without a value, the system assumes a null value. If this argument is not used, the system assumes the *user-ID* value to be the password.

For more information about managing password files, see [Manage password files](#).

This argument is mutually exclusive with the **-p** argument.

-g

Specifies the group associated with the user.

If used without a value, the user's default group is assumed.

-inputItemkey

Specifies the key of the root item of the input structure; for example, **000001**.

-revID

Specifies the revision ID of the input item revision.

-SMItemKey

Specifies the structure map item key; for example, **000002**.

-SMRev

Specifies the structure map item revision ID.

-snapshotOutput

Specifies the output snapshot name of the output model item revision.

-h

Displays help for this utility.

ENVIRONMENT

As specified in [Manually configuring your environment for Teamcenter utilities](#).

FILES

As specified in [Log files produced by Teamcenter](#).

RESTRICTIONS

None.

EXAMPLES

To execute structure map rules on a input structure, enter the following command on a single line:

Example 1:

```
cae_execute_structuremap
-u=cae_analyst
-g=CAE_designer
-inputItemKey=000001 (key-to-the-root-item-of-the-input-structure)
-revID=A (revision-ID-of-the-input-item-revision)
-SMItemKey=000002 (structure-map-item-key)
-SMRev=B (structure-map-item-revision-ID)
-snapshotOutput=snapshot-folder
```

Example 2:

```
cae_execute_structuremap
-u=cae_analyst
-g=CAE_designer
-inputItemKey=000001 (key-to-the-root-item-of-the-input-structure)
-revID=A (revision-ID-of-the-input-item-revision)
-SMItemKey=000002 (structure-map-item-key)
-SMRev=B (structure-map-item-revision-ID)
```

cae_migrate_datamap_definition

Migrates the data mapping definition files from the *TC_DATA* directory in the existing installation to a dataset attached to an item revision in the database. The data map definition files consist of the data mapping XML file—named according to the [CAE_dataMapping_file](#) preference—and an optional **NodeXMLConfig.xml** configuration file.

This utility checks the value of the [CAE_datamap_files_location](#) preference.

- If the preference is set, the utility imports the files located in the *TC_DATA* directory to a **CAEStructureMap** dataset attached to the item revision indicated by the preference and creates a dataset, if needed. The files are *not* removed from the *TC_DATA* directory.
- If the preference is set, and the indicated item revision already has an attached **CAEStructureMap** dataset with a valid XML reference, the utility does not do anything, that is, it does not overwrite any existing data map definition files in the database.
- If the preference is not set, the utility creates an **ItemRevision** type and assigns ownership according to the **-ou** and **-og** arguments supplied to the utility, and sets the preference value to the created item revision.

SYNTAX

cae_migrate_datamap_definition

```
[-u=user-id  
{-p=password  
-pf=password-file}  
-g=group]  
-ou=owning-user-id  
-og=owning-group-name  
[-h]
```

ARGUMENTS

-u

Specifies the user ID.

This is generally **infodba** or another user with administration privileges. If this argument is used without a value, the operating system user name is used.

Note

If Security Services single sign-on (SSO) is enabled for your server, the **-u** and **-p** arguments are authenticated externally through SSO rather than being authenticated against the Teamcenter database. If you do not supply these arguments, the utility attempts to join an existing SSO session. If no session is found, you are prompted to enter a user ID and password.

-p

Specifies the password.

If used without a value, the system assumes a null value.

If this argument is not used, the system assumes the *user-ID* value to be the password.

This argument is mutually exclusive with the **-pf** argument.

-pf

Specifies the password file. If used without a value, the system assumes a null value. If this argument is not used, the system assumes the *user-ID* value to be the password.

For more information about managing password files, see [Manage password files](#).

This argument is mutually exclusive with the **-p** argument.

-g

Specifies the group associated with the user.

If used without a value, the user's default group is assumed.

-ou

(Optional) Specifies the owning user ID.

This user owns all created objects.

-og

(Optional) Specifies the group associated with the owning user.

Note If you specify the **-ou** argument and do not specify the **-og** argument, the user's default group is used.

-h

Displays help for this utility.

ENVIRONMENT

As specified in [Manually configuring your environment for Teamcenter utilities](#).

FILES

As specified in [Log files produced by Teamcenter](#).

RESTRICTIONS

None.

EXAMPLES

To migrate the data mapping definition files from the operating system to a dataset, enter the following command in a single line:

```
cae_migrate_datamap_definition
-u=infodba
-p=infodba
-g=dba
-ou=cae_admin
-og=simulation-administrator
```

cae_validate_structuremap

Validates the **StructureMap** item revision to ensure that the configured structure map rules are properly validated before they are executed.

SYNTAX

```
cae_validate_structuremap
[-u=infodba
{-p=infodba
-pf=password-file}
-g=dba]
-ou=cae_admin
-og=simulation_administrator
-or=simulation_administrator
-SMItemKeyList=item-id=0001;item-id=0002;item-id=0003
-SMRevList =A;A;C
[-h]
```

ARGUMENTS

-u

Specifies the user ID.

This is generally **infodba** or another user with administration privileges. If this argument is used without a value, the operating system user name is used.

Note

If Security Services single sign-on (SSO) is enabled for your server, the **-u** and **-p** arguments are authenticated externally through SSO rather than being authenticated against the Teamcenter database. If you do not supply these arguments, the utility attempts to join an existing SSO session. If no session is found, you are prompted to enter a user ID and password.

-p

Specifies the password.

If used without a value, the system assumes a null value.

If this argument is not used, the system assumes the *user-ID* value to be the password.

This argument is mutually exclusive with the **-pf** argument.

-pf

Specifies the password file. If used without a value, the system assumes a null value. If this argument is not used, the system assumes the *user-ID* value to be the password.

For more information about managing password files, see [Manage password files](#).

This argument is mutually exclusive with the **-p** argument.

-g

Specifies the group associated with the user.

If used without a value, the user's default group is assumed.

-ou

(Optional) Specifies the owning user ID.

This user owns all created objects.

-og

(Optional) Specifies the group associated with the owning user.

Note If you specify the **-ou** argument and do not specify the **-og** argument, the user's default group is used.

-or

(Optional) Specifies the role associated with the owning user.

-SMItemKeyList

Specifies the list of structure map items key. At least one **SMItemKey** is required, for example, **-SMItemKey=item-id=value**.

When using multiple item keys, use a semicolon separated string, for example, **SMItemKeyList=item1key;item2key;item3key**.

-SMRevList

Specifies the list of structure map item revision IDs for corresponding item keys.

Note This list must have the same number of entries as the **SMItemKeyList** list.

When using multiple item keys, use a semicolon separated string, for example, **SMRevList=item1-revision-id;item2-revision-id;item3-revision-id**.

-h

Displays help for this utility.

ENVIRONMENT

As specified in [Manually configuring your environment for Teamcenter utilities](#).

FILES

As specified in [Log files produced by Teamcenter](#).

RESTRICTIONS

None.

EXAMPLES

To validate the structure map, enter the following command in a single line:

```
cae_validate_structuremap
-u=CAE_analyst
-p=*****
-g=CAE_designer
-ou=cae_admin
-og=simulation_administrator
-or=simulation_administrator
SMItemKeyList=item-id=0001;item-id=0002;item-id=0003
-SMRevList=A;A;C
```

cae_execute_cae_accountability_check

Compares the attributes of a product and model structure using the accountability check framework along with data mapping rules. The attributes are compared by applying data mapping rules. The attributes marked as **mapped** in the **NodeXMLConfig.xml** file used by data mapping is considered for comparison. Data mapping rules are applied on the input product structure and the data mapped values are compared with the model attribute value. The product and model item keys, and the revision IDs are required along with the data mapping domain as input. You can optionally specify the configuration information for the product and the model. The result of the comparison is created as a named reference containing a dataset (Microsoft Excel file) with the name you specify in the **-comparison_dataset_name** argument. The result is attached to the root of the model structure.

SYNTAX

```
cae_execute_cae_accountability_check
[-u=infodba
{-p=infodba
-pf=password-file}
-g=dba]
-ou=cae_admin
-og=simulation_administrator
-or=simulation_administrator
-product_item_key=key-to-the-root-item-of-the-product-structure
-product_rev_id=revision-ID-of-the-product-item-revision
-product_rev_rule=revision-rule-of-the-product-structure
-product_snapshot=snapshot-folder-name-for-product-structure
-product_variant_rule=variant-rule-for-the-product-structure
-model_item_key=key-to-the-root-item-of-the-model-structure
-model_rev_id=revision-ID-of-the-model-item-revision
-model_rev_rule=revision-rule-of-the-model-structure
-model_snapshot=snapshot-folder-name-for-the-model-structure
-model_variant_rule=variant-rule-of-the-model-structure
-domain=data-mapping-domain-you-want-to-use
-comparison_dataset_name=dataset-name-to-save-the-comparison-log-if-
the-mode-is-compare
-param_file=parameter-file-in-text-format-containing-parameters-for-the-utility
[-h]
```

ARGUMENTS

-u

Specifies the user ID.

Note

If Security Services single sign-on (SSO) is enabled for your server, the **-u** and **-p** arguments are authenticated externally through SSO rather than being authenticated against the Teamcenter database. If you do not supply these arguments, the utility attempts to join an existing SSO session. If no session is found, you are prompted to enter a user ID and password.

-p

Specifies the password.

If used without a value, the system assumes a null value.

If this argument is not used, the system assumes the *user-ID* value to be the password.

This argument is mutually exclusive with the **-pf** argument.

-pf

Specifies the password file. If used without a value, the system assumes a null value. If this argument is not used, the system assumes the *user-ID* value to be the password.

For more information about managing password files, see [Manage password files](#).

This argument is mutually exclusive with the **-p** argument.

-g

Specifies the group associated with the user.

If used without a value, the user's default group is assumed.

-ou

(Optional) Specifies the owning user ID.

This user owns all created objects.

-og

(Optional) Specifies the group associated with the owning user.

Note If you specify the **-ou** argument and do not specify the **-og** argument, the user's default group is used.

-or

(Optional) Specifies the role associated with the owning user.

-product_item_key

Specifies the key to the root item of the product structure.

-product_rev_id

Specifies the revision ID of the product item revision.

-product_rev_rule

(Optional) Specifies the revision rule of the product structure.

This argument is not considered if you specify the **-product_snapshot** argument.

-product_snapshot

(Optional) Specifies the snapshot folder name for product structure.

-product_variant_rule

(Optional) Specifies the variant rule for the product structure.

-model_item_key

Specifies the key to the root item of the model structure.

-model_rev_id

Specifies the revision ID of the model item revision.

-model_rev_rule

(Optional) Specifies the revision rule of the model structure.

This argument is not considered if you specify the **-model_snapshot** argument.

-model_snapshot

(Optional) Specifies the snapshot folder name for the model structure.

-model_variant_rule

(Optional) Specifies the variant rule of the model structure.

-domain

Specifies the data mapping domain you want to use.

-comparison_dataset_name

Specifies the dataset name to save the comparison log if the mode is compare.

If the dataset name already exists in the model revision, the dataset content is replaced.

-param_file

A parameter file in text format containing parameters for the utility.

-h

Displays help for this utility.

ENVIRONMENT

As specified in [Manually configuring your environment for Teamcenter utilities](#).

FILES

As specified in [Log files produced by Teamcenter](#).

RESTRICTIONS

None.

EXAMPLE 1

Using item key and rev ID, enter the following command in a single line:

```
cae_execute_cae_accountability_check
-u=adminjones
-p=adminjones
-g=admin
-product_item_key=item_id=0001
-product_rev_id=A
-product_rev_rule=myRevRule
-product_variant_rule=myVariantRule
-model_item_key=item_id=0002
-model_rev_id=A
-model_rev_rule=myRevRule
-model_variant_rule=myVariantRule
-domain=CAE
```

EXAMPLE 2

Using product and model snapshot, enter the following command in a single line:

```
cae_execute_cae_accountability_check
-u=adminjones
-p=adminjones
-g=admin
-product_snapshot=prodSnapshot
-product_variant_rule=myVariantRule
-model_snapshot=ModelSnapshot
-model_variant_rule=myVariantRule
-domain=CAE
```

EXAMPLE 3

Using a parameter file, enter the following command in a single line:

```
cae_execute_cae_accountability_check  
-u=adminjones  
-p=adminjones  
-g=admin  
-param_file=C:\temp\myparamfile.txt
```

Parameter file contents:

```
product_item_key=item_id=0001  
-product_rev_id=A  
-product_rev_rule=myRevRule  
-product_variant_rule=myVariantRule  
-model_item_key=item_id=0002  
-model_rev_id=A  
-model_rev_rule=myRevRule  
-model_variant_rule=myVariantRule  
-domain=CAE  
-comparsion_dataset_name=ComparisonReport
```

Chapter

*21 Teamcenter mechatronics process
management utilities*

install_algebraicformulas

Provides capabilities to create algebraic formula definitions (identical, linear, quadratic, and rational) in Teamcenter.

If algebraic formulas provided in Teamcenter are deleted, you can run this utility to reinstall them.

SYNTAX

```
install_algebraicformulas [-u=user-id {-p=password |  
-pf=password-file} -g=group]  
[-h]
```

ARGUMENTS**-u**

Specifies the user ID.

This is generally **infodba** or another user with administration privileges. If this argument is used without a value, the operating system user name is used.

Note If Security Services single sign-on (SSO) is enabled for your server, the user and password arguments are authenticated externally through SSO rather than being authenticated against the Teamcenter database. If you do not supply these arguments, the utility attempts to join an existing SSO session. If no session is found, you are prompted to enter a user ID and password.

-p

Specifies the password.

If used without a value, the system assumes a null value.

If this argument is not used, the system assumes the *user-ID* value to be the password.

This argument is mutually exclusive with the **-pf** argument.

-pf

Specifies the password file. If used without a value, the system assumes a null value. If this argument is not used, the system assumes the *user-ID* value to be the password.

For more information about managing password files, see [Manage password files](#).

This argument is mutually exclusive with the **-p** argument.

-g

Specifies the group associated with the user.

If used without a value, the user's default group is assumed.

-h

Displays help for this utility.

ENVIRONMENT

As specified in [Manually configuring your environment for Teamcenter utilities](#).

FILES

As specified in [Log files produced by Teamcenter](#).

RESTRICTIONS

None.

EXAMPLES

- To *create out of the box algebraic formula definitions in Teamcenter*, enter the following command on a single line:

```
install_algebraicformulas -u=infodba -p=password -g=dba
```

install_kbl

Extends the schema to provide Teamcenter support of wire harnesses meeting the KBL standard.

Note This utility installs all KBL types. If any of these types already exist in the system, it is skipped and a warning is displayed in the console. The message also gets printed in the system log file.

SYNTAX

install_kbl [-u=infodba {-p=infodba | -pf=password-file} -g=dba] -h

ARGUMENTS**-u**

Specifies the user ID.

This is generally **infodba** or another user with administration privileges. If this argument is used without a value, the operating system user name is used.

Note If Security Services single sign-on (SSO) is enabled for your server, the **-u** and **-p** arguments are authenticated externally through SSO rather than being authenticated against the Teamcenter database. If you do not supply these arguments, the utility attempts to join an existing SSO session. If no session is found, you are prompted to enter a user ID and password.

-p

Specifies the password.

If used without a value, the system assumes a null value.

If this argument is not used, the system assumes the *user-ID* value to be the password.

-pf

Specifies the password file. If used without a value, the system assumes a null value. If this argument is not used, the system assumes the *user-ID* value to be the password.

For more information about managing password files, see [Manage password files](#).

This argument is mutually exclusive with the **-p** argument.

-g

Specifies the group associated with the user.

If used without a value, the user's default group is assumed.

-h

Displays help for this utility.

ENVIRONMENT

As specified in [Manually configuring your environment for Teamcenter utilities](#).

FILES

As specified in [Log files produced by Teamcenter](#).

RESTRICTIONS

None.

update_gde_types

Allows site administrators to update the parent types of existing GDE types based on information provided in an input file.

SYNTAX

```
update_gde_types [-u=user-id {-p=password | -pf=password-file} -g=group]
[-s=parent-type -t=type1,type2] | -f=input-file -h
```

ARGUMENTS

-u

Specifies the user ID.

This is generally **infodba** or another user with administration privileges. If this argument is used without a value, the operating system user name is used.

Note

If Security Services single sign-on (SSO) is enabled for your server, the **-u** and **-p** arguments are authenticated externally through SSO rather than being authenticated against the Teamcenter database. If you do not supply these arguments, the utility attempts to join an existing SSO session. If no session is found, you are prompted to enter a user ID and password.

-p

Specifies the password.

If used without a value, the system assumes a null value.

If this argument is not used, the system assumes the *user-ID* value to be the password.

This argument is mutually exclusive with the **-pf** argument.

-pf

Specifies the password file. If used without a value, the system assumes a null value. If this argument is not used, the system assumes the *user-ID* value to be the password.

For more information about managing password files, see [Manage password files](#).

This argument is mutually exclusive with the **-p** argument.

-g

Specifies the group associated with the user.

If used without a value, the user's default group is assumed.

-f

Specifies the input file containing one or more lines with parent type and child type GDE information in the following format:

```
parent_type child_type_name1,child_type_name2,child_type_name3,...
```

If the input file is specified, it takes precedence over information provided by the **-s** and **-t** arguments.

-s

Specifies the parent type to be set for the GDE types.

-t

Specifies the GDE types, separated by commas, of the parent type to be updated.

-h

Displays help for this utility.

ENVIRONMENT

As specified in [Manually configuring your environment for Teamcenter utilities](#).

FILES

As specified in [Log files produced by Teamcenter](#).

RESTRICTIONS

None.

EXAMPLES

- Enter the following command on a single line to update specific children of a GDE type:

```
update_gde_types -u=user-name -p=password -g=dba  
-s=InterfaceDefinition -t=port1,port2
```

- Enter the following command on a single line to update GDE types based on an input file:

```
update_gde_types -u=user-name -p=password -g=dba -f=test.txt
```

The following is an example of format of the input file:

```
#Parent child1,child2  
InterfaceDefinition port1,port2,  
ProcessVariable pv1,pv2,
```

migrate_eda_data

Allows you to bulk migrate pre-Teamcenter 8.1 EDA data to the current data model.

Run this utility to *manually* perform a bulk EDA migration. This utility runs automatically when you upgrade from a pre-Teamcenter 8.1 database to a more recent data model and select **Teamcenter EDA Server Support** as part of the upgrade.

Note The migration process assumes the data does not contain variants.

SYNTAX

```
migrate_eda_data [-u=user-id {-p=password | -pf=password-file} -g=group]
[-dryrun] [-ccaSelectFile=path-name] [-migrationList=path-name]
[-logFile=path-name] [-h]
```

ARGUMENTS

-u

Specifies the user ID.

This is generally **infodba** or another user with administration privileges. If this argument is used without a value, the operating system user name is used.

Note If Security Services single sign-on (SSO) is enabled for your server, the **-u** and **-p** arguments are authenticated externally through SSO rather than being authenticated against the Teamcenter database. If you do not supply these arguments, the utility attempts to join an existing SSO session. If no session is found, you are prompted to enter a user ID and password.

-p

Specifies the password.

If used without a value, the system assumes a null value.

If this argument is not used, the system assumes the *user-ID* value to be the password.

-pf

Specifies the password file. If used without a value, the system assumes a null value. If this argument is not used, the system assumes the *user-ID* value to be the password.

For more information about managing password files, see [Manage password files](#).

This argument is mutually exclusive with the **-p** argument.

-g

Specifies the group associated with the user.

If used without a value, the user's default group is assumed.

-dryrun

Performs a dry run of the migration, making no changes to the database. Use this argument to identify problematic data prior to migration.

-ccaSelectFile

Specifies the input file containing the list of items to select when a schematic is identified as being related to multiple CCA item objects. The utility uses this information to migrate EDA schematic data to one of x number of CCA item objects related to it.

The file format is:

```
Schematic-Item-ID, CCA-Item-ID
Schematic-Item-ID2, CCA-Item-ID2
...
```

Note Inconsistencies in the file are logged as errors. Erroneous entries are skipped during migration.

-migrationList

Specifies the input file containing the list of items to migrate. If this argument is not provided, all **EDASchem** item objects are migrated.

The file format is:

```
Schematic-Item-ID
Schematic-Item-ID2
Schematic-Item-ID3
...
```

Note Inconsistencies in the file are logged as errors. Possible errors include, but are not limited to: item IDs of non-**EDASchem** item objects, duplicate entries, and invalid item IDs. Erroneous entries are skipped during migration.

-logFile

Specifies the location of the migration log file. If this argument is not specified, the default location is *TEMP\program-name_date-time.log*.

-h

Displays help for this utility.

ENVIRONMENT

As specified in [Manually configuring your environment for Teamcenter utilities](#).

FILES

As specified in [Log files produced by Teamcenter](#).

The log file is generated in the user's **TEMP** directory. The file name format is **migrate_eda_data-date-time.log**.

RESTRICTIONS

None.

EXAMPLES

To manually perform a bulk migrate of EDA data from a pre-Teamcenter 8.1 database to a more recent data model:

```
migrate_eda_data -u=infodba -p=infodba -g=dba
```

To perform a dry run of the EDA data bulk migration:

```
migrate_eda_data -u=infodba -p=infodba -g=dba -dryrun
```

To perform a dry run of the EDA data bulk migration using the **selectionFile.txt** CCA selection file:


```
migrate_eda_data -dryrun -u=infodba -p=infodba -g=dba  
-ccaSelectFile=D:\migration\selectionFile.txt
```

To perform a dry run of the EDA data bulk migration using the **selectionFile.txt** CCA selection file and the **edaSchemList.txt** migration list:

```
migrate_eda_data -u=infodba -p=infodba -g=dba -dryrun  
-ccaSelectFile=D:\migration\selectionFile.txt  
-migrationList=D:\migration\edaSchemList.txt
```

To bulk migrate the EDA data specified in the **selectionFile.txt** migration list:

```
migrate_eda_data -u=infodba -p=infodba -g=dba  
-ccaSelectFile=D:\migration\selectionFile.txt  
-migrationList=D:\migration\edaSchemList.txt
```

To bulk migrate the EDA data specified in the **selectionFile.txt** CCA selection file:

```
migrate_eda_data -u=infodba -p=infodba -g=dba  
-ccaSelectFile=D:\migration\selectionFile.txt
```

To bulk migrate the EDA data specified in the **selectionFile.txt** CCA selection file, with the migration information sent to the **migration_output.log** log file:

```
migrate_eda_data -u=infodba -p=infodba -g=dba  
-ccaSelectFile=D:\migration\selectionFile.txt  
-logFile=D:\migration\migration_output.log
```

Chapter

*22 Volume and database
management utilities*

collect_garbage

Collects unreferenced workspace objects and places them in a **WASTE BASKET** folder in the **Home** folder of the **infodba** user. Datasets, envelopes, folders, items, and forms objects can be collected. Released objects are not collected. A special case operation is the orphan option that collects item revisions that do not have valid parent items. These items revisions may be referenced in other folders, in which case you are warned while collection is taking place. Orphan operations should not be combined with other operations.

The **collect_garbage** utility should be run in two phases. The first phase is run without the **-delete** option, which allows objects to be collected in the **WASTE BASKET** folder. This allows you to examine the contents of the waste basket. Once you are satisfied with the contents, the **collect_garbage** can be rerun with the **-delete** option to empty the **WASTE BASKET** folder.

When working with large databases, use the **-query** argument to create a report of all unreferenced objects. You can restrict the report to specific object types with various arguments. When working with a large report, you can split the report into separate files that can be executed in batches. Use the **-rf** and **-if** arguments to define the file to which you want to write the report. The batch jobs can then be executed simultaneously on multiple workstations. Use the **-delete** argument to delete the unreferenced objects from the specified folder.

SYNTAX

```
collect_garbage [-u=user-id {-p=password | -pf=password-file} -g=group]
-rf=report-file-name -if=input-file-name
[-dataset] [-item] [-occurrence] [-absocdataqualifier]
[-form] [-folder] [-envelope] [-all]
[-orphan] [-delete] [-report] [-query]
[-dataset] [-child_references] [-ignore_relation]
[-gidentity] [-plmappuid=ReportOnly | ReportAndDelete]
[-start=number] [-end=number] [-h]
```

Caution You must run Teamcenter Workspace with system administration privileges to access the **WASTE BASKET** folder.

ARGUMENTS**-u**

Specifies the user ID.

This is generally **infodba** or another user with administration privileges. If this argument is used without a value, the operating system user name is used.

Note

If Security Services single sign-on (SSO) is enabled for your server, the **-u** and **-p** arguments are authenticated externally through SSO rather than being authenticated against the Teamcenter database. If you do not supply these arguments, the utility attempts to join an existing SSO session. If no session is found, you are prompted to enter a user ID and password.

-p

Specifies the password.

If used without a value, the system assumes a null value.

If this argument is not used, the system assumes the *user-ID* value to be the password.

This argument is mutually exclusive with the **-pf** argument.

-pf

Specifies the password file. If used without a value, the system assumes a null value. If this argument is not used, the system assumes the *user-ID* value to be the password.

For more information about managing password files, see [Manage password files](#).

This argument is mutually exclusive with the **-p** argument.

-g

Specifies the group associated with the user.

If used without a value, the user's default group is assumed.

-dataset

Specifies that datasets be collected or deleted.

-item

Specifies that items be collected or deleted.

Note This argument also collects all item revisions associated with the item.

-occurrence

Specifies that occurrences (appearance path nodes, absolute occurrences, and occurrence threads) are collected.

-absoccddataqualifier

Collects or deletes locally owned unreferenced, and locally owned referenced, absolute occurrence data qualifier (**AbsOccDataQualifier**) objects associated with replica BOM view revisions (BVRs).

-form

Specifies that forms be collected or deleted.

-folder

Specifies that object folders be collected or deleted.

-envelope

Specifies that envelopes be collected or deleted.

-all

Collects or deletes datasets, items, forms, object folders, and envelopes. This argument does not include orphans.

Siemens PLM Software does not recommend using this argument when processing a large database.

-orphan

Collects or deletes all item revisions that do not have a valid parent item. These item revisions may be referenced in other folders, in which case you are warned while collection is taking place. Use the **-orphan** argument alone to collect orphan item revisions in the **WASTE BASKET** folder. Use the **-delete** argument in conjunction

with the **-orphan** argument to delete orphans from the **WASTE BASKET** folder. Orphan operations should not be combined with other operations. See restriction #2.

Note Because orphan operations collect or delete item revisions that do not have valid parent items, it is normal for the **collect_garbage** application log file to include some errors. In most cases, you can disregard them.

-delete

Deletes all objects of a specified type. One or more of the **-dataset**, **-item**, **-form**, **-folder**, or **-envelope** arguments or the **-all** or **-orphan** arguments must be supplied. You can also include the **-gsidentity** option with the **-delete** argument.

-query

Queries the database for the instances of the specified object type when used in combination with a defined object type.

Note This argument works only in combination with an object type argument (**-item**, **-dataset**, **-form**, **-envelope**, **-folder**, **-all**) and the **-rf** argument.

-report

Creates a report of the objects moved to the **WASTE BASKET** folder of the **infodba** user.

Note The **-report** argument generates output in a different format than the **-orphan** operations because orphan objects are not valid workspace objects.

-gsidentity

Locates and removes all invalid **GSIdentity** records for objects that do not exist.

-plmappuid

Deletes unreferenced entries from the **plmappuid** table.

When objects are imported into Teamcenter using PLM XML, an **Application Ref** tag can be used to define individual IDs for the imported objects. The IDs are stored in the **plmappuid** table. If these objects are later deleted in Teamcenter, the entries are not deleted from the table. Over time the table size increases, decreasing performance.

Use the **ReportOnly** value to generate a count of all the unreferenced entries in the **plmappuid** table. Use the **ReportAndDelete** value to generate a count of all the unreferenced entries in the table and delete them.

-rf= report-file-name

Creates a report file listing instances of a specified type. This argument may be used in combination with object type arguments (**-item**, **-dataset**, **-form**, **-envelope**, **-folder**, **-orphan**) and the **-query** argument.

When used in combination with object type arguments, the **-rf** argument retrieves a list of all instances of a specified class and writes the list to a specified file.

Note This argument works only in combination with an object type argument (**-item**, **-dataset**, **-form**, **-envelope**, **-folder**, **-all**) and the **-if** argument.

A file name is required when using this argument. If a file name is not provided, the list is written to a default file named *argument-name_report.txt*, where argument

name is equal to **item**, **dataset**, **form**, **folder**, **envelope**, or **orphan**. If the default file already exists in the directory where this utility is executed, instances are overwritten to the default file.

When the report is large, Siemens PLM Software recommends that it be split into multiple reports. The suggested naming convention is *argument-name_report_aa.txt*, *argument-name_report_ab.txt*, and so on.

-if= *input-file-name*

Uses the report file name as input to identify the unreferenced objects of a given object type. Unreferenced objects are placed in a specified subfolder within the **Waste Basket** folder.

Note This argument works only in combination with an object type argument (**-item**, **-dataset**, **-form**, **-envelope**, **-folder**, **-all**) and the **-if** argument. If no value is specified for this argument, the utility exits with a message.

-start

Specifies the starting number of objects to process. The default value is **1**. Use this option in conjunction with the **-end** option.

The **-start** and **-end** arguments are recommended when the utility runs out of memory when loading too many objects of the given class for processing.

-end

Specifies the ending number of objects to process. Use this option in conjunction with the **-start** option.

-dataset

Specifies that datasets qualify as garbage for collection.

-child_references

Specifies that datasets qualify as garbage if they are unreferenced in the system but have secondary objects. Use with the **-dataset** option.

-ignore_relation

Specifies that datasets are excluded from garbage collection if they have at least one secondary object attached with any of the relations in the list. The valid value for this argument is a comma-separated list containing internal relation names. Use with the **-dataset** option.

-h

Displays help for this utility.

ENVIRONMENT

As specified in [Manually configuring your environment for Teamcenter utilities](#).

FILES

As specified in [Log files produced by Teamcenter](#).

RESTRICTIONS

1. The **collect_garbage** utility must be run from the **infodba** user account. This automatically enables the bypass feature and collects and deletes all garbage objects regardless of owning user and group.
2. Do not use the **-orphan** argument with an object type argument **-dataset**, **-item**, **-form**, **-folder**, **-envelope** or **-all**.

EXAMPLES

The following examples illustrate how to use the **-query** argument with this utility:

- The following example displays a message and exits the program because no file name was provided for the **-rf** argument:

```
collect_garbage -u=infodba -p=infodba -g=dba -item -rf -query
```

- The following example collects a list of unreferenced objects of the type **item** and writes the report to the **list_items.txt** file:

```
collect_garbage -u=infodba -p=infodba -g=dba -item -rf=list_items.txt  
-query
```

- The following example collects a list of unreferenced objects of type **item** and writes the report to the **item_report.txt** file:

```
collect_garbage -u=infodba -p=infodba -g=dba -item -query
```

- To collect unreferenced folders, enter the following command on a single line:

```
collect_garbage -folder
```

- To collect unreferenced folders and items, enter the following command on a single line:

```
collect_garbage -folder -item
```

- To collect unreferenced folders and items and get a report to **stdout**, enter the following command on a single line:

```
collect_garbage -folder -item -report
```

- To collect all unreferenced objects except orphans, enter the following command on a single line:

```
collect_garbage -all
```

- To delete folders collected in the **WASTE BASKET** folder, enter the following command on a single line:

```
collect_garbage -folder -delete
```

- To delete all objects collected in the **WASTE BASKET** folder except orphans, enter the following command on a single line:

```
collect_garbage -all -delete
```

- To collect all item revisions with no parent item, enter the following command on a single line:

```
collect_garbage -orphan
```

- To delete orphan item revisions in the **WASTE BASKET**, enter the following command on a single line:

```
collect_garbage -orphan -delete
```

- To collect unreferenced items into the **item_rep** file, enter the following command on a single line:

```
collect_garbage -item -query -rf=item_rep
```


- To process these items and insert into folder, enter the following command on a single line:

```
collect_garbage -item -report -if=item_rep
```

- To delete items in the **SUB WASTE BASKET** folder, enter the following command on a single line:

```
collect_garbage -item -delete -sub_folder=WBITEM_item_rep
```

- To collect unreferenced forms when there are too many forms in the database to load in memory, enter the following command on a single line:

```
collect_garbage -form -end=1000
collect_garbage -form -delete
collect_garbage -form -end=1000
collect_garbage -form -delete
```

Or

```
collect_garbage -form -start=1 -end=1000
collect_garbage -form -start=1001 -end=2000
collect_garbage -form -delete
```

- To delete unreferenced occurrence threads, appearance path nodes, and absolute occurrences, enter the following command on a single line:

```
collect_garbage -occurrence -delete
```

- To collect all locally owned referenced, and locally owned unreferenced **AabsOccDataQualifier** objects and place them in the respective **BadAbsOccFolder** and **BadAbsOccFolder_unrefdirectory** folders:

```
collect_garbage -absocccdataqualifier -query -sub_folder=BadAbsOccFolder
```

- To delete all locally owned unreferenced **AbsOccDataQualifier** objects:

```
collect_garbage -absocccdataqualifier -delete
```

- To delete all locally owned referenced **AbsOccDataQualifier** objects:

```
collect_garbage -absocccdataqualifier -delete -sub_folder=BadAbsOccFolder
```

When working with a large database, or with a large number of instances in a report file, Siemens PLM Software recommends that you split the query report into multiple files. The following examples illustrate how to split a report into specified files:

- The following example splits 50,000 instances reported from a query into files of 5,000 lines each:

```
split -l 5000 item_rep.txt ITEM_rep_
```

- The following example processes each instance from the report and identifies unreferenced objects. These objects are placed in a subfolder of the **WASTE BASKET** folder:

```
collect_garbage -u=infodba -p=infodba -g=dba -item -if=list_items.txt
```

- The following example processes each instance from the report and identifies unreferenced objects. These objects are placed in a subfolder of the **WASTE BASKET** folder. It also displays the object information, such as object name, ID, object type and owner's name.

```
collect_garbage -u=infodba -p=infodba -g=dba -item -if=list_items.txt
-report
```

- The following example retrieves all unreferenced objects of type **item** and places them in the **WASTE BASKET** folder:

```
collect_garbage -u=infodba -p=infodba -g=dba -item
```

- The following example retrieves all unreferenced objects of type **item** and places them in the **WASTE BASKET** folder. It also displays object information, such as object name, ID, object type, and owner's name.

```
collect_garbage -u=infodba -p=infodba -g=dba -item -report
```

The following examples illustrate how to use the **-delete** argument to delete unreferenced objects:

- The following example deletes all objects in the **WBITEM_item_rep.txt** subfolder within the **WASTE BASKET** folder.

```
collect_garbage -item -delete -sub_folder=WBITEM_item_rep.txt
```

- The following example deletes all unreferenced instances of type **item** in the **WASTE BASKET** folder, as well as all instances from any subfolders with names beginning with **WBITEM_**:

```
collect_garbage -u=infodba -p=infodba -g=dba -item -delete
```

- The following example displays a message and exits the program because no subfolder value is specified:

```
collect_garbage -u=infodba -p=infodba -g=dba -item -sub_folder=
```

- The following example deletes all objects of type **folder**, but does not delete the contents of the folder object. (The same concept is true for other object types, such as item, dataset, form, and envelope.)

```
collect_garbage -u=infodba -p=infodba -g=dba -folder -delete
```

- The following example deletes all **GSIdentity** records from the database:

```
collect_garbage -u=infodba -p=infodba -g=dba -gsidentity
-delete -rf=report-file-name
```

The following examples illustrate how to use the **-rf** argument:

- The following example displays a message and exits the program because no file name is provided for the **-rf** argument:

```
collect_garbage -u=infodba -p=infodba -g=dba -item -rf -query
```

- The following example collects a list of unreleased objects of type **item** and writes them to the **list_items.txt** file:

```
collect_garbage -u=infodba -p=infodba -g=dba -item -rf=list_items.txt
-query
```

- The following example collects a list of unreleased objects of type **item** and writes them to the **item_report.txt** file:

```
collect_garbage -u=infodba -p=infodba -g=dba -item -query
```

- The following example generates a list of objects for the following types: item, dataset, form, envelope, and folder. The list is placed in the following files, respectively: **item_report.txt**, **dataset_report.txt**, **form_report.txt**, **envelope_report.txt**, and **folder_report.txt**.

```
collect_garbage -all -query
```

The following examples illustrate how to use the **-if** argument:

- The following example displays a message and exits the program because no value was provided for the **-if** argument:

```
collect_garbage -item -if=
```

- The following example checks each entry in the **item_report_aa** file. A folder named **WBITEM_item_report_aa** is created within the **WASTE BASKET** folder and all unreferenced objects are placed within this folder.

```
collect_garbage -item -if=item_report_aa
```

- The following example checks each entry in the given list and identifies unreferenced objects. A subfolder is created within the **WASTE BASKET** folder and all unreferenced objects are placed within this subfolder. It also displays object information, such as object name, ID, object type and owner's name.

```
collect_garbage -u=infodba -p=infodba -g=dba -item -if=list_items.txt  
-report
```

The following examples illustrate how to use the **-sub_folder** argument:

- The following example deletes the objects of type **dataset** within the **WBDSET_dataset_report1** folder within the **WASTE BASKET** folder.
- The following example deletes all instances of type **dataset** within the **WASTE BASKET** folder and all instances in the subfolder named **WBDSET_file-name**:

```
collect_garbage -dataset -delete
```

This section illustrates how to use the **-query** argument. The following example compiles a list of objects of type **item** that are not released and writes the list to the **list_items.txt** file:

```
collect_garbage -u=infodba -p=infodba -g=dba -item -rf=list_items.txt  
-query
```

The following example shows how to use the **-dataset**, **-child_references**, and **-ignore_relation** arguments. In the example, all unreferenced datasets that have secondary objects are collected as garbage, except those that have any of the secondary object attached with the **IMAN_Rendering** or **IMAN_specification** relationship.

```
collect_garbage -u=infodba -p=<password> -g=dba  
-dataset -query -rf=report.txt -child_references  
-ignore_relation=IMAN_Rendering,IMAN_specification
```

IMPORTANT NOTES

- The **-report** option for orphan operations outputs in a different format than for other object types, because orphans are not valid workspace objects.
- The **-item** option collects the associated item revisions along with the item.

- Errors are reported in the **logfile** when running in orphan collection mode. The error reported indicates that an error attempting to load an indirected object. These errors can be disregarded.

dataset_cleanup

Repairs corrupted datasets and removes orphaned revision anchors.

Caution Siemens PLM Software recommends that you run this utility only when there is no other activity on the database.

PROBLEM IDENTIFIERS

A dataset is identified as corrupted if any of the following problems are found:

- Dataset has no reference to an **ImanFile** object.
- Dataset has reference to an **ImanFile** object, but the corresponding operating system file does not exist and the dataset is not archived.
- Dataset is an orphan (that is, the dataset refers to the anchor but the anchor does not go to dataset).
- Anchor refers to datasets that do not exist.
- Anchor size = 0.

OBJECT CLEANUP RULES

A dataset object is reattached to revision anchor if it is an orphan but is referenced by some other objects, or deleted if it meets the following criteria:

- Dataset is an orphan and is not referenced.
- Dataset is not archived and the associated operating system file does not exist.

ANCHOR CLEANUP RULES

The **dataset_cleanup** utility repairs dataset revision anchors as follows:

- If the anchor refers to nonexistent datasets, the references are removed from the anchor.
- If the anchor size = 0, the anchor is deleted.

SYNTAX

```
dataset_cleanup [-u=user-id {-p=password | -pf=password-file} -g=group]
-rf=file-name | -if=file-name
[-of=log-file-name] [-b=beginning-anchor]
[-e=ending-anchor] [-start_date=start-date] [-end_date=end-date] -h
```

ARGUMENTS

-u

Specifies the user ID.

This is generally **infodba** or another user with administration privileges. If this argument is used without a value, the operating system user name is used.

Note If Security Services single sign-on (SSO) is enabled for your server, the **-u** and **-p** arguments are authenticated externally through SSO rather than being authenticated against the Teamcenter database. If you do not supply these arguments, the utility attempts to join an existing SSO session. If no session is found, you are prompted to enter a user ID and password.

-p

Specifies the password.

If used without a value, the system assumes a null value.

If this argument is not used, the system assumes the *user-ID* value to be the password.

This argument is mutually exclusive with the **-pf** argument.

-pf

Specifies the password file. If used without a value, the system assumes a null value. If this argument is not used, the system assumes the *user-ID* value to be the password.

For more information about managing password files, see [Manage password files](#).

This argument is mutually exclusive with the **-p** argument.

-g

Specifies the group associated with the user.

If used without a value, the user's default group is assumed.

-rf

Creates a report file listing the corrupted datasets.

-if

Uses the report file as input to purge corrupted datasets or repair revision anchors.

-of

Cleans up and logs the results to a log file. This argument must be supplied if the **-if** argument is used but is optional with the **-rf** argument.

-a

Specifies that corrupt anchors (those that are orphaned and are not referenced by a dataset) be deleted and a message be provided.

-b

Specifies the first revision anchor of a contiguous series to be repaired. The default value is **1**.

-e

Specifies the last revision anchor of a contiguous series to be repaired. The default value is **last**.

Note A revision anchor is an object that keeps track of a set of revisions of some object. One such class of objects is datasets.

-start_date

Specifies the starting date to search for datasets that have been modified from this date. Use this argument with the **-end_date** argument.

The format of the date is “*DD-MMM-YYYY HH:MM:SS*” and must be inside the double quotes because of the space between the year and the hour. This argument is used only with the **-rf** argument.

-end_date

Specifies the ending date to search for datasets that have been modified until this date. This argument is optional and is used only with the **-start_date** argument. If this argument is not specified, the end date is the current date.

The format of the date is “*DD-MMM-YYYY HH:MM:SS*” and must be inside the double quotes because of the space between the year and the hour. This argument is used only with the **-rf** argument.

-h

Displays help for this utility.

ENVIRONMENT

As specified in [Manually configuring your environment for Teamcenter utilities](#).

FILES

As specified in [Log files produced by Teamcenter](#).

RESTRICTIONS

None.

EXAMPLES

- To generate a report file called **myreportfile** listing corrupted dataset objects, enter the following command on a single line:

```
$TC_ROOT/bin/dataset_cleanup -u=infodba -p=password -g=dba
-rf=myreportfile
```

- To run the **dataset_cleanup** utility using the **myreportfile** file as input, enter the following command on a single line:

```
$TC_ROOT/bin/dataset_cleanup -u=infodba -p=password -g=dba
-if=myreportfile -of=mylogfile
```

- On a database with 1000 dataset revision anchors, you could run the **dataset_cleanup** utility as follows:

```
$TC_BIN/dataset_cleanup -u=infodba -p=infodba -g=dba -b=1 -e=500
-rf=dataset_cleanup_500.report
$TC_BIN/dataset_cleanup -u=infodba -p=infodba -g=dba -b=501 -e=1000
-rf=dataset_cleanup_1000.report
```

- To purge all datasets with modification dates between Oct-01-2007 and Oct-10-2007:

```
dataset_cleanup -u=infodba -p=password -g=dba
-start_date="01-OCT-2007 00:00:00" -end_date="10-Oct-2007
00:00:00" -rf=ttt.txt
```

- To purge all datasets with modification dates from Oct-01-2007 to the current date:

```
dataset_cleanup -u=infodba -p=password -g=dba
-start_date="01-OCT-2007 00:00:00" -rf=ttt.txt
```

CLEANING UP
DATASETS
AND
REPAIRING
REVISION
ANCHORS

Perform the following steps to clean up corrupted datasets:

1. Use the **dataset_cleanup** utility to generate a report file called **myreportfile** listing the corrupted dataset objects in the database by entering the following command on a single line:

```
$TC_ROOT/bin/dataset_cleanup -u=infodba -p=password -g=dba
-rf=myreportfile
```

The report file contains a list of corrupted datasets sorted by **Object_UID**. The report also contains the problem identifier, dataset name, and ownership.

If the **-a** argument is specified on the command line, the utility deletes the corrupt anchors and displays a message to the user. If the **-a** argument is not supplied, a message is displayed indicating that the anchor was skipped and the **-a** option should be used.

You must review the report file and decide which datasets, if any, should not be purged from the database.

2. Use a text editor to remove any references to dataset objects that should not be purged from the database from the report file.
3. Run the **dataset_cleanup** utility using the **myreportfile** file as input to purge corrupted dataset objects from the database or fix anchors and log the results to the **mylogfile** file by entering the following command on a single line:

```
$TC_ROOT/bin/dataset_cleanup -u=infodba -p=password -g=dba
-if=myreportfile -of=mylogfile
```

The utility attempts to fix the revision anchor, attach the dataset to another revision anchor, or purge the datasets from the database.

A final output report is generated showing the results for each dataset. The report displays the following message if the operation is successful:

```
problem deleted
```

If the operation is unsuccessful, the following message is displayed:

```
could not delete error stack number
```

4. Display information about the dataset cleanup process by entering the following command:

```
ps -ef | grep data
```

5. Kill the dataset cleanup process by entering the following command:

```
kill -9 PID
```

PID is the operating system process ID returned in step 4.

delete_item_data

Deletes unused item revisions from the database. The item revisions to be removed are contained in an input file created by the user.

SYNTAX

```
delete_item_data [-u=user-id {-p=password | -pf=password-file} -g=group]
[-inputFile=input-file | -inputKeyFile=input-file]
-configFile=configuration-file [-outputDir=output-file-directory]
{-mode=report | delete} [-delimiter=delimiter] [-h]
```

ARGUMENTS

-u

Specifies the user ID.

This is generally **infodba** or another user with administration privileges. If this argument is used without a value, the operating system user name is used.

Note

If Security Services single sign-on (SSO) is enabled for your server, the **-u** and **-p** arguments are authenticated externally through SSO rather than being authenticated against the Teamcenter database. If you do not supply these arguments, the utility attempts to join an existing SSO session. If no session is found, you are prompted to enter a user ID and password.

-p

Specifies the password.

If used without a value, the system assumes a null value.

If this argument is not used, the system assumes the *user-ID* value to be the password.

-pf

Specifies the password file. If used without a value, the system assumes a null value. If this argument is not used, the system assumes the *user-ID* value to be the password.

For more information about managing password files, see [Manage password files](#).

This argument is mutually exclusive with the **-p** argument.

-g

Specifies the group associated with the user.

If used without a value, the user's default group is assumed.

-inputFile

Specifies the absolute path of the input file. The input file has the following format (the forward slash, /, is the delimiter):

```
item id/Item Revision1
item id/Item Revision2
item id/Item Revision3
```

-inputKeyFile

Specifies the absolute path of the input file. The input file has the following format:

```
[keyAttr1=keyVal1] [,keyAttr2=keyVal2]...[,keyAttrN=keyValN],
```

```
rev_id=A item_id=000100,rev_id=B
```

-configFile

Specifies the absolute path of the configuration file. The configuration file has the following field names and format:

```
EXCLUDE = Relationship@C=Class; Relationship@T=Type; ...;
INCLUDE = Relationship@T=Type; Relationship@C=Class; ...;
```

The **EXCLUDE** and **INCLUDE** expressions contain three values as shown in the following examples:

```
Relationship@C=Class
Relationship@T=Type
```

- The first value of the expression contains the relation value followed by the @ separator.
- The second value of the expression contains the type name (**T**) or the class name (**C**) followed by the = separator.
- The third value of the expression contains the value of the type or class followed by the ; delimiter.

Note

- If there is only one relation specified in the expression, it must be terminated with the ; delimiter. For example:

```
IMAN_reference;
```

If there is more than one expression, the expressions must be separated by the ; separator. For example:

```
IMAN_specification@T=MSWord;IMAN_reference@C=PSBOMViewRevision;
```

- To exclude the based-on item revisions to be deleted, add the **IMAN_based_on@C=ItemRevision;** expression to the **EXCLUDE** section.

For example, assume item revision B was revised from A. When item revision B is deleted, revision A will also be deleted unless the **IMAN_based_on@C=ItemRevision;** expression is in the **EXCLUDE** section.

This argument is required.

-outputDir

Specifies the path where report and log files are to be written.

The default value is the current directory.

-mode

Specifies one of the following the modes:

- **report**
Generates a summary report containing the following information:
 - o Target item revision ID

- o Reference status
- o Usage status
- o Site ownership
- o Exported replica status
- o Deletion status
- **delete**

Deletes the item revision and its associated objects except those specifically excluded by entries in the configuration file. If the item revision is the only revision of an item, the item is also deleted. If the object is a dataset, all versions of the target dataset are deleted as well as all forms and named references associated with the dataset.

This argument is required.

-delimiter

Specifies the delimiter character separator between the item ID and the item revision ID. The default is the forward slash (/).

-h

Displays help for this utility.

RESTRICTIONS

None.

EXAMPLES

- The following is an example of an input file:

```
ABC000075/A
ABC000074/A
ABC000092/A
ABN000002/A
ABN000011/A
ABN000058/A
```

- The following is an example of a configuration file:

```
EXCLUDE = IMAN_specification@T=UGMASTER;  
INCLUDE = IMAN_specification@T=MSWord;IMAN_reference@C=PSBOMViewRevision;
```

If there are no configuration entries in the configuration file, the **EXCLUDE** and **INCLUDE** values must be assigned with the ; delimiter as shown below:

```
EXCLUDE=;  
INCLUDE=;
```

Any statements not conforming to the format above are not processed for evaluation. This example indicates that the attached objects are not processed for evaluation and but they are only dereferenced from their parent item revision.

- The following is an example of the **delete_item_data** command line entry:

```
delete_item_data -u=infodba -p=infoda -g=dba  
-inputFile=c:\temp\input.txt  
-configFile=c:\temp\config.txt -mode=report
```

- The following is an example of using the **delimiter=@** argument:

```
000001@A  
000002@B
```

hsm_capacity_alert

Evaluates if the Teamcenter volume tiers filled capacity exceeds the specified alert capacity levels. When the filled capacity exceeds the alert capacity level, an e-mail is sent to the system administrator. The capacity levels are measured in percentage of total capacity.

Sites can schedule this utility to execute overnight using a UNIX **cron** job or the Microsoft Windows **at** command. This utility can require considerable time to evaluate the capacity levels on different tiers.

SYNTAX

hsm_capacity_alert [-u=*user-id* {-p=*password* | -pf=*password-file*} -g=*group*]
-alertcapacity=*percentage* -tier=*tier-level* [-v] [-h]

ARGUMENTS

-u

Specifies the user ID.

This is generally **infodba** or another user with administration privileges. If this argument is used without a value, the operating system user name is used.

Note

If Security Services single sign-on (SSO) is enabled for your server, the **-u** and **-p** arguments are authenticated externally through SSO rather than being authenticated against the Teamcenter database. If you do not supply these arguments, the utility attempts to join an existing SSO session. If no session is found, you are prompted to enter a user ID and password.

-p

Specifies the password.

If used without a value, the system assumes a null value.

If this argument is not used, the system assumes the *user-ID* value to be the password.

-pf

Specifies the password file. If used without a value, the system assumes a null value. If this argument is not used, the system assumes the *user-ID* value to be the password.

For more information about managing password files, see [Manage password files](#).

This argument is mutually exclusive with the **-p** argument.

-g

Specifies the group associated with the user.

If used without a value, the user's default group is assumed.

-alertcapacity

Specifies alert capacity as a percentage of total capacity. You can include the percent symbol (%) in this argument.

-tier

Specifies volume tier. Valid values are **1**, primary, and **2** secondary.

-v

Verbose mode. Provides information about results and progress.

-h

Displays help for this utility.

ENVIRONMENT

- The generic command window set with all Teamcenter-related environments.
- As specified in *Manually configuring your environment for Teamcenter utilities*.

FILES

As specified in *Log files produced by Teamcenter*.

RESTRICTIONS

This utility requires that the **HSM_primary_tier_hosts** and **HSM_secondary_tier_capacity** preferences contain estimated total capacity values. These preferences are set using **preferences_manager** utility.

**RETURN
VALUES**

Return value 0
upon success

Return value 1
upon failure

EXAMPLES

To determine if the primary tier capacity exceeds 80% of total capacity, enter the following command:

```
hsm_capacity_alert -u=infodba -p=infodba -g=dba -tier=1 -alertcapacity=80%
```

When the filled capacity exceeds 80 percent of total capacity level, an e-mail is sent to the system administrator.

hsm_report

Evaluates any or all hierarchical storage management (HSM) policies to generate a report. Because of performance considerations using the rich client application interface, Siemens PLM Software recommends the system administrator execute this utility to create pending migration file sets.

Sites can schedule this utility to execute overnight through a UNIX **cron** job or Microsoft Windows **at** command. This utility can take considerable time to evaluate all migration policies.

SYNTAX

```
hsm_report [-u=user-id {-p=password | -pf=password-file} -g=group]
-tier={1 | 2 | 3} -migrationreport=[ALL | migration-policy] [-before=before-date]
[-after=after-date] -filepath=filepath
[-listpolicies=ALL | policy-name] [-v] [-h]
```

ARGUMENTS

-u

Specifies the user ID.

This is generally **infodba** or another user with administration privileges. If this argument is used without a value, the operating system user name is used.

Note

If Security Services single sign-on (SSO) is enabled for your server, the **-u** and **-p** arguments are authenticated externally through SSO rather than being authenticated against the Teamcenter database. If you do not supply these arguments, the utility attempts to join an existing SSO session. If no session is found, you are prompted to enter a user ID and password.

-p

Specifies the password.

If used without a value, the system assumes a null value.

If this argument is not used, the system assumes the *user-ID* value to be the password.

-pf

Specifies the password file. If used without a value, the system assumes a null value. If this argument is not used, the system assumes the *user-ID* value to be the password.

For more information about managing password files, see [Manage password files](#).

This argument is mutually exclusive with the **-p** argument.

-g

Specifies the group associated with the user.

If used without a value, the user's default group is assumed.

-tier

Specifies migration tier. Valid values are:

1 Primary to secondary.

2 Secondary to tertiary.

3 Primary to tertiary.

-migrationreport

Specifies migration policy for which the report need to be reported. To generate a report for all active migration policies, specify **-migrationreport=ALL**.

-before

Specifies the beginning date for reporting migration, for example, **23_Mar_2004**.

-after

Specifies the ending date for reporting migration, for example, **23_May_2006**.

-filepath

Specifies the operating system file path to which the report is saved.

-listpolicies

Indicates that the utility is to list all active policies.

-v

Verbose mode provides information about results and progress.

-h

Displays help for this utility.

ENVIRONMENT

- The generic command window set with all Teamcenter-related environments.
- As specified in *Manually configuring your environment for Teamcenter utilities*.

FILES

As specified in *Log files produced by Teamcenter*.

RESTRICTIONS

None

EXAMPLES

- Execute the following command to generate report on all policies in the database:

```
hsm_report -u=infodba -p=infodba -g=dba  
-migrationreport=ALL -tier=1 -filepath=c:\temp\report.txt
```

- Execute the following command to generate report about a specific policy in the database:

```
hsm_report -u=infodba -p=infodba -g=dba  
-migrationreport=policy-name -filepath=c:\temp\report.txt
```

- Execute the following command to list of all active policies names defined for a database:

```
hsm_report -u=infodba -p=infodba -g=dba -listpolicies
```

index_verifier

Detects missing indexes in a Teamcenter database. There are five types of indexes that can be detected using this utility:

- Indexes on the primary key of each Teamcenter class.

The PUID (internal attribute of each table mapped to a Teamcenter class) must have a unique index.

- Indexes on variable length arrays (VLA).

Each VLA must have two indexes, as follows:

- o An index on the PUID+PSEQ. This index must be unique.
- o An index on the PVAL attribute (**pvalu_0** for **POM_typed/untyped_reference** and **pval_0** for the other data type).

- Indexes created by Teamcenter.

These indexes are created using Teamcenter POM ITK and information about these indexes resides in the POM data dictionary **pom_indexes** table.

- Functional indexes

This utility detects the necessary functional indexes required by the version of Teamcenter in use. These functional indexes may include, but are not limited to the following:

- o A functional index on **WorkspaceObject.object type**
- o A functional index on **WorkspaceObject.object_desc**
- o A functional index on **WorkspaceObject.object_name**
- o A functional index on **Item.Item_id**
- o A functional index on **ItemRevision.Item_revision_id**

- Indexes on system tables such as **pom_backpointer**, **pom_m_lock**, and the **pm_process_list** tables.

SYNTAX

```
index_verifier [-u=user-id {-p=password | -pf=password-file} -g=group]
[-o= [DRYRUN|DO_IT]] [-h]
```

ARGUMENTS

-u

Specifies the user ID.

This is generally **infodba** or another user with administration privileges. If this argument is used without a value, the operating system user name is used.

Note If Security Services single sign-on (SSO) is enabled for your server, the **-u** and **-p** arguments are authenticated externally through SSO rather than being authenticated against the Teamcenter database. If you do not supply these arguments, the utility attempts to join an existing SSO session. If no session is found, you are prompted to enter a user ID and password.

-p

Specifies the password.

If used without a value, the system assumes a null value.

If this argument is not used, the system assumes the *user-ID* value to be the password.

This argument is mutually exclusive with the **-pf** argument.

-pf

Specifies the password file.

For more information about managing password files, see [Manage password files](#).

This argument is mutually exclusive with the **-p** argument. One of the two mutually exclusive password elements is required.

-g

Specifies the group associated with the user.

If used without a value, the user's default group is assumed.

-o

Specifies the action to take with the output of the **index_verifier** utility. **index_verifier** outputs SQL statements that can create database indexes. The **-o** argument has the following parameters.

-o DRYRUN

Specifies to output the SQL statements without applying them to the database. This is the same behavior as not using the **-o** argument.

-o DO_IT

Specifies to immediately apply the SQL statements to the database to create indexes.

For Oracle, if the **SA** user who runs **index_verifier** wants missing indexes to be created in a specific tablespace, the **SA** user must set the **TC_INDEX_STORAGE** environment variable:

```
TC_INDEX_STORAGE=PARALLEL 8 NOLOGGING TABLESPACE tablespace-name
```

This setting uses up to 8 parallel processes to create indexes and push these indexes into *tablespace-name*.

-h

Displays help for this utility.

ENVIRONMENT

As specified in [Manually configuring your environment for Teamcenter utilities](#).

FILES

As specified in [Log files produced by Teamcenter](#).

RESTRICTIONS

This utility can be run while users are logged on to Teamcenter.

EXAMPLES

None.

mark_for_migrate

Evaluates any or all active policies to determine if there are any Teamcenter volume files pending for migration, and if found, mark it for migration. Because of performance considerations using the rich client application interface, Siemens PLM Software recommends the system administrator execute this utility to create pending migration file sets.

Sites can schedule this utility to execute overnight through a UNIX **cron** job or Microsoft Windows **at** command. This utility can take considerable time to evaluate all migration policies.

SYNTAX

mark_for_migrate [-u=*user-id* {**-p**=*password* | **-pf**=*password-file*} **-g**=*group*]
-migrationpolicy=[**ALL** | *policy-name*]
[-listpolicies] [-v] [-h]

ARGUMENTS**-u**

Specifies the user ID.

This is generally **infodba** or another user with administration privileges. If this argument is used without a value, the operating system user name is used.

Note

If Security Services single sign-on (SSO) is enabled for your server, the **-u** and **-p** arguments are authenticated externally through SSO rather than being authenticated against the Teamcenter database. If you do not supply these arguments, the utility attempts to join an existing SSO session. If no session is found, you are prompted to enter a user ID and password.

-p

Specifies the password.

If used without a value, the system assumes a null value.

If this argument is not used, the system assumes the *user-ID* value to be the password.

-pf

Specifies the password file. If used without a value, the system assumes a null value. If this argument is not used, the system assumes the *user-ID* value to be the password.

For more information about managing password files, see [Manage password files](#).

This argument is mutually exclusive with the **-p** argument.

-g

Specifies the group associated with the user.

If used without a value, the user's default group is assumed.

-migrationpolicy

Evaluates a given migration policy and marks pending files for migration. To evaluate all active migration policies and mark them for migration, specify

-migrationpolicy=ALL.

-listpolicies

List all active policies in chronological order.

-v

Verbose mode. Provides information about results and progress.

-h

Displays help for this utility.

ENVIRONMENT

- The generic command window set with all Teamcenter-related environments.
- As specified in [Manually configuring your environment for Teamcenter utilities](#).

FILES

As specified in [Log files produced by Teamcenter](#).

RESTRICTIONS

None.

**RETURN
VALUES**

Return value 0
upon success

Return value 1
upon failure

EXAMPLES

- Execute the following command to evaluate all policies in the database:

```
mark_for_migrate -u=infodba -p=infodba -g=dba -migrationpolicy=ALL
```
- Execute the following command to evaluate a specific policy in the database:

```
mark_for_migrate -u=infodba -p=infodba -g=dba -migrationpolicy=policy_name
```
- Execute the following command to receive a list of all active policies names defined for a database:

```
mark_for_migrate -u=infodba -p=infodba -g=dba -listpolicies
```

move_volume_files

Moves files from one Teamcenter volume to another using FMS to move the files over your WAN/LAN. You can specify selected files to move based on file date, file age, or by volume allocation rules.

For more information about using volume allocation rules to reallocate volume files, see the [System Administration Guide](#).

You can run this utility as a **cron** job or as a scheduled task using **process_move_file_volumes** as a **.sh** or **.bat** script, respectively. Use this script if your scheduling tools are not running in the Teamcenter environment with the appropriate **TC_ROOT** and **TC_DATA** variables set. The script sets these variables and calls **tc_providevars** directly before running the **move_volume_files** utility. The script accepts the same arguments as the utility. The arguments specified in the script are run by the utility.

SYNTAX

```
move_volume_files -u=user-id {-p=password | -pf=password-file} -g=group
-f={list | move} [-output_file=file-name] [-srcvol=source-volume]
[-destvol=destination-volume]
[-listvolumes] [-before=access-time-ending-range] [-after=access-time-beginning-range]
[-maxage=days] [-fszlessthan=bytes] [-fszgreaterthan=bytes]
[-outrulesfile=file-name] [-rulesfile=file-name] [-excludedvollist=file-name]
[-presorted_file=file-name] [-listaf] [-transferaf] [-v] [-h]
```

ARGUMENTS**-u**

Specifies the user ID.

This is generally **infodba** or another user with administration privileges. If this argument is used without a value, the operating system user name is used.

Note

If Security Services single sign-on (SSO) is enabled for your server, the **-u** and **-p** arguments are authenticated externally through SSO rather than being authenticated against the Teamcenter database. If you do not supply these arguments, the utility attempts to join an existing SSO session. If no session is found, you are prompted to enter a user ID and password.

-p

Specifies the password.

If used without a value, the system assumes a null value.

If this argument is not used, the system assumes the *user-ID* value to be the password.

-pf

Specifies the password file. If used without a value, the system assumes a null value. If this argument is not used, the system assumes the *user-ID* value to be the password.

For more information about managing password files, see [Manage password files](#).

This argument is mutually exclusive with the **-p** argument.

-g

Specifies the group associated with the user.

If used without a value, the user's default group is assumed.

-f

Specifies whether the utility lists or moves files.

-list

Lists the files to be moved in the specified source volume that are candidates to be moved.

This argument must be used with the **-srcvol** argument.

-move

Moves the Teamcenter files to the specified destination volume.

This argument must be used with both the **-srcvol** and **-destvol** arguments.

-output_file

Specifies the file to which list output is written when using the **-f=list** or **-listsf** arguments. If not specified, the **move_tcfiles_list.txt** file is created.

-srcvol

Specifies the name of the Teamcenter source volume.

-destvol

Specifies the name of the Teamcenter destination volume.

-listvolumes

Displays a list of all Teamcenter volumes configured in the database.

-before

Specifies the ending access time in the form of *dd-mmm-yyyy*, for example, **06-Jan-2008**. All files with this last access date or earlier are selected. If the **-after** argument is specified, then the files selected are within that range, inclusive.

-after

Specifies the beginning access time in the form of *dd-mmm-yyyy*, for example, **23-Mar-2007**. All files with this last access date or later are selected. If the **-before** argument is specified, then the files selected are within that range, inclusive.

-maxage

Specifies the maximum age (in days) a file can reach and still be eligible for transfer. For example, if set to **24**, any file 24-days old or younger can be transferred.

If used outside of a rules environment, the current date minus the age specified by this argument is used as the **-after** value.

-fszlessthan

Specifies the maximum file size (in bytes) a file can reach and still be eligible for transfer. For example, if set to **64000**, all files that are 64,000 bytes or smaller can be transferred.

If the **-fszgreaterthan** argument is specified, the files selected are within that range, inclusive.

-fszgreaterthan

Specifies the minimum file size (in bytes) a file can reach and still be eligible for transfer. For example, if set to **2048**, all files that are 2,408 bytes or larger can be transferred.

If the **-fszlessthan** argument is specified, the files selected are within that range, inclusive.

-outrulesfile

Outputs the volume allocation rules XML template to the current directory. The XML template is given the filename specified by this argument. For example, if set to **VolRules.xml**, an XML template of the allocation rules named **VolRules.xml** is created and stored in the current directory.

-rulesfile

Specifies the path and name of the volume allocation rules XML file to evaluate. The file is validated against the DTD, and then its rules are evaluated. Volume files are moved or listed, based on the setting of the **-f** argument.

This argument must be used with the **-f** argument. It does not work with the **-listvolumes**, **-srcvol**, **-destvol**, or **-presorted_file** arguments.

-excludedvollist

Specifies the path and name of the file containing a list of volumes to be excluded from both listing and transfer actions. This argument is typically used to list default local volumes (store and forward volumes) to ensure the files stored in these temporary volume location are not transferred.

Use either the full path to the file, or use the partial path/file name, in which case the utility searches for the file name in the current directory.

Any number of volumes can be specified in this file. Each entry must be a valid volume name, listed on its own row in the file.

-presorted_file

Specifies the path and name of the file containing a list of files to be moved. This argument must be used with the **-f** and **-move** arguments.

Use either the full path to the file, or use the partial path/file name, in which case the utility searches for the file name in the current directory.

Any number of files can be specified in this file. The file format is the same format as the file output by the **-f=list** argument. Typically, you generate a file using this **-f=list** argument, edit the file as necessary, and then set the **-presorted_file** argument with the name of the edited file.

-listaf

Lists files evaluated for a store and forward operation.

-transferaf

Transfers all files evaluated for a store and forward operation from all of the user's local volumes to their corresponding default volumes. It also creates and schedules a dispatcher request to clean up files in local volumes.

Note The **-listaf** and **-transferaf** arguments require installation of dispatcher scheduler, module, client, and the **store_and_forward** translator. They also require setting the **TC_Store_and_Forward** and **FMS_SAF_Batch_Transfer_Enabled** preferences to true.

For more information, see the *System Administration Guide*

-v

Executes the utility in verbose mode.

-h

Displays help for this utility.

RESTRICTIONS

If the volume path definition is defined in multiple locations, the definition must be exactly the same in each location. For example, if you define a volume in the Organization application using a drive/path definition format, you must use the same drive/path definition in the FMS master configuration file (you cannot use the UNC format).

Conversely, the volume can be defined in both Organization and in the master configuration file in UNC format.

EXAMPLES

- Enter the following command on a single line to generate the list of all files in Teamcenter volume **vol003**:

```
move_volume_files -u=infodba -p=infodba -g=dba -f=list
-srcvol=vol003 -destvol=vol1001 -v
```

- Enter the following command on a single line to generate the list of all files in Teamcenter volume **vol003** that were last accessed in the given date range:

```
move_volume_files -u=infodba -p=infodba -g=dba -f=list -srcvol=vol003
-destvol=vol1001 -after=03-Feb-2008 -before=03-Mar-2008 -v
```

The list of evaluated files is stored in the **move_tcfiles_list.txt** file, in the current directory.

- Enter the following command on a single line to move all the files listed in the **move_tcfiles_list.txt** file to volume **vol003**:

```
move_volume_files -u=infodba -p=infodba -g=dba -f=move
-preSorted_file=move_tcfiles_list.txt -destvol=vol1003 -v
```

- Enter the following command on a single line to relocate all Teamcenter files from volume **vol003** to the destination volume **newvol002**:

```
move_volume_files -u=infodba -p=infodba -g=dba -f=move
-srcvol=vol003 -destvol=newvol002 -v
```

- Enter the following command on a single line to relocate Teamcenter files that were last accessed on November 29, 2006, or later, from Teamcenter **vol002** to the destination volume **newvol003**:

```
move_volume_files -u=infodba -p=infodba -g=dba -f=move
-srcvol=vol002 -destvol=newvol003 -after=29-Nov-2006 -v
```

- Enter the following command on a single line to generate the volume allocation rules XML template named **VolSelectionRules.xml** and stored in the current directory:

```
move_volume_files -u=infodba -p=infodba -g=dba -outrulesfile=VolSelectionRules.xml
```

- Enter the following command on a single line to evaluate the volume allocation rules in the **VolSelectionRules.xml** file and store the results in the **move_tcfiles_list.txt** file in the current directory:

```
move_volume_files -u=infodba -p=infodba -g=dba -rulesfile=VolSelectionRules.xml  
-f=list -v
```

- Enter the following command on a single line to evaluate the volume allocation rules in the **VolSelectionRules.xml** file and move the evaluated files:

```
move_volume_files -u=infodba -p=infodba -g=dba -rulesfile=VolSelectionRules.xml  
-f=move -v
```

- Enter the following command on a single line to generate the list of all files evaluated for a store and forward operation:

```
move_volume_files -u=infodba -p=infodba -g=dba -listsaf -v
```

- Enter the following command on a single line to transfer files evaluated for a store and forward operation from all of the users' local volumes to their corresponding default volumes and to schedule a dispatcher request to clean up files in local volumes:

```
move_volume_files -u=infodba -p=infodba -g=dba -transfersaf -v
```

purge_datasets

Removes (purges) old versions of datasets from the database and outputs a list of each dataset purged, along with the owning user and group.

Note Normally, Teamcenter stores a fixed number of dataset versions in the database. The maximum number of datasets retained is set using the [AE_dataset_default_keep_limit](#) preference.

For more information, see the [Preferences and Environment Variables Reference](#).

Certain conditions prevent automatic purging of old datasets. For example, when a user does not have permission to purge a dataset owned by another user, or when a group is given read/write permission but not delete permission.

Additionally, datasets are not purged when the named references in **version0** do not match the named references in the latest dataset version. Use the **-skipInconsistencyCheck** argument to bypass this named references check. Use this argument only in situations in which you know why the named references differ and are confident that purging the older dataset versions will not result in loss of needed data. Possible situations include CAD integrations in which custom code is implemented, wrong coding exists from custom ITK programs, and PLM XML import.

Before using the **-skipInconsistencyCheck** argument, run this utility without the argument and review the output for any failed purges. Investigate all datasets that did not purge, correcting problems if necessary.

Example **version0** of a dataset contains three named references: **a.txt**, **b.txt** and **c.txt**. The latest version of the same dataset contains two named references: **a.txt** and **b.txt**. Using this utility to purge this dataset fails because of the inconsistency between the named references. The utility logs an inconsistent data message.

In this situation, you should compare the named references between the versions and resolve the inconsistency if necessary.

- Compare the named references of the two versions in the rich client by choosing **View→Named References**.
- Correct the inconsistency by copying the **c.txt** named reference from **version0** to the clipboard, then pasting it into the latest version. The named references must be in the same order for both versions. Rerunning the utility would purge this dataset.

Alternatively, checking the dataset out, making changes, and checking it back in synchronizes the named references between the versions.

If it is not possible to synchronize the named references, for example, the datasets are already released (and thus read-only), use this argument to purge the datasets.

SYNTAX

```
purge_datasets [-u=user-id {-p=password | -pf=password-file} -g=group]
-b=beginning_anchor -e=ending_anchor [-k=keep-limit]
-start_date=DD-MMM-YYY HH:MM:SS -end_date=DD-MMM-YYY HH:MM:SS
[-set] [-report] [-replica_only -site=site-name]
[-itemidsfile=item-id-file-name] [-grmtypesfile=relation-types-file-name]
[-includeFolderContents] [-skipInconsistencyCheck] [-h]
```

ARGUMENTS

-u

Specifies the user ID.

This is generally **infodba** or another user with administration privileges. If this argument is used without a value, the operating system user name is used.

Note

If Security Services single sign-on (SSO) is enabled for your server, the **-u** and **-p** arguments are authenticated externally through SSO rather than being authenticated against the Teamcenter database. If you do not supply these arguments, the utility attempts to join an existing SSO session. If no session is found, you are prompted to enter a user ID and password.

-p

Specifies the password.

If used without a value, the system assumes a null value.

If this argument is not used, the system assumes the *user-ID* value to be the password.

-pf

Specifies the password file. If used without a value, the system assumes a null value. If this argument is not used, the system assumes the *user-ID* value to be the password.

For more information about managing password files, see [Manage password files](#).

This argument is mutually exclusive with the **-p** argument.

-g

Specifies the group associated with the user.

If used without a value, the user's default group is assumed.

-b

Specifies the first version anchor (*beginning_anchor*) of a contiguous series to be purged. The default value is **1**. Version anchors are objects that keep track of a set of versions of an object. Datasets are one such class of objects.

-e

Specifies the last version anchor (*ending_anchor*) of a contiguous series to be purged. The default value is **last**. Version anchors are objects that keep track of a set of versions of an object. Datasets are one such class of objects.

-set

Determines if the current keep limit is to be reset to the version limit. If this argument is not used, the current keep limit is not assigned to the version limit. If this argument is used, the current keep limit is set to the version limit, as follows:

- If the **-k** argument is used, always assign the keep limit to the version limit no matter whether the dataset is to be purged or not.
- If the **-k** argument is not used, the current version limit is used to purge; therefore, it need not be reset to the version limit.

-k

Determines the keep limit to be used. If the **-k** argument is set to a particular keep limit, for example **-k=4**, a dataset is purged down to that keep limit. If the **-k** argument is not used, the current version limit of the dataset is used as the purge keep limit.

-start_date

Specifies the start date/time for which datasets are purged. This argument must be used with the **-end_date** argument.

-end_date

Specifies the end date/time for which datasets are purged. This argument must be used with the **-start_date** argument.

-h

Displays help for this utility.

-report

Produces a report of the datasets to be purged, but does not purge the datasets from the database.

-replica_only

Specifies that only replica datasets are purged. The **-site** argument can be used in conjunction with the **-replica_only** argument to purge only the datasets replicated from a specific site.

-site

Specifies the site from which replica datasets will be purged. Valid only in conjunction with the **-replica_only** argument.

-itemidsfile

Specifies an input file containing a list of item IDs. Enter one ID per line or you can enter multiple IDs on one line separated by commas.

-grmtypesfile

Specifies an input file containing a list of specified relation types. Enter one relation type per line or you can enter multiple relation types on one line separated by commas. This parameter is optional. If it is not specified, all relation types are used. This parameter is valid only with the **-itemidsfile** argument.

-includeFolderContents

Specifies a folder containing datasets to be purged. This parameter is optional and is valid only with the **-itemidsfile** argument.

-skipInconsistencyCheck

Bypasses the consistency check of named references. If not specified, a dataset is not purged if the named references in **version0** of the dataset are not the same as the named references in the latest version.

Running the utility with this argument purges previous versions of the dataset.

ENVIRONMENT

As specified in [Manually configuring your environment for Teamcenter utilities](#).

FILES

As specified in [Log files produced by Teamcenter](#).

RESTRICTIONS

1. The **purge_datasets** utility must be run by the **infodba** user. This automatically enables the bypass feature and purges old datasets regardless of owning user and group.

EXAMPLES

The default options are not set and use the version limit of the current dataset. The complete **purge_datasets** command looks like this:

```
$TC_BIN/purge_datasets -u=infodba -p=password -g=dba  
-b=beginning-anchor -e=ending-anchor -k=keep-limit -set
```

purge_volumes

Removes (purges) operating system files that represent deleted Teamcenter objects. The **purge_volumes** utility can be run interactively, in forced execution mode, or periodically.

During a Teamcenter session, users delete objects. However, if an object's associated files remain in the Teamcenter volume, the **purge_volumes** utility unlocks these files so they can be deleted at the operating system level.

SYNTAX

purge_volumes [-u=*user-id* {-p=*password* | -pf=*password-file*} -g=*group*]
[-f] [-s=*time*] -h

ARGUMENTS

-u

Specifies the user ID.

This is generally **infodba** or another user with administration privileges. If this argument is used without a value, the operating system user name is used.

Note

If Security Services single sign-on (SSO) is enabled for your server, the **-u** and **-p** arguments are authenticated externally through SSO rather than being authenticated against the Teamcenter database. If you do not supply these arguments, the utility attempts to join an existing SSO session. If no session is found, you are prompted to enter a user ID and password.

-p

Specifies the password.

If used without a value, the system assumes a null value.

If this argument is not used, the system assumes the *user-ID* value to be the password.

-pf

Specifies the password file. If used without a value, the system assumes a null value. If this argument is not used, the system assumes the *user-ID* value to be the password.

For more information about managing password files, see [Manage password files](#).

This argument is mutually exclusive with the **-p** argument.

-g

Specifies the group associated with the user.

If used without a value, the user's default group is assumed.

-f

Specifies forced execution mode. When the **-f** argument is supplied, files are deleted without prompting for confirmation. When the **-f** argument is not supplied, the **purge_volumes** utility runs interactively and prompts the user before deleting each file.

-s

Specifies sleep time in seconds. After each **purge_volumes** session is complete, it is dormant for the specified time before running again. If the **-s** argument is not supplied, the **purge_volumes** only runs once.

-h

Displays help for this utility.

ENVIRONMENT

As specified in *Manually configuring your environment for Teamcenter utilities*.

FILES

As specified in *Log files produced by Teamcenter*.

RESTRICTIONS

1. When running the **purge_volumes** utility periodically with the **-s** argument, include the **-f** argument. This disables interactive mode and ensures that the utility runs to completion.

EXAMPLES

To run the **purge_volumes** utility interactively, enter the following command on a single line:

```
$TC_ROOT/bin/purge_volumes -u=infodba -p=password -g=dba
```

The **purge_volumes** utility prompts the user before deleting each volume. When complete, the system displays the following message:

```
Purge_Volumes: terminating  
Stop returned 0
```

If no files were deleted, the system displays the following:

```
There are no deleted files in your system
```

Note

In order to run the **purge_volumes** utility at boot time on UNIX platforms, add the following line to the appropriate startup script on your UNIX system, or add it to the **rc** script created by the postinstallation routine for the database configuration you want this activity to work against:

```
purge_volumes -u=infodba -p=password -g=dba -f
```

This starts **purge_volumes** at boot time and disables confirmation of each delete action.

reencode_filenames

Re-encodes volume file names. You may need to rename volume file names on some system configurations depending on the character set of the system where the volume resides.

This operation is required when upgrading system configurations where the character sets of the database and the volume servers do not match and pre-existing volume files exist. This situation can occur when file names are created using TCFS and the system is upgraded to a Teamcenter version that utilizes FMS file services.

By default, the utility does not change the system state and therefore the file names are not renamed.

To commit the changes, add the **-modify=TRUE** argument to the command line.

Siemens PLM Software recommends that the utility be executed first without the **-modify=TRUE** argument to determine the current state of the volumes on the system.

Note Siemens PLM Software recommends sites use this utility with caution and back up all volumes affected by this utility.

SYNTAX

```
reencode_filenames [-u=user-id {-p=password | -pf=password-file} -g=group]
[-l] [-vb] -rf=report-file-name [-vh=volume-host-name]
[-v=volume] [-f=file] [-modify= TRUE | FALSE] [-h]
```

ARGUMENTS

-u

Specifies the user ID.

This is generally **infodba** or another user with administration privileges. If this argument is used without a value, the operating system user name is used.

Note If Security Services single sign-on (SSO) is enabled for your server, the **-u** and **-p** arguments are authenticated externally through SSO rather than being authenticated against the Teamcenter database. If you do not supply these arguments, the utility attempts to join an existing SSO session. If no session is found, you are prompted to enter a user ID and password.

-p

Specifies the password.

If used without a value, the system assumes a null value.

If this argument is not used, the system assumes the *user-ID* value to be the password.

This argument is mutually exclusive with the **-pf** argument.

-pf

Specifies the password file. If used without a value, the system assumes a null value. If this argument is not used, the system assumes the *user-ID* value to be the password.

For more information about managing password files, see [Manage password files](#).

This argument is mutually exclusive with the **-p** argument.

-g

Specifies the group associated with the user.

If used without a value, the user's default group is assumed.

-l

Specifies the utility is to list all volumes.

-vb

Specifies verbose mode.

-rf

Specifies the report file name. This argument is required.

-vh

Specifies the volume host name. The default value is all hosts.

-v

Specifies the volume. The default value is all volumes.

-f

Specifies the file. The default value is all files. If this argument is specified, you must also specify the **-v** argument.

-modify

Specifies whether to rename the files. The default value is to not rename the files and only generate a report. If you specify this argument, you must also specify the **-vh** and/or the **-v** arguments. The value set by this argument is case sensitive. For example, **-modify=TRUE** is valid; **-modify=true** is not valid.

-h

Displays help for this utility.

ENVIRONMENT

As specified in [Manually configuring your environment for Teamcenter utilities](#).

FILES

As specified in [Log files produced by Teamcenter](#).

RESTRICTIONS

None.

EXAMPLES

- To generate a report on all volumes and hosts:

```
reencode_filenames -u=myuserid -p=mypassword -g=mygroup  
-rf=reencode_report.txt
```

- To generate a report and re-encode all file names on a host:

```
reencode_filenames -u=myuserid -p=mypassword -g=mygroup  
-vh=myvolumehost -rf=reencode_report.txt -modify=TRUE
```

- To generate a report and re-encode all file names on a volume:

```
reencode_filenames -u=myuserid -p=mypassword -g=mygroup  
-v=myvolumename -rf=reencode_report.txt -modify=TRUE
```

- To generate a report and re-encode all file names on a volume:

```
reencode_filenames -u=myuserid -p=mypassword -g=mygroup  
-v=myvolumename -f=myfilename -rf=reencode_report.txt -modify=TRUE
```

report_volume

Lists the operating system path of all existing Teamcenter volumes. The path does not include the network node name.

SYNTAX

report_volume [-u=*user-id* {-p=*password* | -pf=*password-file*} -g=*group*]
-f=*file-name* [-date=*yymmddhhmmss*] -h

ARGUMENTS**-u**

Specifies the user ID.

This is generally **infodba** or another user with administration privileges. If this argument is used without a value, the operating system user name is used.

Note If Security Services single sign-on (SSO) is enabled for your server, the **-u** and **-p** arguments are authenticated externally through SSO rather than being authenticated against the Teamcenter database. If you do not supply these arguments, the utility attempts to join an existing SSO session. If no session is found, you are prompted to enter a user ID and password.

-p

Specifies the password.

If used without a value, the system assumes a null value.

If this argument is not used, the system assumes the *user-ID* value to be the password.

-pf

Specifies the password file. If used without a value, the system assumes a null value. If this argument is not used, the system assumes the *user-ID* value to be the password.

For more information about managing password files, see [Manage password files](#).

This argument is mutually exclusive with the **-p** argument.

-g

Specifies the group associated with the user.

If used without a value, the user's default group is assumed.

-f

Specifies the name of the output report file. The utility automatically appends this file name with the **.volumes** extension.

-date

Specifies the date string. Must be in *yymmddhhmmss* format where *yy* is the year, *mm* is the month, *dd* is the day, *hh* is the hour, *mm* is minutes and *ss* is seconds.

This argument is optional. If not supplied, all volumes are reported. Otherwise, only volumes and files created after the input date are reported.

-h

Displays help for this utility.

ENVIRONMENT

As specified in *Manually configuring your environment for Teamcenter utilities*.

FILES

As specified in *Log files produced by Teamcenter*.

RESTRICTIONS

Do not include a directory path with the **-f** argument. The output file must be written to the current working directory.

EXAMPLES

To list all existing volumes in a file called **out.volume**, enter the following on a single line:

```
$TC_ROOT/bin/report_volume -u=infodba -p=infodba -g=dba -f=out
```

review_volumes

Allows you to view volume file attributes regarding OS volumes (file size, last modification date, and so on) and to remove unreferenced operating system files from these volumes.

This utility can generate a report file describing volume usage by various groups and users, as well as reporting any unreferenced operating system files, missing operating system files, and unreferenced Teamcenter files.

Unreferenced operating system files can be deleted at the time a report file is generated or at a later time using a previously-generated report file as an input.

The report file format is plain text (ASCII) and can be manually edited in order to not delete certain files. To prevent files from being deleted, remove any file names before using the report file as input.

You can also save any deleted files to a ZIP format compressed file.

Note Before creating volumes, you must have an FMS server cache (FSC) installed and running, and you must set the [FMS_BootStrap_Urls](#) preference with the FSC host and port information.

SYNTAX

```
review_volumes [-u=user-id {-p=password | -pf=password-file} -g=group]  
-v=volume -rf= file-name | -if=file-name  
[-of=file-name] [-zf=file-name] [-lv]  
[-parallel=number-of-parallel-processes] [-rfolder=folder-name] [-h]
```

ARGUMENTS**-u**

Specifies the user ID.

This is generally **infodba** or another user with administration privileges. If this argument is used without a value, the operating system user name is used.

Note If Security Services single sign-on (SSO) is enabled for your server, the **-u** and **-p** arguments are authenticated externally through SSO rather than being authenticated against the Teamcenter database. If you do not supply these arguments, the utility attempts to join an existing SSO session. If no session is found, you are prompted to enter a user ID and password.

-p

Specifies the password.

If used without a value, the system assumes a null value.

If this argument is not used, the system assumes the *user-ID* value to be the password.

-pf

Specifies the password file. If used without a value, the system assumes a null value. If this argument is not used, the system assumes the *user-ID* value to be the password.

For more information about managing password files, see [Manage password files](#).

This argument is mutually exclusive with the **-p** argument.

-g

Specifies the group associated with the user.

If used without a value, the user's default group is assumed.

-v

Runs the utility against a single, specified volume. This argument is required unless the **-lv** argument is specified.

-rf

Creates a report file listing unreferenced files in volumes.

Note The **-of**, **-if**, and **-zf** arguments do not work in combination with the **-rf** argument. When the **-rf** argument is present, only the report file generation is performed.

-if

Specifies the report file to be used as input to delete unreferenced files in volumes.

-of

Deletes and logs the results to a specified file. This argument must be supplied if the **-if** argument is used.

-zf

Saves deleted files to the specified **ZIP** file. The **.zip** extension is automatically appended to the file name if another extension is not specified.

-lv

Lists all volumes defined in the database.

-parallel

Specifies the number of volumes on which to simultaneously run this utility. You can use this argument to generate reports on all volumes defined in your database simultaneously (use the **-lv** argument to determine the number of volumes defined in your database). However, you must consider available computing resources while setting this value. Even if you have 800 volumes defined in your database, you might only have enough computing power to run five or ten processes in parallel.

This argument must be used with the **-rfolder** argument.

-rfolder

Specifies the folder in which the multiple reports generated by the **-parallel** argument are stored.

This argument must be used with the **-parallel** argument.

-h

Displays help for this utility.

ENVIRONMENT

As specified in [Manually configuring your environment for Teamcenter utilities](#).

FILES

As specified in [Log files produced by Teamcenter](#).

RESTRICTIONS

In versions prior to 8.3.2, you could supply the **-rf** argument in combination with the **-of** argument in order to have the **review_volumes** utility simultaneously generate a report file and delete files from the specified volume.

Beginning with Teamcenter 8.3.2, this requires two separate steps:

1. Run the **review_volumes** utility with the **-rf** argument.
2. Use the **-if** and **-of** arguments to input the previously created **report** file and **delete volume** files.

This change in how arguments are processed was required by the need to improve scalability and performance of the **review_volumes** utility, for example, the addition of the new **-parallel** argument.

EXAMPLES

- To generate a report on a single volume, enter the following command on a single line:

```
$TC_ROOT/bin/review_volumes  
-u=user-id -p=password -g=group -v=volume -rf=file-name
```

- To generate a report on all volumes and delete files, enter the following command on a single line:

```
$TC_ROOT/bin/review_volumes  
-u=user-id -p=password -g=group -of=file-name -v=volume
```

- To delete files on all volumes from a previously executed report, enter the following on a single line:

```
$TC_ROOT/bin/review_volumes  
-u=user-id -p=password -g=group -if=file-name -of=file-name -v=volume
```

- To generate a report on all volumes defined in the database:

```
$TC_ROOT/bin/review_volumes  
-u=user-id -p=password -g=group -parallel=number-of-parallel-processes  
-rfolder=folder-name
```

syncCache

Loops through each dataset row for the specified applications in the cache database and performs the following tests and actions:

- If the **Dataset.Folder** field is either null or empty, the utility performs no action.
- If the **Dataset.Folder** field does not exist or is not a directory, the dataset is deleted from the cache.¹
- If the **Dataset.AppRef** field is not found in Teamcenter, the dataset is deleted from the cache.¹
- The dataset and its related **ItemRev** and **Item** rows in the cache database are updated from Teamcenter. This process performs most of the **Open** process except for downloading the design file. Since no files are downloaded, the staging directory is not modified.

SYNTAX

syncCache *application-name* [-help]

ARGUMENTS

application-name

Specifies a list of ECAD application names so that only the cached datasets created by the applications in the list are processed.

-help

Displays help for this utility.

ENVIRONMENT

Requires the same environment as for running EDA.

FILES AND RESTRICTIONS

None.

EXAMPLES

- To run **syncCache** on datasets created by Cadence Allegro:

```
syncCache cadenceSchematic cadencePcb
```
- To run **syncCache** on datasets created by Mentor BoardStation:

```
syncCache mentor
```

1. When a dataset is deleted from the cache, then the dataset's related ItemRev and Item are also deleted unless they are referenced by other datasets or ItemRev's.

tcmemstat

Monitors the Teamcenter model event manager (TcMEM).

By default, this utility is stored in the *TC_ROOT/tccs/bin* directory.

SYNTAX

tcmemstat [-h] [-x] [-status] [-restart] [-start] [-stop] [-kill]

ARGUMENTS**none**

Displays TcMEM component versions and run time, if running.

-help or -h or -?

Displays help for this utility.

-x

Displays a summary of TcMEM status.

-status

Displays a complete TcMEM status report

-restart

Stops (if running) and restarts TcMEM, effectively reloading the configuration.

-start

Starts TcMEM if it is not already started.

-stop

Shuts down the TcMEM process immediately if no other clients are connected or if all connected clients are idle. Otherwise, a warning message is displayed.

-kill

Immediately and unconditionally shuts down the TcMEM process.

ENVIRONMENT

As specified in [Manually configuring your environment for Teamcenter utilities](#).

FILES

As specified in [Log files produced by Teamcenter](#).

RESTRICTIONS

None.

tspstat

Monitors the Teamcenter server proxy.

By default, this utility is stored in the *TC_ROOT/tccs/bin* directory.

SYNTAX

tspstatt [-h] [-x] [-status] [-config] [-reconfig] [-restart] [-start] [-stop] [-kill]

ARGUMENTS**none**

Displays **TcServerProxy** component versions and run time, if running.

-help or -h or -?

Displays help for this utility.

-x

Displays a summary of **TcServerProxy** status.

-status

Displays a complete **TcServerProxy** status report.

-config

Displays the name of **TcServerProxy** configuration file.

-reconfig

Reloads the **TcServerProxy** configuration. Use this argument to update a running **TcServerProxy** with any HTTP/HTTPS proxy changes.

-restart

Stops (if running) and restarts **TcServerProxy**, effectively reloading the configuration.

-start

Starts **TcServerProxy** if it is not already started.

-stop

Shuts down the **TcServerProxy** process immediately if no other clients are connected or if all connected clients are idle. Otherwise, a warning message is displayed.

-kill

Immediately and unconditionally shuts down the **TcServerProxy** process.

ENVIRONMENT

As specified in [Manually configuring your environment for Teamcenter utilities](#).

FILES

As specified in [Log files produced by Teamcenter](#).

RESTRICTIONS

None.

vms_upgrade

Modifies any commercial part, vendor part, and vendor data created before Teamcenter 2007.1 MP4 to conform to the updated vendor management data model.

The utility creates GRM relations between these items where a **VendorIdentifier** object had previously been used. The **CommercialPart** object is now directly related to the **ManufacturerPart (Vendor Part)** object by the new **VMRepresents** relation, which holds vendor status information. The **VendorIdentifier** and **IdContext** objects are no longer associated with the **CommercialPart** object. The new **TC_vendor_part_rel** GRM relation is used to associate the **ManufacturerPart** with the vendor.

SYNTAX

```
application_root/bin/vms_upgrade [-u=user-id {-p=password | -pf=password-file}
-g=group] -h
```

ARGUMENTS**-u**

Specifies the user ID.

This is generally **infodba** or another user with administration privileges. If this argument is used without a value, the operating system user name is used.

Note

If Security Services single sign-on (SSO) is enabled for your server, the **-u** and **-p** arguments are authenticated externally through SSO rather than being authenticated against the Teamcenter database. If you do not supply these arguments, the utility attempts to join an existing SSO session. If no session is found, you are prompted to enter a user ID and password.

-p

Specifies the password.

If used without a value, the system assumes a null value.

If this argument is not used, the system assumes the *user-ID* value to be the password.

This argument is mutually exclusive with the **-pf** argument.

-pf

Specifies the password file. If used without a value, the system assumes a null value. If this argument is not used, the system assumes the *user-ID* value to be the password.

For more information about managing password files, see [Manage password files](#).

This argument is mutually exclusive with the **-p** argument.

-g

Specifies the group associated with the user.

If used without a value, the user's default group is assumed.

-h

Displays help for this utility.

unmigrate_from_hsm

Reverses migration of the existing Teamcenter volume files objects from the purview of hierarchical storage management (HSM) by removing the **hsm_info** object associated with file objects from the database. Use this utility for the following situations:

- Encountering errors while migrating files to HSM.
- Maintenance requirement to remove a specific volume from purview of HSM.
- Remove all volumes from the purview of HSM.

After the execution of this utility completes, all physical volume files must be brought back to a primary tier in the event it has already migrated by third-party HSM software to secondary or tertiary tier. Siemens PLM Software recommends sites execute this utility on a specific volume when no other users are accessing Teamcenter, especially during maintenance or upgrades. This utility can take considerable time to reverse migrate all files from the purview of HSM.

SYNTAX

unmigrate_from_hsm [-u=*user-id* {-p=*password* | -pf=*password-file*} -g=*group*] [-listvolumes] [-volume= ALL | *volume-name*] [-v] [-h]

ARGUMENTS

-u

Specifies the user ID.

This is generally **infodba** or another user with administration privileges. If this argument is used without a value, the operating system user name is used.

Note

If Security Services single sign-on (SSO) is enabled for your server, the **-u** and **-p** arguments are authenticated externally through SSO rather than being authenticated against the Teamcenter database. If you do not supply these arguments, the utility attempts to join an existing SSO session. If no session is found, you are prompted to enter a user ID and password.

-p

Specifies the password.

If used without a value, the system assumes a null value.

If this argument is not used, the system assumes the *user-ID* value to be the password.

-pf

Specifies the password file. If used without a value, the system assumes a null value. If this argument is not used, the system assumes the *user-ID* value to be the password.

For more information about managing password files, see [Manage password files](#).

This argument is mutually exclusive with the **-p** argument.

-g

Specifies the group associated with the user.

If used without a value, the user's default group is assumed.

-listvolumes

List all volumes defined for the database.

-volume

Specifies particular volume to unmigrate. The system administrator can specify **volume=ALL** to unmigrate all volumes from the purview of HSM.

-v

Verbose mode. Provides information about results and progress.

-h

Displays help for this utility.

ENVIRONMENT

- The generic command window set with all Teamcenter-related environments.
- As specified in [Manually configuring your environment for Teamcenter utilities](#).

FILES

As specified in [Log files produced by Teamcenter](#).

RESTRICTIONS

This utility is intended only for system-level users.

**RETURN
VALUES**

Return value upon success

0

Return value upon failure

1

EXAMPLES

- To unmigrate all volumes defined for the database:

```
unmigrate_from_hsm -u=infodba -p=infodba -g=dba -volume=ALL -v
```

- To unmigrate a specific volume:

```
unmigrate_from_hsm -u=infodba -p=infodba -g=dba -volume=volume_name
```

- To list volumes names defined for a database:

```
unmigrate_from_hsm -u=infodba -p=infodba -g=dba -listvolumes
```

upgrade_vendor_part

Populates the new typed reference attribute in the **ManufacturerPart** from the existing data during upgrade. The associated vendor for the new typed reference attribute is retrieved from the **TC_Vendor_part_rel** relation.

SYNTAX

upgrade_vendor_part [-u=infodba -p=\$TC_USER_PASSWD} -g=dba -h

ARGUMENTS

-u

Specifies the user ID.

This is generally **infodba** or another user with administration privileges. If this argument is used without a value, the operating system user name is used.

Note

If Security Services single sign-on (SSO) is enabled for your server, the user and password arguments are authenticated externally through SSO rather than being authenticated against the Teamcenter database. If you do not supply these arguments, the utility attempts to join an existing SSO session. If no session is found, you are prompted to enter a user ID and password.

-p

Specifies the password.

-g

Specifies the group associated with the user, in this case, **dba**

-h

Displays help for this utility.

ENVIRONMENT

As specified in [Manually configuring your environment for Teamcenter utilities](#).

FILES

As specified in [Log files produced by Teamcenter](#).

RESTRICTIONS

None.

vm_report

Evaluates any or all volume management policies to generate a report. Because of performance considerations using the rich client application interface, Siemens PLM Software recommends the system administrator execute this utility to create pending migration file sets.

Sites can schedule this utility to execute overnight through a UNIX **cron** job or Microsoft Windows **at** command. This utility can take considerable time to evaluate all migration policies.

SYNTAX

```
vm_report [-u=user-id {-p=password | -pf=password-file} -g=group]
[-listvolumes] [-listpolicies]
[-migrationreport=migration-policy] [-filepath=filepath]
[-migrationreport=ALL {-srcvol=source-volume-name
-destvol=destination-volume-name} -filepath=filepath]
[-before=before-date] [-after=after-date] [-v] [-h]
```

ARGUMENTS**-u**

Specifies the user ID.

This is generally **infodba** or another user with administration privileges. If this argument is used without a value, the operating system user name is used.

Note

If Security Services single sign-on (SSO) is enabled for your server, the **-u** and **-p** arguments are authenticated externally through SSO rather than being authenticated against the Teamcenter database. If you do not supply these arguments, the utility attempts to join an existing SSO session. If no session is found, you are prompted to enter a user ID and password.

-p

Specifies the password.

If used without a value, the system assumes a null value.

If this argument is not used, the system assumes the *user-ID* value to be the password.

-pf

Specifies the password file. If used without a value, the system assumes a null value. If this argument is not used, the system assumes the *user-ID* value to be the password.

For more information about managing password files, see [Manage password files](#).

This argument is mutually exclusive with the **-p** argument.

-g

Specifies the group associated with the user.

If used without a value, the user's default group is assumed.

-listvolumes

Indicates that the utility is to list all volumes in the database.

-listpolicies

Indicates that the utility is to list all active policies.

-migrationreport

Specifies migration policy for which the report need to be reported.

To generate a report on all active migration policies, specify **-migrationreport=ALL**. If you specify **ALL** for the migration policy, you must include either the **-srcvol** or **-destvol** arguments.

-srcvol

Specifies source volume name.

-destvol

Specifies destination volume name.

-before

Specifies the beginning date for reporting migration. For example, **23_Mar_2004**.

-after

Specifies the ending date for reporting migration. For example, **23_May_2006**.

-filepath

Specifies the operating system file path to which the report is saved.

-v

Verbose mode. Provides information about results and progress.

-h

Displays help for this utility.

ENVIRONMENT

- The generic command window set with all Teamcenter-related environments.
- As specified in [Manually configuring your environment for Teamcenter utilities](#).

FILES

As specified in [Log files produced by Teamcenter](#).

RESTRICTIONS

Due to performance considerations, Siemens PLM Software recommends a system administrator execute this utility instead of using the rich client application interface.

EXAMPLES

- Execute the following command to generate report on all policies in the database:

```
vm_report -u=infodba -p=infodba -g=dba -migrationreport=ALL
-srcvol=volume-one -destvol=volume-two -filepath=c:\temp\report.txt
```

- Execute the following command to generate report about a specific policy in the database:

```
vm_report -u=infodba -p=infodba -g=dba
-migrationreport=policy-name -filepath=c:\temp\report.txt
```

- Execute the following command to list of all active policies names defined for a database:

```
vm_report -u=infodba -p=infodba -g=dba -listpolicies
```

xml_validator

Checks the XML file against the document type definition (DTD) to which it should conform.

SYNTAX

xml_validator [-u=*user-id* {-p=*password* | -pf=*password-file*} -g=*group*]
[-v=*always* | *never* | *auto**] *file.xml*

ARGUMENTS**-u**

Specifies the user ID.

This is generally **infodba** or another user with administration privileges. If this argument is used without a value, the operating system user name is used.

Note

If Security Services single sign-on (SSO) is enabled for your server, the **-u** and **-p** arguments are authenticated externally through SSO rather than being authenticated against the Teamcenter database. If you do not supply these arguments, the utility attempts to join an existing SSO session. If no session is found, you are prompted to enter a user ID and password.

-p

Specifies the password.

If used without a value, the system assumes a null value.

If this argument is not used, the system assumes the *user-ID* value to be the password.

-pf

Specifies the password file. If used without a value, the system assumes a null value. If this argument is not used, the system assumes the *user-ID* value to be the password.

For more information about managing password files, see [Manage password files](#).

This argument is mutually exclusive with the **-p** argument.

-g

Specifies the group associated with the user.

If used without a value, the user's default group is assumed.

-v

Specifies the validation scheme: **always**, **never**, **auto***. If not explicitly stated, defaults to **auto***.

file .xml

Specifies the XML file to be validated.

-h

Displays help for this utility.

ENVIRONMENT

As specified in [Manually configuring your environment for Teamcenter utilities](#).

FILES

As specified in *Log files produced by Teamcenter*.

EXAMPLES

1. The following example checks only the structure of the XML file:

```
xml_validator -v=never file1.xml
```

2. This example produces an error if the XML file does not have an associated DTD file. It always checks the validity of the XML file.

```
xml_validator -v=always file1.xml
```

3. This example checks the structure of the XML file if it does not correspond to a DTD. If the file corresponds to a DTD, it checks the validity of the file's XML against that of the DTD.

```
xml_validator file1.xml
```

File Management System (FMS) utilities

fscadmin.sh/.bat

Monitors and controls File Management System FSC servers. This can be used to check the status of a server, perform a shutdown, modify logging levels, query performance counters, or to clear or inspect caches.

SYNTAX

```
$FMS_HOME/fscadmin.sh [-u=user-id {-p=password |  
-pf=password-file} -g=group]  
[-h] [-k keyfile] [-s serveraddr] [-f tickets-file] [command]
```

ARGUMENTS**-u**

Specifies the user ID.

This is generally **infodba** or another user with administration privileges. If this argument is used without a value, the operating system user name is used.

Note

If Security Services single sign-on (SSO) is enabled for your server, the **-u** and **-p** arguments are authenticated externally through SSO rather than being authenticated against the Teamcenter database. If you do not supply these arguments, the utility attempts to join an existing SSO session. If no session is found, you are prompted to enter a user ID and password.

-p

Specifies the password.

If used without a value, the system assumes a null value.

If this argument is not used, the system assumes the *user-ID* value to be the password.

This argument is mutually exclusive with the **-pf** argument.

-pf

Specifies the password file. If used without a value, the system assumes a null value. If this argument is not used, the system assumes the *user-ID* value to be the password.

For more information about managing password files, see [Manage password files](#).

This argument is mutually exclusive with the **-p** argument.

-g

Specifies the group associated with the user.

If used without a value, the user's default group is assumed.

-h

Displays help for this utility.

-k

Specifies a file containing the encryption key required by this system. A key file is a text file containing an ASCII-HEX encryption key. This is generally the same key file referenced in the **fmsmaster.xml** file for this system.

This argument is optional.

Example: **fscadmin -k site123keyfile.txt ./status**
 Default: **<none>**

-s

Specifies the protocol server for the FSC and the port you wish to communicate with.

Example: **fscadmin -s http://myserver:4445 ./status**
 Default: **http://127.0.0.1:4444**

-f

Specifies the name of the tickets file.

command

Command is a formatted string with the following fields:

FSCID/FUNCTION[/SUBFUNCTION/...]

FSCID

The FSC with the given ID, as defined in the master configuration, for which this command is intended. A period (.) can be used to indicate the local (current) FSC you are connecting to, as indicated by the **-s** parameter.

FUNCTION

[/SUBFUNCTION/...]

The functions and subfunctions are enumerated in [Function\[/Subfunction/...\] Details](#), later in this section.

Example: **fscadmin -s http://myserver:4445 ./status**
 Example: **fscadmin -s http://myserver:4445 ./log**
 Default: **./status**.

Example: **fscadmin -s http://myserver:4444 fsc123/log**

ENVIRONMENT

- The **FSC_HOME** variable must be set to a valid FMS server directory.
- The **JAVA_HOME** variable must be set to a valid Java SDK directory.

FILES

- **fscadmin.properties**

(Optional) A property file used to configure proxy and SSL options if required by the **fscadmin** utility.

- **fscadmin.properties.template**

A template of the available properties that can be set for the **fscadmin** utility.

RESTRICTIONS

- You can route **fscadmin** commands to remote FSCs without having to specify that FSCs address on the command line.
- Input and output through the **fscadmin** utility is not localized. Commands are subject to change without notice.

FUNCTION[/SUBFUNCTION/...]
DETAILS

FUNCTION[/SUBFUNCTION]	Description
cachesummary	Summary of the read and write caches (number of files, bytes, hits, misses).
cachesummary/read	Summary of the read cache.
cachesummary/write	Summary of the write cache.
cachesummary/whole	Summary of the whole file cache.
cachedetail	Summary and detail of the read and write caches (GUIDS, filesizes).
cachedetail/read	Summary and detail of the read cache.
cachedetail/write	Summary and detail of the write cache.
cachedetail/whole	Summary and detail of the whole file cache.
clearcache	Clears (empties) both read and write caches.
clearcache/read	Clears the read cache.
clearcache/write	Clears the write cache.
clearcache/whole	Clears the whole file cache.
config	Dumps the configuration for this FSC.
config/hash	Displays the MD5 hash for the current master configuration file.
config/reload	Reloads the latest configuration from the usual configuration sources, either from disk or via download from another FSC.
config/reload/all	Performs a coordinated configuration reload first reloading all FSC configuration masters, then reloading all FSC configuration slaves. Returns final configuration hash results in a comma-separated list.
config/reload/slaves	Performs a coordinated configuration reload of all FSC configuration slaves. Returns final configuration hash results in a comma-separated list.
config/report	Queries the configuration hashes from all FSCs and returns results in a comma-separated list.
filestoresummary	Summary of all filestores (volumeid, root path, files, dirs, bytes).
filestoresummary/xxx	Summary of a filestore specified by <i>xxx</i> .
filestoredetail	Summary and detail of all filestores (file name, length, last modified, last accessed).
filestoredetail/xxx	Summary and detail of a filestore specified by <i>xxx</i> .

FUNCTION[/SUBFUNCTION]	Description
loadfscache/status	Displays the simple Load FSC Cache status. This includes the number of tickets received, valid, invalid, expired, etc. It also includes the number of files that have been downloaded, failed to download, and are waiting to be downloaded.
loadfscache/status/reset	Resets the Load FSC Cache statistics.
loadfscache/queue	Displays the tickets remaining in the Load FSC Cache queue.
loadfscache/queue/clear	Removes all remaining tickets in the Load FSC Cache queue.
loadfscache/filelist	Uses -f option to upload a file of tickets that should be populated into the specified FSC.
loadfscache/stat	Returns the number of transactions processed by the Load FSC Cache context.
loadfscache/stat/valid	Returns the number of valid tickets processed.
loadfscache/stat/invalid	Returns the number of invalid tickets processed.
loadfscache/stat/expired	Returns the number of expired tickets processed.
loadfscache/stat/total	Returns the total number of tickets processed.
loadfscache/stat/ok	Returns the number of files successfully populated.
loadfscache/stat/failed	Returns the number of files which failed to populate.
loadfscache/stat/waiting	Returns the number of tickets remaining to be populated
loadfscache/stat/processing	Returns 1 if population is occurring, otherwise 0.
loadfscache/stat/localfiles	Returns the number of local files on which population was attempted, but was not required.
loglevel/show	Shows the current logging level of all loggers.
loglevel/[fatal error warn info debug]	Sets the log level for all loggers starting with com to <i>xxx</i> .

FUNCTION[/SUBFUNCTION]	Description
loglevel / <i>logger</i> /[fatal error warn info debug]	Sets error log level on the logger and all children. <i>logger</i> is in the form of the name of the class.package of which you wish to change the log level. com.teamcenter.fms.servercache is all of the server com.teamcenter.fms.servercache. FileHandleCacheManager sets only the loglevel on that class.
log	Dumps the current logfile contents.
purgecache	Purges the caches, reclaiming disk space.
purgecache/read	Purges the read cache.
purgecache/write	Purges the write cache.
purgecache/whole	Purges the whole file cache.
purgeguid / <i>xxx</i>	Removes the GUID (file) specified by <i>xxx</i> from the cache.
perfcounters	Performance counters
perfcounters/reset	Resets the performance counters.
shutdown	Stops the FSC when it becomes idle (when all current file transfers are complete). The FSC will reject all new incoming file requests.
shutdown/maxwait / <i>xxx</i>	Stops the FSC when it becomes idle (when all current file transfers are complete) or when a maximum number of seconds have been exceeded. The FSC will reject all new incoming file requests. If the server does not become idle, <i>xxx</i> is the number of seconds to wait before forcing the shutdown.
shutdown/now	Stops the FSC.
status	Displays simple status about the FSC (FSCID, site, running time) and also prints the number of concurrent admin and file-based connections. This also shows the remaining time before a forced shutdown, if one is pending.
stop	Stops the FSC.
useragents	Prints a tally of all the useragents (clients) that have connected.
version	Prints the versions of the FSC JAR files.

EXAMPLES

- To see general statistics of the server, enter the following command:


```
./fscadmin.sh -s http://myserver:4445 ./status
```

Example output:

```
FSC id: myfsc, site:fms.teamcenter.com
running for 3 days, 17 hours, 3 min, 46 sec
Current number of file connections: 0
Current number of admin connections: 1
```

- To see general statistics of the caches, enter the following command:

```
./fscadmin.sh -s http://myserver:4445 ./cachesummary
```

Example output:

```
Cache summary: myfsc-FSCReadMap
Files: 0, Bytes: 0, Hits: 0, Misses: 0
Cache summary: myfsc-FSCWriteMap
Files: 0, Bytes: 0, Hits: 0, Misses: 0
```

- To see version information for the server, enter the following command:

```
./fscadmin.sh -s http://myserver:4445 ./version
```

Example output:

```
FMSServerCache version: 1.1, build date: 20050729
FMSUtil version: 1.1, build date: 20050729
FSCJavaClientProxy version: 1.1, build date: 20050729
```

- To set the FSC loglevel to **WARN**, enter the following command on a single line:

```
./fscadmin.sh -s http://myserver:4445 ./loglevel/warn
```

Example output:

```
loglevel done.
```

- To cause an idle shutdown, waiting no more than 1 hour, enter the following command on a single line:

```
./fscadmin.sh -s http://myserver:4445 ./shutdown/maxwait/3600
```

Example output:

```
FSC will shutdown when idle, or in 3600 seconds...
```

- The following command prepopulates FSC using the **loadfscache/filelist** command:

```
fscadmin -s http://server:port -f ticketsfile ./loadfscache/filelist
```

FSCWholeFileCacheUtil

Purges files from the whole file cache. Purge the cache based on file age, subdirectory size, and/or available disk space. This utility also purges misplaced files.

By default, this utility is stored in the *FSC_HOME/bin* directory.

SYNTAX

FSCWholeFileCacheUtil **{-clear | -purge}** [**-d=cache-root-directory-path**]
[**-l=file-name**] [**-maxf=maximum-files-per-subdirectory**]
[**-maxage=maximum-file-age**] [**-pctfree=percent-disk-free**]
[**-temp=temporary-cache-directory-path**] [**-h**]

ARGUMENTS**-clear**

Clears all permanent cache files from the cache.

-purge

Performs a single purge cycle on the cache.

-d

Specifies the version subdirectory path of the cache. If not set, the current directory is used.

Use with either the **-clear** or **-purge** arguments.

-l

Specifies the full path name of the log file. If not set, no logging is performed.

Use with either the **-clear** or **-purge** arguments.

-maxf

Specifies the maximum number of files per subdirectory. If the number of files stored in the subdirectory exceeds the configured amount, files are automatically purged to the maximum number of files specified.

Valid values are **1024** through **10240**. The default setting is **10240**.

Use only with the **-purge** argument.

-maxage

Specifies (in days) the maximum file age. All cache files older than the specified age are purged.

Valid values are **0** through **3650**. The default setting is **3650**. Setting this argument to **0** clears the entire cache.

Use only with the **-purge** argument.

-pctfree

Specifies the minimum percentage of disk space to remain free.

Valid values are **0** through **100**. The default setting is **0**. Setting this argument to **0** purges the cache by only the subdirectory count specified by the **-maxf** argument and the file age specified by the **-maxage** argument. Setting this argument to **100** clears the entire cache.

Use only with the **-purge** argument.

-temp

Specifies the directory location to store temporary cache purge data. The default setting is the directory set by the **-d** argument.

Use only with the **-purge** argument.

-h

Displays help for this utility.

ENVIRONMENT

As specified in *Manually configuring your environment for Teamcenter utilities*.

FILES

As specified in *Log files produced by Teamcenter*.

RESTRICTIONS

None.

fccstat

Monitors the local FMS cache (FCC).

By default, this utility is stored in the *TC_ROOT/tccs/bin* directory.

SYNTAX

\$FMS_HOME/fccstat [-u=*user-id* {-p=*password* | -pf=*password-file*} -g=*group*]
[-x] [-status] [-config] [-reconfig] [-purge] [-clear] [-restart]
[-start] [-stop] [-kill]

ARGUMENTS**-u**

Specifies the user ID.

This is generally **infodba** or another user with administration privileges. If this argument is used without a value, the operating system user name is used.

Note

If Security Services single sign-on (SSO) is enabled for your server, the **-u** and **-p** arguments are authenticated externally through SSO rather than being authenticated against the Teamcenter database. If you do not supply these arguments, the utility attempts to join an existing SSO session. If no session is found, you are prompted to enter a user ID and password.

-p

Specifies the password.

If used without a value, the system assumes a null value.

If this argument is not used, the system assumes the *user-ID* value to be the password.

This argument is mutually exclusive with the **-pf** argument.

-pf

Specifies the password file. If used without a value, the system assumes a null value. If this argument is not used, the system assumes the *user-ID* value to be the password.

For more information about managing password files, see [Manage password files](#).

This argument is mutually exclusive with the **-p** argument.

-g

Specifies the group associated with the user.

If used without a value, the user's default group is assumed.

none

Prints whether FCC is online, displays FCC versions and run time if running.

-help or -h or -?

Displays help for this utility. Help can be localized if the FCC is running.

-x

Prints FCC cache statistics summary, including whole file read, whole file write, and segment cache statistics. In addition, this option displays all offline FSC connections.

Offline FSC connections are those that have been attempted and failed but have not yet been restored to service.

-status

Prints FCC status and statistics summary, including whole file read, whole file write, and segment cache statistics, as well as client request statistics, FSC upload and download statistics, and the currently active assigned FSC. Assigned FSCs are listed as active by default, even if they have never been used. FSC addresses that have been attempted and failed, but have not yet been restored to service, are reported as offline.

-config

Displays the name of the local FCC configuration file used for bootstrapping.

-purge

Purges all files from the FCC cache, including the segment cache extent files.

-clear

Purges the cache completely. This removes all data but retains the segment cache extent files.

-reconfig

Reloads the FCC configuration. Use this option to update a running FCC with changes made to the local FCC XML configuration files.

-restart

Stops (if running) and restarts the FCC, effectively reloading the configuration. Environment variables still override any configuration file settings or changes.

-start

Starts the FCC if it is not already started.

-stop

Shuts down the FCC process immediately if no other clients are connected or if all connected clients are idle. Otherwise, a warning message is displayed.

-kill

Immediately and unconditionally shuts down the FCC process.

ENVIRONMENT

As specified in [Manually configuring your environment for Teamcenter utilities](#) and the following:

- The **FMS_HOME** variable must be set to a valid FMS server directory.
- The **JAVA_HOME** variable must be set to a valid Java SDK directory.
- FCC must be running.
- This command must be run in an FMS operating environment.

FILES

As specified in [Log files produced by Teamcenter](#).

RESTRICTIONS

None.

EXAMPLES

- To determine whether the FCC is running, run the **fccstat** command. For example:

```
C:\Tc2007\portal\fms\bin>fccstat
```

Example output:

```
fccstat: FCC offline.
```

- To display help for this utility and a statistic summary, run the **fccstat** command with the **status** argument. For example:

```
C:\Tc2007\portal\fms\bin>fccstat -? -x -status fccstat -?:
```

Example output:

```
FMS Client Cache Status Utility
fccstat command line options:
  no options      Display FCC component versions and execution time.
  -help           Display this help message.
  -h              Display this help message.
  -?              Display this help message.
  -x              Display FCC status summary.
  -status         Display complete FCC status.
  -config         Display FCC configuration filename.
  -purge          Clear the FCC Cache and remove extents.
  -clear          Clear the FCC Cache and retain extents.
  -remove (UID)   Remove a UID from the FCC Cache.
  -restart        Stop and restart the FCC.
  -stop           Stop the FCC if no clients are connected.
  -kill           Stop the FCC even if clients are connected.

fccstat -x:
Cache:
  segment: 1 files, 507904 bytes.
  read:    38 files, 268433410 bytes.
  write:   0 files, 0 bytes.
Servers:
  0 files downloaded.
  Active assigned FSC is 'http://127.0.0.1:4444/'
fccstat -status:
Cache:
  segment: 1 files, 507904 bytes, 0 hits, 0 misses.
  read:    38 files, 268433410 bytes, 0 hits, 0 misses.
  write:   0 files, 0 bytes.
Clients:
  3 Client connections established.
  6 Client request messages processed.
  5 Client response messages processed.
  0 Client status messages processed.
  0 Client error messages processed.
Servers:
  0 segments downloaded.
  0 files downloaded.
  0 files uploaded.
  Active assigned FSC is 'http://127.0.0.1:4444/'
```

- To display the name of the local FCC configuration file and clear the cache, run the **fccstat** command with the **clear** argument. For example:

```
C:\Tc2007\portal\fms\bin>fccstat -config -clear
```

Example output:

```
fccstat -config:
```

```
Z:\pkmvob1\fms\build\install\fms\fcc.xml
fccstat -clear:
Cache cleared.
```

- To purge a UID from the cache, run the **fccstat** command with the **remove** argument. For example:

```
C:\Tc2007\portal\fms\bin>fccstat
-remove abcd1234fedc9876ef005678ba005432
```

Example output:

```
fccstat -remove:
UID abcd1234fedc9876ef005678ba005432 has been removed from the FCC Cache.
```

- To stop the FCC, run the **fccstat** command with the **stop** argument. For example:

```
C:\Tc2007\portal\fms\bin>fccstat -stop
```

Example output:

```
fccstat -stop:
FCC Stopped.
```

- To restart the FCC, run the **fccstat** command with the **restart** argument. For example:

```
C:\Tc2007\portal\fms\bin>fccstat -restart
```

Example output:

```
fccstat -restart:
FCC Started.
```

- To shut down the FCC process immediately when no clients are connected or all clients are idle, run the **fccstat** command with the **stop** argument. For example:

```
C:\Tc2007\portal\fms\bin>fccstat -stop
```

Example output:

```
fccstat -stop:
FCC Stopped.
```

- To kill the FCC, run the **fccstat** command with the **kill** argument. For example:

```
C:\Tc2007\portal\fms\bin>fccstat -kill
```

Example output:

```
fccstat -kill:
FCC Stopped.
```

- To print an FCC status and statistics summary, including whole file read, whole file write, and segment cache statistics, as well as client request statistics, FSC

upload and download statistics, and the currently assigned FSC, run the **fccstat** command with the **status** argument. For example:

```
C:\Tc2007\portal\fms\bin>fccstat -status
```

Example output:

```
fccstat -status:
Cache:
  segment: 19 files, 52854784 bytes, 590722 hits, 1781 misses.
  read:    19 files, 134216705 bytes, 106253 hits, 38 misses.
  write:   0 files, 0 bytes.
Clients:
  5 Client connections established.
  388741 Client request messages processed.
  392284 Client response messages processed.
  79 Client status messages processed.
  1 Client error messages processed.
Servers:
  839 segments downloaded.
  19 files downloaded.
  0 files uploaded.
  Active assigned FSC is 'http://127.0.0.1:4444/'
```

- Entering **fccstat -reconfig** and the operation is successful returns the following message:

```
FCC successfully reconfigured.
```

- Entering **fccstat -reconfig** and the operation is not successful returns the following message:

```
FCC reconfiguration failed:
  Cannot reconfigure FCC pipe name.
```

install_encryptionkeys

Creates encryption keys for the File Management System (FMS). This utility is initially invoked by the Teamcenter installation procedure to install the predefined encryption keys. After the initial installation, this utility can be used to list, modify, and delete encryption keys.

SYNTAX

install_encryptionkeys [-u=*user-id* {-p=*password* | -pf=*password-file*} -g=*group*] -f=install | list | modify | delete | install_mediator_key [-h]

ARGUMENTS

-u

Specifies the user ID.

This is generally **infodba** or another user with administration privileges. If this argument is used without a value, the operating system user name is used.

Note

If Security Services single sign-on (SSO) is enabled for your server, the **-u** and **-p** arguments are authenticated externally through SSO rather than being authenticated against the Teamcenter database. If you do not supply these arguments, the utility attempts to join an existing SSO session. If no session is found, you are prompted to enter a user ID and password.

-p

Specifies the password.

If used without a value, the system assumes a null value.

If this argument is not used, the system assumes the *user-ID* value to be the password.

This argument is mutually exclusive with the **-pf** argument.

-pf

Specifies the password file. If used without a value, the system assumes a null value. If this argument is not used, the system assumes the *user-ID* value to be the password.

For more information about managing password files, see [Manage password files](#).

This argument is mutually exclusive with the **-p** argument.

-g

Specifies the group associated with the user.

If used without a value, the user's default group is assumed.

-f

Specifies one of the following functions:

install	Installs the default encryption keys in the database.
list	Lists the encryption keys currently installed in the database.
modify	Modifies one or more encryption keys in the database.

delete	Deletes the encryption keys from the database.
install_mediator_key	Prompts the user to enter the Teamcenter Security Services Mediator Password Value key. The key is installed in the EncryptionKey table.
-h	Displays help for this utility.

ENVIRONMENT

As specified in *Manually configuring your environment for Teamcenter utilities*.

FILES

As specified in *Log files produced by Teamcenter*.

RESTRICTIONS

None.

EXAMPLES

- To install the default encryption keys, enter the following command on a single line:

```
install_encryptionkeys -u=infodba -p=password -g=dba -f=install
```

load_fccache

Prepopulates the FMS client cache (FCC) using ITK calls, resulting in improved cache performance. You can also use this utility to read the PLM XML file generated by the [bomwriter](#) utility, and then parse the UIDs and generate read tickets. The read tickets can be prepopulated on the target FCC process. Additional options include:

- Repopulate the FCC based on dataset type.
- Manage integrated clash management (ICM) behavior.
- Copy JT files to a local staging area.
- Update the PLM XML file to point to local JT files.

SYNTAX

```
load_fccache [-u=user-id {-p=password | -pf=password-file} -g=group]
-f=list [-plmxml=file-name] [-dataset_type=dataset-type-name]
-f=load [-plmxml=file-name] [-input_file=file-name] [-dataset_type=dataset-type-name]
[-latest_version=yes | no] [-output_file=file-name] [-copy_out=directory]
[-output_plmxml=file-name] [-use_absolute_location=yes | no]
[-config=configuration-file] [-log_filename=log-file-name] [-log_types=levels]
[-copy_out_lifetime=value] [-lifetime_check=yes | no]
[-lifetime_check_interval=value] [-lifetime_process_limit=value] [-purge] [-h]
```

ARGUMENTS

-u

Specifies the user ID.

This is generally **infodba** or another user with administration privileges. If this argument is used without a value, the operating system user name is used.

Note

If Security Services single sign-on (SSO) is enabled for your server, the **-u** and **-p** arguments are authenticated externally through SSO rather than being authenticated against the Teamcenter database. If you do not supply these arguments, the utility attempts to join an existing SSO session. If no session is found, you are prompted to enter a user ID and password.

-p

Specifies the password.

If used without a value, the system assumes a null value.

If this argument is not used, the system assumes the *user-ID* value to be the password.

This argument is mutually exclusive with the **-pf** argument.

-pf

Specifies the password file. If used without a value, the system assumes a null value. If this argument is not used, the system assumes the *user-ID* value to be the password.

For more information about managing password files, see [Manage password files](#).

This argument is mutually exclusive with the **-p** argument.

-g

Specifies the group associated with the user.

If used without a value, the user's default group is assumed.

-f

Runs either the **list** or **load** function.

The **list** function lists specified UIDs, which can then be used to prepopulate the FCC using the **load** function. The **list** function accepts the following input:

- **-plmxml=file-name**
Lists the UIDs of all datasets in the PLM XML file generated by the **bomwriter** utility.
- **-dataset_type=dataset-type-name**
Lists the UIDs of all datasets of the specified dataset type.

The **load** function prepopulates the FCC as specified by the following input:

- **-plmxml=file-name**
Prepopulates the FCC with all the datasets in the PLM XML file generated by the **bomwriter** utility.
- **-input_file=file-name**
Prepopulates the FCC based on the UIDs previously generated by the **list** function.
- **-dataset_type=dataset-type-name**
Prepopulates the FCC with all datasets of the specified dataset type.

-latest_version

Use **yes** to prepopulate the FCC with only the latest dataset version or **no** to use the version specified by the UID. The default value is **yes**.

Use this argument with the **load** function.

-output_file

Direct the output to a specific file by specifying a file name. If this argument is not used, the default output file is **stdout**.

-copy_out

Copy each downloaded file to a specific directory by specifying the directory name.

Use this argument with the **load** function.

-output_plmxml

If the **load** function is set to prepopulate the FCC with a PLM XML file, and the **-copy_out** argument is used to specify a specific directory location to which copies of each downloaded file is loaded, and then use this argument to update the PLM XML with the same directory location as specified by the **-copy_out** argument.

You must use this argument with the **load** function and the **-copy_out** argument.

-use_absolute_location

Use **yes** to create absolute location references in the output PLM XML file. For more information about configuring default settings, use **-h -config**.

-config

Specifies the file containing default settings. For more information about configuring default settings, use **-h -config**.

-log_filename

Specifies the file to which logging messages are printed. If not set, the default location is the **stderr** file. For more information about configuring logging messages, use **-h -log_types**.

-log_types

Specifies which types of messages are logged. Valid values are:

NONE

ERROR

WARNING

INFORMATION

DEBUG

PERFORMANCE

ALL

Combine log types using the plus (+) symbol as a delimiter. For example:

ERROR+WARNING+INFORMATION

For more information about configuring logging messages, use **-h -log_types**.

-copy_out_lifetime

Specifies the lifetime date of the files stored in the directory specified by the **-copy_out** directory. The directory is scanned for files that are older than the specified lifetime date. Use this argument to automatically clean up older files that are no longer used. This argument must be used in conjunction with the **-copy_out** argument.

Use **-h -config** to display more information about configuring default settings.

-lifetime_check

Use **yes** to scan the directory specified by the **-copy_out** directory during startup and perform a lifetime check. The lifetime check is performed in random order. This argument must be used in conjunction with the **-copy_out** and **-copy_out_lifetime** arguments.

Use **-h -config** to display more information about configuring default settings.

-lifetime_check_interval

Specifies how frequently to scan the directory specified by the **-copy_out** directory and perform a random cleanup based on the interval value. For example, if set to **10**, a lifetime check is performed once every ten executions.

This argument must be used in conjunction with the **-copy_out**, **-copy_out_lifetime** and **-lifetime_check** arguments. If the directory specified by the **-copy_out** argument contains many files, and if it is not important to check the lifetime on each execution, setting the **-lifetime_check_interval** argument can improve performance.

Use **-h -config** to display more information about configuring default settings.

-lifetime_process_limit

Specifies how long (in seconds) lifetime processing can continue. The lifetime check is performed in random order. If the directory specified by the **-copy_out** directory contains a large number of files, lifetime processing can be lengthy. You can use this argument to randomly process a subset of the files by setting a low value. This argument must be used in conjunction with the **-copy_out**, **-copy_out_lifetime** and **-lifetime_check** arguments.

Use **-h -config** to display more information about configuring default settings.

-purge

Purges all files from the directory specified by the **-copy_out** directory.

-h

Displays help for this utility. Use **-h -config** to display more information about configuring default settings. Use **-h -log_types** for more information about logging output.

ENVIRONMENT

As specified in [Manually configuring your environment for Teamcenter utilities](#).

FILES

As specified in [Log files produced by Teamcenter](#).

RESTRICTIONS

None.

EXAMPLES

To ensure the JT files referenced in the **xyz001.plmxml** file are using the same dataset versions as when it was generated (rather than the latest version), enter the following command on a single line:

```
load_fccache -u=infodba -p=password -f=load -plmxml=xyz001.plmxml
-latest_version=no -copy_out=plmxml_cache -output_plmxml=xyz001_local.plmxml
```

load_fscache

Reads a PLM XML file, parses it for globally unique identifiers (GUIDs) and generates read tickets for files referenced in the PLM XML file. The read tickets can be stored in the operating system (OS) file. The utility can also be used to load the file server cache (FSC) of a target/distant FSC serving the same database using the read tickets file.

SYNTAX

```
load_fscache [-u=user-id {-p=password | -pf=password-file} -g=group]
{ [-f=list {-plmxml=file-name | -dataset_type=dataset-type-name} ] |
[-f=load -fsctargets=string { [-plmxml=file-name] | [-input_file=file-name] |
[-dataset_type=dataset-type-name] } } [-latest_version= yes | no]
[-output_file=file-name] [[-config=file-name][-log_filename=file-name]
[-log_types=log-level] [-h[-config] [-log_types] ]
```

ARGUMENTS

-u

Specifies the user ID.

This is generally **infodba** or another user with administration privileges. If this argument is used without a value, the operating system user name is used.

Note

If Security Services single sign-on (SSO) is enabled for your server, the **-u** and **-p** arguments are authenticated externally through SSO rather than being authenticated against the Teamcenter database. If you do not supply these arguments, the utility attempts to join an existing SSO session. If no session is found, you are prompted to enter a user ID and password.

-p

Specifies the password.

If used without a value, the system assumes a null value.

If this argument is not used, the system assumes the *user-ID* value to be the password.

This argument is mutually exclusive with the **-pf** argument.

-pf

Specifies the password file. If used without a value, the system assumes a null value. If this argument is not used, the system assumes the *user-ID* value to be the password.

For more information about managing password files, see [Manage password files](#).

This argument is mutually exclusive with the **-p** argument.

-g

Specifies the group associated with the user.

If used without a value, the user's default group is assumed.

-f

Runs either the **list** or **load** function.

The **list** function creates a list of GUIDs for object in the specified PLM XML file, which can then be used to prepopulate the FSC using the **load** function. The **list** function requires one of the following input arguments:

- **-plmxml=***file-name*
Lists the GUIDs of all datasets in the specified PLM XML file.
- **-dataset_type=***dataset-type-name*
Lists the GUIDs of all datasets of the specified dataset type.

The **load** function prepopulates the FSCs specified in the **-fsctargets** argument using the information provided by one of the following input arguments:

- **-plmxml=***file-name*
Prepopulates the FSC with all the datasets in the specified PLM XML file.
- **-input_file=***file-name*
Prepopulates the FSC based on the GUIDs previously generated by the **list** function.
- **-dataset_type=***dataset-type-name*
Prepopulates the FSC with all datasets of the specified dataset type.

-latest_version

Specifies the dataset version to use to prepopulate the FSC. Set to **yes** to send the latest dataset version or **no** to use the version specified by the GUID. The default value is **yes**.

Use this argument with the **load** function.

-output_file

Directs the output to a specific file by specifying a file name. If this argument is not used, the default output file is **stdout**.

-config

Specifies the file containing default settings. For more information about configuring default settings, use the **-h** **-config** arguments.

-log_filename

Specifies the file to which logging messages are printed. If not set, the default location is the **stderr** file. To display information about configuring logging messages, use the **-h** and **-log_types** arguments together.

-log_types

Specifies which types of messages are logged. Valid values are:

NONE

ERROR

WARNING

INFORMATION

DEBUG**PERFORMANCE****ALL**

Combine log types using the plus (+) symbol as a delimiter. For example:

```
ERROR+WARNING+INFORMATION
```

To display information about configuring logging messages, use the **-h** and **-log_types** arguments together.

-h

Displays help for this utility. Use the **-h** and **-config** arguments together to display more information about configuring default settings. Use **-h** and **-log_types** arguments together to display information about logging output.

ENVIRONMENT

As specified in [Manually configuring your environment for Teamcenter utilities](#).

FILES

As specified in [Log files produced by Teamcenter](#).

RESTRICTIONS

None.

EXAMPLES

- To display a list of GUIDs for datasets in the **abc001.xml** file:

```
load_fsccache -u=infodba -p=password -g=dba -f=list -plmxml=abc001.xml
```

- To prepopulate the FSC for **cvg001** (identified with **fscid fsc_cvg001_infodba**) with the datasets in the **abc001.xml** file:

```
load_fsccache -u=infodba -p=password -g=dba -f=load -plmxml=abc001.xml
-fsctargets=fsc_cvg001_infodba
```

- To create a **readtickets.txt** file containing the GUIDs of the datasets in the **abc001.xml** file:

```
load_fsccache -u=infodba -p=password -g=dba -f=list -plmxml=abc001.xml
-output_file=readtickets.txt
```

- To prepopulate the FSC for **cvg001** (identified with **fscid fsc_cvg001_infodba**) with the datasets in the **readtickets.txt** file:

```
load_fsccache -u=infodba -p=password -g=dba -fsctargets=fsc_cvg001_infodba
-f=load -input_file=readtickets.txt
```

- To create a **readtickets.txt** file containing the GUIDs of all the **CAEAnalysisDS** datasets:

```
load_fsccache -u=infodba -p=password -g=dba -f=list
-dataset_type=CAEAnalysisDS -output_file=readtickets.txt
```

- To prepopulate the FSC for **cvg001** (identified with **fscid fsc_cvg001_infodba**) with the datasets in the **abc001.xml** file and store all types of log messages to the **load.out** file:

```
load_fsccache -u=infodba -p=password -g=dba -fsctargets=fsc_cvg001_infodba
-f=load -plmxml=abc001.xml -log_types=ALL -log_filename=c:\temp\load.out
```

Chapter

23 *Systems Engineering utilities*

add_req_templates

Installs the default Microsoft Word and Excel requirements management templates.

Note This utility usually runs during the installation and upgrade processes and does not need to be run manually. However, if the installation or upgrade does not install the templates, you can run this utility manually.

SYNTAX

```
add_req_templates [-u=user-id] [-p=password | -pf=password-file] [-g=group]  
[-f=path-of-folder -t={SpecTemplate | ObjectTemplate | ExcelTemplate}]  
[-i=path-of-file]  
[-h]
```

ARGUMENTS**-u**

Specifies the user ID.

This is generally **infodba** or another user with administration privileges. If this argument is used without a value, the operating system user name is used.

Note If Security Services single sign-on (SSO) is enabled for your server, the **-u** and **-p** arguments are authenticated externally through SSO rather than being authenticated against the Teamcenter database. If you do not supply these arguments, the utility attempts to join an existing SSO session. If no session is found, you are prompted to enter a user ID and password.

-p

Specifies the password.

If used without a value, the system assumes a null value.

If this argument is not used, the system assumes the *user-ID* value to be the password.

This argument is mutually exclusive with the **-pf** argument.

-pf

Specifies the password file. If used without a value, the system assumes a null value. If this argument is not used, the system assumes the *user-ID* value to be the password.

For more information about managing password files, see [Manage password files](#).

This argument is mutually exclusive with the **-p** argument.

-g

Specifies the group associated with the user.

If used without a value, the user's default group is assumed.

-f

Specifies the path of the folder from which the templates are imported. Use this argument to import multiple templates from the same folder.

If you use this argument, you must also use the **-t** argument.

-t

Specifies the template type being imported. It is either **SpecTemplate**, **ObjectTemplate**, or **ExcelTemplate**.

If you use this argument, you must also use the **-f** argument.

-i

Specifies the path of the template file to be imported. Use this argument to import a single template.

-h

Displays help for this utility.

ENVIRONMENT

As specified in [Manually configuring your environment for Teamcenter utilities](#).

FILES

As specified in [Log files produced by Teamcenter](#).

RESTRICTIONS

- To use this utility, you must be a user with system administration privileges or be granted authorization by a user with system administration privileges.
- The **-f** and **-t** arguments must be used together if they are used.

add_se_templates

Installs the system engineering templates, diagramming templates, and Visio templates required for Systems Engineering.

Note This utility usually runs during the installation and upgrade processes and does not need to be run manually. However, if the installation or upgrade does not install the templates, you can run this utility manually.

SYNTAX

add_se_templates [-u=*user-id*] [-p=*password* | -pf=*password-file*] [-g=*group*]
-dir=*templates-directory*
[-h]

ARGUMENTS

-u

Specifies the user ID.

This is generally **infodba** or another user with administration privileges. If this argument is used without a value, the operating system user name is used.

Note If Security Services single sign-on (SSO) is enabled for your server, the **-u** and **-p** arguments are authenticated externally through SSO rather than being authenticated against the Teamcenter database. If you do not supply these arguments, the utility attempts to join an existing SSO session. If no session is found, you are prompted to enter a user ID and password.

-p

Specifies the password.

If used without a value, the system assumes a null value.

If this argument is not used, the system assumes the *user-ID* value to be the password.

This argument is mutually exclusive with the **-pf** argument.

-pf

Specifies the password file. If used without a value, the system assumes a null value. If this argument is not used, the system assumes the *user-ID* value to be the password.

For more information about managing password files, see [Manage password files](#).

This argument is mutually exclusive with the **-p** argument.

-g

Specifies the group associated with the user.

If used without a value, the user's default group is assumed.

-dir

Specifies the directory containing the Systems Engineering templates; for example, **\$(TC_INSTALL_DIR)/systemsengineering**.

-h

Displays help for this utility.

ENVIRONMENT

As specified in [*Manually configuring your environment for Teamcenter utilities.*](#)

FILES

As specified in [*Log files produced by Teamcenter.*](#)

RESTRICTIONS

To use this utility, you must be a user with system administration privileges or be granted authorization by a user with system administration privileges.

req_convert_to_plaintext

Removes rich text, such as images, URLs, and OLEs, from requirements and converts the body text into plain text. It also applies the content type as plain text to the requirement revision.

SYNTAX

```
req_convert_to_plaintext [-u=user-id] [-p=password |  
-pf=password-file] [-g=group]  
-i=specification-element-id -r=revision-id  
[-h]
```

ARGUMENTS**-u**

Specifies the user ID.

This is generally **infodba** or another user with administration privileges. If this argument is used without a value, the operating system user name is used.

Note

If Security Services single sign-on (SSO) is enabled for your server, the **-u** and **-p** arguments are authenticated externally through SSO rather than being authenticated against the Teamcenter database. If you do not supply these arguments, the utility attempts to join an existing SSO session. If no session is found, you are prompted to enter a user ID and password.

-p

Specifies the password.

If used without a value, the system assumes a null value.

If this argument is not used, the system assumes the *user-ID* value to be the password.

This argument is mutually exclusive with the **-pf** argument.

-pf

Specifies the password file. If used without a value, the system assumes a null value. If this argument is not used, the system assumes the *user-ID* value to be the password.

For more information about managing password files, see [Manage password files](#).

This argument is mutually exclusive with the **-p** argument.

-g

Specifies the group associated with the user.

If used without a value, the user's default group is assumed.

-i

Specifies the specification element's item ID to convert.

-r

Specifies the revision's ID.

-h

Displays help for this utility.

ENVIRONMENT

As specified in *Manually configuring your environment for Teamcenter utilities*.

FILES

As specified in *Log files produced by Teamcenter*.

RESTRICTIONS

To use this utility, you must be a user with system administration privileges or be granted authorization by a user with system administration privileges.

req_migrate_fulltext

Migrates all full text from the input PLM XML file to the Teamcenter database. It can create or update the full text and set the rich text contents of full text.

SYNTAX

req_migrate_fulltext [-u=*user-id*] [-p=*password* | -pf=*password-file*] [-g=*group*]
-file=*complete-path-to-plmxml-file*
[-h]

ARGUMENTS**-u**

Specifies the user ID.

This is generally **infodba** or another user with administration privileges. If this argument is used without a value, the operating system user name is used.

Note

If Security Services single sign-on (SSO) is enabled for your server, the **-u** and **-p** arguments are authenticated externally through SSO rather than being authenticated against the Teamcenter database. If you do not supply these arguments, the utility attempts to join an existing SSO session. If no session is found, you are prompted to enter a user ID and password.

-p

Specifies the password.

If used without a value, the system assumes a null value.

If this argument is not used, the system assumes the *user-ID* value to be the password.

This argument is mutually exclusive with the **-pf** argument.

-pf

Specifies the password file. If used without a value, the system assumes a null value. If this argument is not used, the system assumes the *user-ID* value to be the password.

For more information about managing password files, see [Manage password files](#).

This argument is mutually exclusive with the **-p** argument.

-g

Specifies the group associated with the user.

If used without a value, the user's default group is assumed.

-file

Specifies the complete path of input PLM XML file.

-h

Displays help for this utility.

ENVIRONMENT

As specified in [Manually configuring your environment for Teamcenter utilities](#).

FILES

As specified in [Log files produced by Teamcenter](#).

RESTRICTIONS

To use this utility, you must be a user with system administration privileges or be granted authorization by a user with system administration privileges.

req_migrate_grm

Migrates all trace links from the input PLM XML file to the Teamcenter database. It creates trace links between the source objects and target objects as specified in the PLM XML file. Trace links that already exist in the database are skipped.

SYNTAX

req_migrate_grm [-u=*user-id*] [-p=*password* | -pf=*password-file*] [-g=*group*]
-file=*complete-path-to-plmxml-file*
[-h]

ARGUMENTS**-u**

Specifies the user ID.

This is generally **infodba** or another user with administration privileges. If this argument is used without a value, the operating system user name is used.

Note

If Security Services single sign-on (SSO) is enabled for your server, the **-u** and **-p** arguments are authenticated externally through SSO rather than being authenticated against the Teamcenter database. If you do not supply these arguments, the utility attempts to join an existing SSO session. If no session is found, you are prompted to enter a user ID and password.

-p

Specifies the password.

If used without a value, the system assumes a null value.

If this argument is not used, the system assumes the *user-ID* value to be the password.

This argument is mutually exclusive with the **-pf** argument.

-pf

Specifies the password file. If used without a value, the system assumes a null value. If this argument is not used, the system assumes the *user-ID* value to be the password.

For more information about managing password files, see [Manage password files](#).

This argument is mutually exclusive with the **-p** argument.

-g

Specifies the group associated with the user.

If used without a value, the user's default group is assumed.

-file

Specifies the complete path of input PLM XML file.

-h

Displays help for this utility.

ENVIRONMENT

As specified in [Manually configuring your environment for Teamcenter utilities](#).

FILES

As specified in *Log files produced by Teamcenter*.

RESTRICTIONS

To use this utility, you must be a user with system administration privileges or be granted authorization by a user with system administration privileges.

req_migrate_richtext

Migrates rich text from Teamcenter 2007 to Teamcenter 8 and later versions. It is required only in special cases to recover rich text from the migrated data that is inaccessible because of format differences.

SYNTAX

req_migrate_richtext [-u=*user-id*] [-p=*password*] -pf=*password-file*] [-g=*group*] [-h]

ARGUMENTS**-u**

Specifies the user ID.

This is generally **infodba** or another user with administration privileges. If this argument is used without a value, the operating system user name is used.

Note

If Security Services single sign-on (SSO) is enabled for your server, the **-u** and **-p** arguments are authenticated externally through SSO rather than being authenticated against the Teamcenter database. If you do not supply these arguments, the utility attempts to join an existing SSO session. If no session is found, you are prompted to enter a user ID and password.

-p

Specifies the password.

If used without a value, the system assumes a null value.

If this argument is not used, the system assumes the *user-ID* value to be the password.

This argument is mutually exclusive with the **-pf** argument.

-pf

Specifies the password file. If used without a value, the system assumes a null value. If this argument is not used, the system assumes the *user-ID* value to be the password.

For more information about managing password files, see [Manage password files](#).

This argument is mutually exclusive with the **-p** argument.

-g

Specifies the group associated with the user.

If used without a value, the user's default group is assumed.

-h

Displays help for this utility.

ENVIRONMENT

As specified in [Manually configuring your environment for Teamcenter utilities](#).

FILES

As specified in [Log files produced by Teamcenter](#).

RESTRICTIONS

To use this utility, you must be a user with system administration privileges or be granted authorization by a user with system administration privileges.

Index

Numerics/Symbols

4gd_populate_cd utility 5-2
4th Generation Design (4GD) utilities
 4gd_populate_cd 5-2
 manage_effectivity_options 5-5
 ptn0_persist_dynamicMembers 5-7
 Ptn0_set_is_partition_owned_true 5-9
 purge_historical_revisions 5-10
 validate_revrule_effectivity 5-12

A

Access Manager utilities
 am_install_tree 2-35
ada_util utility 2-37
add_req_templates utility 23-2
add_se_templates utility 23-4
add_specmgr_templates 19-2
add_specmgr_templates utility 19-2
administration.log file 1-3
Aerospace and Defense utilities
 default_adschangemanagement_
 queries 17-4
 default_adsfoundation_queries 17-2
 update_locationcode_from_
 owningorg 17-6
am_install_tree utility 2-35
apn_medic utility 3-74
Appearance Configuration utilities
 appearance_updater 3-81
 appr_update_console 3-78
 appr_update_manager 3-76
 appr_update_supervisor 3-77
 appr_working_scheduler 3-84
 assy_jt_creator 10-20
 create_appearances 3-87
 fix_appearances 3-89
appearance_updater utility 3-81
appr_update_console utility 3-78
 appr_update_env file 3-85
appr_update_manager utility 3-76
appr_update_supervisor utility 3-77
appr_working_scheduler utility 3-84
AppRegUtil utility 2-152
Assembly on-demand sync 7-130

assy_jt_creator utility 10-20
Attribute definitions element, import/export
 file 11-22
Attribute field, import/export file 11-21
Attribute mapping
 tc_config_attr_mapping utility 2-126
 Teamcenter to external attributes 2-126
Attribute sharing utilities
 tc_config_attr_mapping 2-126
attribute_export utility 7-2
Audit Manager utilities
 audit_archive 13-22
 audit_purge 13-19
 combine_audit_files 13-26
 define_auditdefs 13-29
 migrate_audit_configs 13-14
 migrate_audit_data 13-16
audit_archive utility 13-22
audit_purge utility 13-19
Automotive Edition utilities
 gmo_migrate_usage_nves 16-38
 rdv_import_usage 16-52

B

Backup and Recovery utilities
 backup_modes 13-33
 backup_xmlinfo 13-35
 sfr_instances 13-37
backup_modes utility 13-33
backup_xmlinfo utility 13-35
Baseline revision, purging 3-47
Batch meshing utilities
 cae_execute_datamap 20-13
 cae_execute_structuremap 20-15
 epm_import_batch_meshing_
 results 20-8
 epm_notify_batch_meshing_
 results 20-11
Batch mode reporting 2-147, 2-150
batch_export_translate_import utility 7-8
batchmode_clearance_analysis 3-72
BLD keyword 11-22, 11-33
bmide_commontemplategenerator
 utility 2-50

Index

bmide_comparator utility 2-52
bmide_consolidator utility 2-53
bmide_deployment_lock utility 2-54
bmide_generate_compare_report
utility 2-58
bmide_generate_condition_report
utility 2-59
bmide_generate_datamodel_doc_report
utility 2-60
bmide_generate_datamodel_report
utility 2-63
bmide_generatecode utility 2-56
bmide_manage_batch_lovs utility 2-65
bmide_manage_templates utility 2-70
bmide_postupgradetotc utility 2-72
bmide_setupknowledgebase utility 2-74
bom_expand utility 3-6
bom_roll_up_report utility 3-11
bomwriter utility 9-2
BSM keyword 11-22, 11-33
build_fts_index utility 12-2
Business Modeler IDE utilities
bmide_commontemplategenerator 2-50
bmide_comparator 2-52
bmide_consolidator 2-53
bmide_deployment_lock 2-54
bmide_generate_compare_report 2-58
bmide_generate_condition_report 2-59
bmide_generate_datamodel_doc_
report 2-60
bmide_generate_datamodel_report 2-63
bmide_generatecode 2-56
bmide_manage_batch_lovs 2-65
bmide_manage_templates 2-70
bmide_postupgradetotc 2-72
bmide_setupknowledgebase 2-74
business_model_extractor 2-76
business_model_updater 2-78
change_datasets 2-80
clips_dataset_upload 2-82
deploy_archive 2-90
execute_rbf_rules 2-93
get_key_definition 2-97
get_key_string 2-98
getglobalconstantvalue 2-95
getpropertyconstantvalue 2-100
gettypeconstantvalue 2-102
manage_icon_files 2-104
manage_model_files 2-106
mfk_update 2-108
package_live_updates 2-110
process_action_rules 2-112
business_model_extractor utility 2-76
business_model_updater utility 2-78

C

CAE utilities

cae_execute_cae_accountability_
check 20-21
cae_execute_datamap 20-13
cae_execute_structuremap 20-15
cae_migrate_atl_preferences 20-2
cae_migrate_datamap_definition 20-17
cae_save_result_data 20-5
cae_validate_structuremap 20-19
epm_import_batch_meshing_
results 20-8
epm_notify_batch_meshing_
results 20-11
cae_execute_cae_accountability_check
utility 20-21
cae_execute_datamap utility 20-13
cae_execute_structuremap utility 20-15
cae_migrate_atl_preferences utility 20-2
cae_migrate_datamap_definition
utility 20-17
cae_save_result_data utility 20-5
cae_validate_structuremap utility 20-19
cc_writer utility 10-2
change_datasets utility 2-80
Class definitions element 11-22
Classification utilities
ics_connect 11-10
icsutility 11-2
icsxml 11-6
mrm_export_resources 10-13
smlutility 11-12
CLASSPATH preference 1-6
clean_backpointer utility 13-70
cleanup_ic_objects utility 7-11
cleanup_recovery_table utility 13-72
clear_process_stage_list utility 6-2
Clearance analysis 3-72
clearlocks utility 13-65
clips_dataset_upload utility 2-82
collect_garbage utility 22-2
combine_audit_files utility 13-26
Component on-demand sync 7-130
Configure utilities 1-1
Consolidation utilities
tcxml_xfer_ownership 7-150
Consumer Packaged Goods utilities
add_specmgr_templates 19-2
Content Management
contmgmt_migration_100 utility 3-4
contmgmt_upgrade_8x utility 3-2
contmgmt_migration_100 utility 3-4
contmgmt_upgrade_8x utility 3-2

- Conventions
 - Syntax definitions 1-6
- convert_distribution_lists utility 13-47
- convert_forms utility 2-84
- convert_license_log utility 13-74
- convert_replica_files_to_stubs utility 7-15
- Converting distribution lists 13-47
- create_appearances utility 3-87
- create_or_update_bbox_and_tso utility 3-14
- create_project utility 13-51
- create_validationdata utility 2-141
- Creating external attribute mapping 2-126
- Creating FMS encryption keys 22-71
- Creating projects 13-51
- Customization utilities
 - convert_forms 2-84
 - taxonomy 2-114
- D**
- DAT keyword 11-22, 11-34
- Data access management utilities
 - ada_util 2-37
 - install_authorization_rules 2-43
 - install_callback 2-46
 - install_vminfo_acl 2-48
- Data sharing utilities
 - attribute_export 7-2
 - batch_export_translate_import 7-8
 - cleanup_ic_objects 7-11
 - convert_replica_files_to_stubs 7-15
 - data_share 7-20
 - data_sync 7-39
 - database_verify 7-17
 - diff_xml 7-56
 - distributed_execute 7-58
 - dsa_util 7-60
 - ensure_site_consistency 7-65
 - export_recovery 7-70
 - idsminetd 7-74
 - import_file 7-76
 - item_export 7-80
 - item_import 7-86
 - item_relink 7-89
 - item_rename 7-94
 - item_report 7-97
 - migrate_organization 7-102
 - migrate_saved_searches 7-106
 - pdx_export 7-108
 - plmxml_export 7-110
 - plmxml_import 7-117
 - plmxml_tm_edit_xsl 7-121
 - step_export 7-123
 - step_import 7-126
 - sync_form_util 7-128
 - sync_on_demand 7-130
 - tcexcel_import 10-5
 - tcxml_export 7-136
 - tcxml_import 7-144
 - tcxml_validate 7-148
 - upload_plmxml_struct 7-155
 - validate_and_replicate_assembly 7-157
- data_share utility 7-20
- data_sync utility 7-39
- database_verify utility 7-17
- dataset_cleanup utility 22-11
- default_adschangemanagement_queries 17-4
- default_adschangemanagement_queries utility 17-4
- default_adsfoundation_queries 17-2
- default_adsfoundation_queries utility 17-2
- default_queries utility 12-8
- define_auditdefs utility 13-29
- delete_item_data utility 22-15
- Deleting external attribute mapping 2-126
- deploy_archive utility 2-90
- diff_xml utility 7-56
- Directories
 - \$TC_TMP_DIR 1-4
- Dispatcher
 - Creating request objects 13-39
 - Generating service requests 13-45
 - Managing request objects 13-43
- Dispatcher utilities
 - dispatcher_create_rqst 13-39
 - dispatcher_util 13-43
 - runBatch 13-100
 - SS_GenSvcRqst 13-45
- dispatcher_create_rqst utility 13-39
- dispatcher_util utility 13-43
- distributed_execute utility 7-58
- Distribution list conversion 13-47
- dsa_util utility 7-60
- E**
- Effectivity mode utilities
 - effupgrade 3-70
- effupgrade utility 3-70
- Elements
 - Attribute definitions 11-22
 - Class definitions 11-22
 - File header 11-21
 - Group definitions 11-22

Index

- Import/exports files 11-21
- Instances 11-22
- List definitions 11-21
- Subclass definitions 11-22
- Embedded Software Manager 21-2
- ensure_site_consistency utility 7-65
- Environment variables
 - TC_DATA 1-1
 - TC_ROOT 1-1
 - TC_SLOW_SQL 7-153
 - TC_SQL_DEBUG 7-153
- epm_import_batch_meshing_results
 - utility 20-8
- epm_notify_batch_meshing_results
 - utility 20-11
- Error 100228 7-55
- execute_rbf_rules utility 2-93
- export_attr_mappings utility 14-2
- export_recovery utility 7-70

F

- fccstat utility 22-66
- File formats, SML 11-20
- File header element 11-21
- File Management System (FMS)
 - utilities 22-58, 22-66
 - FSCWholeFileCacheUtil 22-64
 - install_encryptionkeys 22-71
 - load_fccache 22-73
 - load_fsccache 22-77
- Files
 - administration.log 1-3
 - .appr_update_env 3-85
 - Export elements 11-21
 - Import elements 11-21
 - security.log 1-4
 - tc_install.log 1-4
- find_all_key_value_pairs utility 2-116
- find_appearances utility 12-10
- find_recently_saved_item_rev
 - utility 12-14
- find_released_datasets utility 16-2
- find_released_item_rev utility 12-16
- fix_appearances utility 3-89
- FMS encryption keys 22-71
- fscadmin utility 22-58
- FSCWholeFileCacheUtil utility 22-64

G

- gcs_import utility 10-16
- generate_client_meta_cache utility 13-76
- generate_metadata_cache utility 13-80
- generate_tc_ps_path utility 3-18

- get_bvr_structure utility 16-4
- get_key_definition utility 2-97
- get_key_string utility 2-98
- get_qpl_harvester_assemblies utility 9-49
- getglobalconstantvalue utility 2-95
- getpropertyconstantvalue utility 2-100
- gettypeconstantvalue utility 2-102
- Global data caching 7-20, 7-39
- global_transfer utility 6-4
- gmo_assoc_items_to_project utility 16-10
- gmo_change_itemid_naming_rule
 - utility 16-12
- gmo_change_owner utility 16-14
- gmo_check_comp_names utility 16-16
- gmo_clone utility 16-18
- gmo_create_material_form_templates
 - utility 16-20
- gmo_find_changed_install_assem
 - utility 16-23
- gmo_get_partspec utility 16-25
- gmo_get_pds_info utility 16-26
- gmo_install_usage_queries utility 16-28
- gmo_ipvbom_export utility 16-32
- gmo_ipvbom_import utility 16-30
- gmo_ipvbom_pulldate utility 16-34
- gmo_migrate_ulink_to_rdvauto
 - utility 16-36
- gmo_migrate_usage_nves utility 16-38
- gmo_set_rel_status utility 16-40
- gmo_split_usage utility 16-42
- gmo_update_vas_data utility 16-44
- gmo_upgrade_dlist_objects utility 16-46
- gmo_validate_xml utility 16-48
- gmo_vds_util utility 16-50
- Group definitions element 11-22

H

- harvest_mmv_index utility 8-1
- harvester_jt.pl utility 9-12
- harvester.pl utility 9-9
- hsm_capacity_alert utility 22-19
- hsm_report utility 22-21

I

- ics_connect utility 11-10
- ics_localize_class_attributes 2-118
- icsutility program 11-20
- icsutility utility 11-2
- icsxml utility 11-6
- identify_non_structure_edges utility 3-21
- identify_non_structure_edges utility 3-21
- idsminetd utility 7-74
- import_attr_mappings utility 14-4

Import/Export file example 11-23
 import_export_reports utility 2-150
 import_file utility 7-76
 import_nxcam_post_files utility 10-8
 Importing data, using the smlutility
 program 11-20
 Importing workflow templates 7-117
 index_verifier utility 22-23
 install utility 13-2
 install_algebraicformulas 21-2
 install_algebraicformulas utility 21-2
 install_authorization_rules utility 2-43
 install_callback utility 2-46
 install_default_report_designs
 utility 2-144
 install_encryptionkeys utility 22-71
 install_event_types utility 13-82
 install_handlers utility 6-6
 install_kbl utility 21-4
 install_vminfo_acl utility 2-48
 Installation utilities, tem 13-13
 Installing report designs 2-144
 installmgr.bat utility 13-58
 Instances
 Element 11-22
 Integration utilities
 proxy_sync 15-8
 tcc_context_upload 15-2
 ipa_b_executer utility 10-27
 item_export utility 7-80
 item_import utility 7-86
 item_relink utility 7-89
 item_rename utility 7-94
 item_report utility 7-97
 item_to_part_design utility 3-25

K

Keywords
 BLD 11-22, 11-33
 BSM 11-22, 11-33
 DAT 11-22, 11-34
 SMD 11-30, 11-32
 SML 11-22, 11-30
 SMV 11-22, 11-27
 STD 11-21–11-22, 11-26
 STV 11-21, 11-26

L

l10n_import_export utility 2-120
 l10n_purge_translations utility 2-124
 LDAP synchronization 2-129
 ldapsync utility 2-129

List definitions element 11-21
 list_ir_with_bvr utility 16-6
 list_types utility 13-87
 list_users utility 13-89
 load_fccecache utility 22-73
 load_fsccecache utility 22-77
 Localization utilities
 find_all_key_value_pairs 2-116
 ics_localize_class_attributes 2-118
 l10n_import_export 2-120
 Log file best practices 1-3

M

Maintenance utilities
 pom_audit_manager 13-31
 make_user utility 2-131
 manage_effectivity_options utility 5-5
 manage_icon_files utility 2-104
 manage_model_files utility 2-106
 Manufacturing utilities
 cc_writer 10-2
 gcs_import 10-16
 import_nxcam_post_files 10-8
 ipa_b_executer 10-27
 mrm_export_resources 10-13
 Mapping to external attributes 2-126
 mark_for_migrate utility 22-26
 material_export 18-2
 material_export utility 18-2
 material_import 18-4
 material_import utility 18-4
 Materials Management utilities
 material_export 18-2
 material_import 18-4
 subscmpl_import_template 18-6
 subscmpl_msd_import 18-8
 subscmpl_validate_compliance_
 results 18-11
 mfk_update utility 2-108
 mgrstop utility 13-60
 migrate_audit_configs utility 13-14
 migrate_audit_data utility 13-16
 migrate_eda_data utility 21-7
 migrate_gmo_to_gcn_events utility 16-8
 migrate_home_folder utility 13-91
 migrate_organization utility 7-102
 migrate_saved_searches utility 7-106
 migrate_wf_handlers utility 6-10
 Migrating Microsoft Office forms 13-49
 Migration utilities
 convert_distribution_lists 13-47
 move_mso_forms 13-49
 Monitoring FMS FSC Servers 22-58

Index

Monitoring local FMS cache 22-66
move_mso_forms utility 13-49
move_volume_files utility 22-28
mrm_export_resources utility 10-13
multiple_svr_variant_configurator
utility 3-30

N

nxmgr_upgrade_bvrsyncform
utility 14-10
nxmgr_upgrade_transforms utility . . . 14-13

O

Object on-demand sync 7-130
Object validation utilities, create_
validationdata 2-141
On-demand sync assembly report 7-135
On-demand sync component report . . . 7-135
On-demand synchronization
Assembly 7-130
Component 7-130
Object 7-130
Site support 7-130
Organization utilities
ldapsync 2-129
make_user 2-131

P

package_live_updates utility 2-110
pdfgenerator utility 13-108
pdx_export utility 7-108
plmxml_export utility 7-110
plmxml_import utility 7-117
plmxml_tm_edit_xsl utility 7-121
pom_audit_manager utility 13-31
Portfolio, Program and Project Management
utilities
create_project 13-51
update_project_data 13-54
Preferences
CLASSPATH 1-6
TC_Administration_Logging 1-4
TC_Application_Logging 1-4, 1-6
TC_force_remote_sites_exclude_
files 7-20, 7-39
TC_Journalling 1-6
TC_Security_Logging 1-4
Preferences Manager utilities
preferences_manager 2-2
preferences_manager -
mode=append 2-7

preferences_manager -
mode=category 2-9
preferences_manager -
mode=cleanup 2-11
preferences_manager -mode=cleanup_
definitions 2-13
preferences_manager -mode=clear . . . 2-14
preferences_manager -
mode=delete 2-27
preferences_manager -
mode=export 2-16
preferences_manager -
mode=generatexml 2-19
preferences_manager -
mode=import 2-21
preferences_manager -
mode=migrate 2-25
preferences_manager -
mode=remove 2-29
preferences_manager -
mode=upgradexml 2-31
Preferences, supervisor.enabled 3-85
preferences_manager -mode=append
utility 2-7
preferences_manager -mode=category
utility 2-9
preferences_manager -mode=cleanup
utility 2-11
preferences_manager -mode=cleanup_
definitions utility 2-13
preferences_manager -mode=clear
utility 2-14
preferences_manager -mode=delete
utility 2-27
preferences_manager -mode=export
utility 2-16
preferences_manager -mode=generatexml
utility 2-19
preferences_manager -mode=import
utility 2-21
preferences_manager -mode=migrate
utility 2-25
preferences_manager -mode=remove
utility 2-29
preferences_manager -mode=upgradexml
utility 2-31
Prerequisites for the utilities 1-1
process_action_rules utility 2-112
Product structure maintenance utilities
bom_expand 3-6
bom_roll_up_report 3-11
create_or_update_bbox_and_tso 3-14
generate_tc_ps_path 3-18

identify_non_structure_edges 3-21
 item_to_part_design 3-25
 multiple_svr_variant_configurator ... 3-30
 ps_exportconfignassembly 3-34
 ps_rename_bvrs 3-37
 ps_traverse 3-39
 ps_upload 3-43
 purge_baselined_item_revisions 3-47
 qsearch_process_queue 3-49
 update_bomchanges 3-55, 3-57
 update_objecttype utility 3-59
 update_project_bom 3-60
 upgrade_rev_rules 3-65
 upgrade_variants 3-67
 Properties tab 11-28, 11-33
 proxy_sync utility 15-8
 ps_exportconfignassembly utility 3-34
 ps_rename_bvrs utility 3-37
 ps_traverse utility 3-39
 ps_upload utility 3-43
 ptn0_persist_dynamicMembers utility .. 5-7
 Ptn0_set_is_partition_owned_true
 utility 5-9
 purge_baselined_item_revisions
 utility 3-47
 purge_datasets utility 22-33
 purge_file_cache utility 13-93
 purge_historical_revisions 5-10
 purge_historical_revisions utility 5-10
 purge_invalid_subscriptions utility ... 13-63
 purge_processes utility 6-13
 purge_volumes utility 22-37
 Purging
 Baselined item revisions 3-47
 Datasets 22-33
 File cache 13-93
 Invalid subscriptions 13-63
 Volumes 22-37
 Workflow processes 6-13

Q

qpl_convert_occs2tc 9-15
 qsearch_process_queue utility 3-49
 Query utilities
 build_fts_index 12-2
 default_queries 12-8
 find_appearances 12-10
 find_recently_saved_item_rev 12-14
 find_released_item_rev 12-16
 query_xml 12-18
 tc_set_query_where_run 12-25
 query_xml utility 12-18

R

rc.tc.mgr_<config> utility 13-61
 RDV utilities
 bomwriter 9-2
 get_qpl_harvester_assemblies 9-49
 harvester_jt.pl 9-12
 harvester.pl 9-9
 qpl_convert_occs2tc 9-15
 rdv_context_download 9-17
 rdv_migrate_architecture 9-23
 rdv_migrate_part_solutions 9-25
 rdv_run_audit_report 9-27
 rdv_set_default_variant_condition ... 9-30
 start_sco_dispatcher 9-32
 sync_product_apns 3-74, 9-35
 sync_product_variant_data 9-42
 tess_server 14-8
 rdv_context_download utility 9-17
 rdv_import_usage utility 16-52
 rdv_migrate_architecture utility 9-23
 rdv_migrate_part_solutions utility 9-25
 rdv_run_audit_report 9-27
 rdv_set_default_variant_condition 9-30
 reencode_filenames utility 22-39
 refile_info utility 14-6
 release_man utility 6-15
 released_parts_collector utility 6-18
 rep_batch_report utility 2-147
 Report Designer utilities
 import_export_reports 2-150
 install_default_report_designs 2-144
 rep_batch_report 2-147
 Report designs, installing 2-144
 report_volume utility 22-42
 Reporting utilities
 import_export_reports 2-150
 rep_batch_report 2-147
 req_convert_to_plaintext utility 23-6
 req_migrate_fulltext utility 23-8
 req_migrate_grm utility 23-10
 req_migrate_richtext utility 23-12
 reset_user_home_folder utility 13-95
 review_volumes utility 22-44
 runBatch utility 13-100

S

Sample file directory 11-20
 Scripts
 tc_cshvars 1-2
 tc_profilevars 1-2
 security.log file 1-4

Index

sfr_instances utility 13-37
site_util utility 13-97
Smart codes
 smartuibldr_configure utility 4-2
smartuibldr_configure utility 4-2
SMD keyword 11-30, 11-32
sml file format 11-20
SML file format 11-20
SML keyword 11-22, 11-30
smlutility program 11-20
smlutility utility 11-12
SMV keyword 11-22, 11-27
SS_GenSvcRqst utility 13-45
start_sco_dispatcher 9-32
STD keyword 11-21–11-22, 11-26
step_export utility 7-123
step_import utility 7-126
STV keyword 11-21, 11-26
Subclass definitions element 11-22
subscmpl_import_template 18-6
subscmpl_import_template utility 18-6
subscmpl_msd_import 18-8
subscmpl_msd_import utility 18-8
subscmpl_validate_compliance_
 results 18-11
subscmpl_validate_compliance_results
 utility 18-11
Subscription Manager utilities,
 pdfgenerator 13-108
Subscription Manager utilities, purge_invalid_
 subscriptions 13-63
supervisor.enabled preference 3-85
sync_form_util utility 7-128
sync_on_demand utility 7-130
sync_product_apns utility 9-35
sync_product_variant_data utility 9-42
syncCache utility 22-47
Synchronizing on demand 7-130
Synchronizing user data 2-129
Syntax definition conventions 1-6
System maintenance utilities
 clean_backpointer 13-70
 cleanup_recovery_table 13-72
 clearlocks 13-65
 convert_license_log 13-74
 generate_metadata_cache 13-80
 install 13-2
 install_event_types 13-82
 installmgr.bat 13-58
 list_types 13-87
 list_users 13-89
 mgrstop 13-60
 migrate_home_folder 13-91

 purge_file_cache 13-93
 rc.tc.mgr_<config> 13-61
 reset_user_home_folder 13-95
 site_util 13-97
 tc_mail_smtp 13-102
 uih_to_xml 13-106
 uninstallmgr.bat 13-59
Systems Engineering utilities
 add_req_templates 23-2
 add_se_templates 23-4
 req_convert_to_plaintext 23-6
 req_migrate_fulltext 23-8
 req_migrate_grm 23-10
 req_migrate_richtext 23-12

T

Tabs, properties 11-28, 11-33
taxonomy utility 2-114
TC_Administration_Logging
 preference 1-4
TC_Application_Logging preference 1-6
TC_Application_Logging preference 1-4
tc_config_attr_mapping utility 2-126
tc_cshvars script 1-2
TC_DATA environment variable 1-1
TC_force_remote_sites_exclude_files
 preference 7-20
TC_force_remote_sites_exclude_files
 preference 7-39
tc_install.log file 1-4
TC_Journaling preference 1-6
tc_mail_smtp utility 13-102
tc_profilevars script 1-2
TC_ROOT environment variable 1-1
TC_Security_Logging preference 1-4
tc_set_query_where_run utility 12-25
\$TC_TMP_DIR directory 1-4
tc_workflow_postprocess utility 6-20
tcc_context_upload utility 15-2
tcexcel_import utility 10-5
tcmemstat utility 22-48
tcxml_export utility 7-136
tcxml_import utility 7-144
tcxml_validate utility 7-148
tcxml_xfer_ownership 7-150
Teamcenter Automotive Edition utilities
 find_released_datasets 16-2
 get_bvr_structure 16-4
 gmo_assoc_items_to_project 16-10
 gmo_change_itemid_naming_rule 16-12
 gmo_change_owner 16-14
 gmo_check_comp_names 16-16

gmo_clone 16-18
 gmo_create_material_form_
 templates 16-20
 gmo_find_changed_install_assem .. 16-23
 gmo_get_partspec 16-25
 gmo_get_pds_info 16-26
 gmo_install_usage_queries 16-28
 gmo_ipvbom_export 16-32
 gmo_ipvbom_import 16-30
 gmo_ipvbom_pulldate 16-34
 gmo_migrate_ulink_to_rdvauto 16-36
 gmo_set_rel_status 16-40
 gmo_split_usage 16-42
 gmo_update_vas_data 16-44
 gmo_upgrade_dlist_objects 16-46
 gmo_validate_xml 16-48
 gmo_vds_util 16-50
 list_ir_with_bvr 16-6
 migrate_gmo_to_gcn_events 16-8
 Teamcenter Integration for NX utilities
 export_attr_mappings 14-2
 import_attr_mappings 14-4
 nxmgr_upgrade_bvrsyncform 14-10
 nxmgr_upgrade_transforms 14-13
 refile_info 14-6
 Teamcenter Interface utilities,
 AppRegUtil 2-152
 Teamcenter mechatronics process
 management utilities
 install_algebraicformulas 21-2
 install_kbl utility 21-4
 migrate_eda_data 21-7
 update_gde_types 21-5
 tem utility 13-13
 tess_server utility 14-8
 tspstat utility 22-49

U

uih_to_xml utility 13-106
 uninstallmgr.bat utility 13-59
 unmigrate_from_hsm utility 22-51
 update_bomchanges utility 3-55
 update_gde_types utility 21-5
 update_locationcode_from_
 owningorg 17-6
 update_locationcode_from_owningorg
 utility 17-6
 update_loggeddatetogmt 3-57
 update_objecttype utility 3-59
 update_project_bom utility 3-60
 update_project_data utility 13-54
 Updating BOM items in projects 3-60
 upgrade_nx_cam_templates utility ... 10-11

upgrade_rev_rules utility 3-65
 upgrade_variants utility 3-67
 upgrade_vendor_part utility 22-53
 upload_plmxml_struct utility 7-155

Utilities

contmgmt_migration_100 3-4
 contmgmt_upgrade_8x 3-2
 l10n_purge_translations 2-124

V

validate_and_replicate_assembly
 utility 7-157
 validate_revrule_effectivity 5-12
 validate_revrule_effectivity utility 5-12
 Variant model
 Upgrading to 3-67
 Verification verdicts 7-43
 verify_tasks utility 6-24
 Visualization utilities
 harvest_mmv_index 8-1
 vm_report utility 22-54
 vms_upgrade utility 22-50
 Volume and database management utilities
 collect_garbage 22-2
 dataset_cleanup 22-11
 delete_item_data 22-15
 hsm_capacity_alert 22-19
 hsm_report 22-21
 index_verifier 22-23
 mark_for_migrate 22-26
 move_volume_files 22-28
 purge_datasets 22-33
 purge_volumes 22-37
 reencode_filenames 22-39
 report_volume 22-42
 review_volumes 22-44
 syncCache 22-47
 tcmemstat 22-48
 tspstat 22-49
 unmigrate_from_hsm 22-51
 upgrade_nx_cam_templates 10-11
 upgrade_vendor_part 22-53
 vm_report 22-54
 vms_upgrade 22-50
 xml_validator 22-56

W

Workflow templates, importing 7-117
 Workflow utilities
 clear_process_stage_list 6-2
 global_transfer 6-4
 install_handlers 6-6

Index

migrate_wf_handlers 6-10
purge_processes 6-13
release_man 6-15
released_parts_collector utility 6-18
tc_workflow_postprocess 6-20
verify_tasks 6-24

X

xml_validator utility 22-56