

*Teamcenter® 2005 SR1  
engineering process management*

*Teamcenter Engineering  
Utilities Reference*



*Teamcenter® 2005 SR1  
engineering process management*

*Teamcenter Engineering  
Utilities Reference*

This product is intended for use only described in this document. UGS cannot be responsible for the proper functioning of undescribed features and parameters.

---

# *Manual History*

<b>Manual Revision</b>	<b>Teamcenter Engineering Version</b>	<b>Publication Date</b>
A	9.0	December 2003
B	9.1	May 2004
C	2005 (10.0)	September 2005
D	2005 SR1	June 2006

This edition obsoletes all previous editions.

---

# Contents

<b>Preface</b> .....	<b>7</b>
Audience .....	7
Organization .....	7
Conventions .....	9
Teamcenter Engineering Documentation .....	12
Submitting Comments .....	12
Software Copyright and Trademark Notices .....	12
<b>Introduction</b> .....	<b>1-1</b>
Manually Setting the Teamcenter Engineering Environment .....	1-1
Log Files .....	1-3
<b>Configuration Utilities</b> .....	<b>2-1</b>
Preferences Manager .....	2-1
Access Manager .....	2-6
Business Modeler .....	2-9
Change Management .....	2-36
ClearCase Integration With Teamcenter Engineering .....	2-39
eIntegrator .....	2-41
Organization .....	2-44
Object Validation .....	2-47
Teamcenter Reporting .....	2-50
Teamcenter Engineering Data Integration Services Adapter .....	2-56
Teamcenter Linking .....	2-60
<b>Product Configuration Utilities</b> .....	<b>3-1</b>
Appearance Configuration .....	3-1
Product Structure Utilities .....	3-13
<b>Workflow Utilities</b> .....	<b>4-1</b>
<b>Data Sharing Utilities</b> .....	<b>5-1</b>
<b>Customization Utilities</b> .....	<b>6-1</b>
<b>Repeatable Digital Validation (RDV) Utilities</b> .....	<b>7-1</b>
RDV Cache Maintenance .....	7-20

<b>Classification Utilities</b> .....	<b>8-1</b>
<b>Query Utilities</b> .....	<b>9-1</b>
<b>Maintenance Utilities</b> .....	<b>10-1</b>
Archive/Restore .....	10-19
Audit Manager .....	10-30
Backup and Recovery .....	10-38
Engineering Translation Services .....	10-50
Migration .....	10-59
Project .....	10-65
Subscription Manager .....	10-75
System Maintenance .....	10-78
<b>NX Manager Utilities</b> .....	<b>11-1</b>
<b>Integration Utilities</b> .....	<b>12-1</b>
Teamcenter Engineering/Teamcenter Community Integration .....	12-1
Teamcenter Engineering/Teamcenter Requirements Integration .....	12-12
<b>Teamcenter Automotive Edition Utilities</b> .....	<b>13-1</b>
<b>Computer-Aided Engineering (CAE) Utilities</b> .....	<b>14-1</b>
<b>Mechatronics Utilities</b> .....	<b>15-1</b>
<b>Volume and Database Management Utilities</b> .....	<b>16-1</b>
File Management System (FMS) Utilities .....	16-60
<b>Index</b> .....	<b>Index-1</b>

---

## **Figures**

---

2-1. XML File Format .....	2-52
3-1. Sample Configuration File .....	3-21
3-2. Product Structure Input File .....	3-28
4-1. tc_workflow_postprocess Utility .....	4-10
4-2. tc_workflow_postprocess Utility .....	4-11
6-1. tc_erp_schema Output .....	6-9
7-1. Vehicle Assembly Structure .....	7-22
9-1. find_appearances Utility .....	9-12
9-2. XML File Format .....	9-22
9-3. XML File Example .....	9-23
9-4. XML File Example .....	9-23
10-1. Auto Archive Template File .....	10-20
10-2. Auto Restore Template File .....	10-23
10-3. Audit Data Table .....	10-34
12-1. Sample PLM XML File .....	12-5
12-2. Sample AJT File .....	12-6

13-1. Sample Directory Structure .....	13-12
16-1. Sample editor_shell.dat file .....	16-11

## **Tables**

---

8-1. Replacement Characters .....	8-6
-----------------------------------	-----





---

# Preface

---

This document is a utilities reference manual for Teamcenter® Engineering. Teamcenter Engineering belongs to the UGS® portfolio of digital product lifecycle management software and services.

---



This document supplements the *Teamcenter Engineering Help Library*. If you find information inconsistent with that found in that *Teamcenter Engineering Help Library*, contact the UGS Global Technical Access Center (GTAC) for clarification.

## Audience

The readers of this guide should be experienced Teamcenter Engineering system administrators.

## Organization

This manual contains the following chapters:

Chapter 1	<i>Introduction</i> describes Teamcenter Engineering utilities, log files, and environment variables.
Chapter 2	<i>Configuration Utilities</i> describes the utilities used to configure your Teamcenter Engineering installation.
Chapter 3	<i>Product Configuration Utilities</i> describes the utilities used to manage Teamcenter Engineering product configurations.
Chapter 4	<i>Workflow Utilities</i> describes the command line utilities related to Teamcenter Engineering workflow.
Chapter 5	<i>Data Sharing Utilities</i> describes the utilities used to manage data sharing activities.
Chapter 6	<i>Customization Utilities</i> describes utilities used to customize Teamcenter Engineering.
Chapter 7	<i>Repeatable Digital Validation (RDV) Utilities</i> describes the utilities used to configure and maintain Teamcenter Engineering RDV.
Chapter 8	<i>Classification Utilities</i> describes the utilities used to define, maintain, import, and export Classification classification hierarchy data.

Chapter 9	<i>Query Utilities</i> describes the utilities used to reinstall default query forms, create queries to find recently saved or released item revisions, build keyword indices, and maintain and run queries from an XML formatted file.
Chapter 10	<i>Maintenance Utilities</i> describes the utilities used to maintain your Teamcenter Engineering installation and database.
Chapter 11	<i>NX Manager Utilities</i> describes the utilities used to import and export attribute mappings and assemblies in to and out of the Teamcenter Engineering database, update files to the latest version, transfer files between Teamcenter Engineering databases, upgrade transforms, upgrade assemblies to use the <b>BVRSyncInfo</b> form, and synchronize attribute protection attributes for <b>UGPART</b> and <b>UGMASTER</b> datasets.
Chapter 12	<i>Integration Utilities</i> describes the utilities associated with integrations between Teamcenter Engineering, other Teamcenter applications, and third-party software applications.
Chapter 13	<i>Teamcenter Automotive Edition Utilities</i> describes the utilities that are used to administer the Teamcenter Automotive Edition and/or Teamcenter Automotive Edition–GM Overlay.
Chapter 14	<i>Computer-Aided Engineering (CAE) Utilities</i> the utilities used to notify users of batch meshing results and import those results in to the Teamcenter database. In addition, this chapter describes the utility that saves result data from an analysis application.
Chapter 15	<i>Mechatronics Utilities</i> describes the utilities used to administer the Mechatronics solution.
Chapter 16	<i>Volume and Database Management Utilities</i> describes the utilities used to manage Teamcenter volumes and databases.

## Conventions

This manual uses the conventions described in the following sections.

### Revision Marks

Technical changes are marked by a bar adjacent to the changed text.

### Note, Caution, and Warning Icons

The following icons represent note, caution, and warning messages:



A note icon identifies general instructions or comments that need to be emphasized.



A caution icon identifies practices that can either produce results contrary to what you expect or result in damage to software or data.



A warning icon identifies practices that could result in permanent loss of data or software.

### Names and Values

This manual represents system names, file names, and values in fonts that help you interpret the name or value. For example:

The file name is **pom\_schema\_server-name\_sid**.

The conventions are:

<b>Bold</b>	<p>Bold font represents unvarying text or numbers within a name or value. Capitalization is as it appears.</p> <p>In the preceding example, <b>pom_schema_</b> identifies an unvarying portion of the name.</p>
<i>Italic</i>	<p>Italic font represents text or numbers that vary. The characters in italic text describe the entry. Letters are shown in lowercase, but the varying text may include uppercase letters.</p> <p>In the preceding example, <i>server-name</i> and <i>sid</i> represent varying portions of the name.</p>
<i>text-text</i>	<p>A hyphen separates two words that describe a single entry.</p> <p>In the preceding example, <i>server-name</i> is a single value.</p>

For example, the name of the **pom\_schema\_server-name\_sid** file may be:

**pom\_schema\_Blue5\_f731**

## Command Line Entries, File Contents, and Code

This manual represents command line input and output, the contents of system files, and computer code in fonts that help you understand how to enter text or to interpret displayed text. For example, the following line represents a command entry:

```
create_change_types -u=user-name -p=password -g=group -f=file-name
```

The conventions are:

<i>Monospace</i>	<p>Monospace font represents text or numbers you enter on a command line, the computer's response, the contents of system files, and computer code.</p> <p>Capitalization and spacing are shown exactly as you must enter the characters or as the computer displays the characters.</p> <p>In the preceding example, <code>create_change_types</code> identifies an unvarying portion of the command.</p>
<i>Italic</i>	<p>Italic font represents text or numbers that vary. The words in italic text describe the entry.</p> <p>The words are shown in lowercase letters, but the varying text may include uppercase letters. When entering text, use the case required by the system.</p> <p>In the preceding example, <i>user-name</i>, <i>password</i>, <i>group</i>, and <i>file-name</i> identify varying portions of the command.</p>
<i>text-text</i>	<p>A hyphen separates two words that describe a single entry.</p> <p>In the preceding example, <i>user-name</i> is a single entry in the command.</p>

The following example is a correct entry for the preceding **create\_change\_types** command:

```
create_change_types -u=infodba -p=KLH3b -g=dba -f=change_types.dat
```

## Syntax Definitions

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

**harvester\_jt.pl** [*bookmark-file-name bookmark-file-name ...*]  
[*directory-name directory-name ...*]

The conventions are:

<b>Bold</b>	Bold text represents words and symbols you must enter exactly as shown.  In the preceding example, you enter <b>harvester_jt.pl</b> exactly as shown.
<i>Italic</i>	Italic text represents values that you supply.  In the preceding example, you supply values for <i>bookmark-file-name</i> and <i>directory-name</i> .
<i>text-text</i>	A hyphen separates two words that describe a single value.  In the preceding example, <i>bookmark-file-name</i> is a single value.
[ ]	Brackets represent optional elements.
...	An ellipsis indicates that you can repeat the preceding element.

Following are examples of correct syntax for the **harvester\_jt.pl** command:

```
harvester_jt.pl
harvester_jt.pl assembly123.bkm
harvester_jt.pl assembly123.bkm assembly124.bkm assembly125.bkm
harvester_jt.pl AssemblyBookmarks
```

## Teamcenter Engineering Documentation

Teamcenter Engineering documentation is provided as online help and as printable manuals:

- You can access online help by choosing **Help** from the menu bar of a Teamcenter Engineering rich client application or by clicking one of the links under the **Help** icon in the Teamcenter Engineering thin client user interface.
- You can access the printable manuals from the Teamcenter Engineering Documentation CD-ROM. To view the PDF-formatted manuals, use Adobe Acrobat Reader, downloadable free-of-charge from Adobe Systems Incorporated at the following URL:

<http://www.adobe.com>

Acrobat Reader allows you to view these manuals online and print selected pages, sections, or the entire manual. UGS grants permission for Teamcenter Engineering users to print, duplicate, and distribute this documentation.

## Submitting Comments

Portions of Teamcenter software are provided by third-party vendors. Special agreements with these vendors require UGS to handle all problem reports concerning the software they provide. Please submit all comments directly to UGS.

Please feel free to give us your opinion of the usability of this manual, to suggest specific improvements, and to report errors. Mail your comments to:

UGS Technical Communications  
4233 Lexington Avenue N., Suite 3290  
Arden Hills, MN 55126-6198  
U.S.A.

To submit an incident report, you can use the UGS GTAC online support tools at the following URL:

<http://support.ugs.com>

## Software Copyright and Trademark Notices

© 2006 UGS Corp. All Rights Reserved. This software and related documentation are proprietary to UGS Corp.

UGS and Teamcenter are registered trademarks or trademarks of UGS Corp. or its subsidiaries in the United States and in other countries.

Java and all Java-based marks are trademarks or registered trademarks of Sun Microsystems, Inc. in the United States and other countries.

All other trademarks or registered trademarks belong to their respective holders.

---

## Chapter

# *1 Introduction*

Manually Setting the Teamcenter Engineering Environment .....	1-1
Log Files .....	1-3





---

## Chapter

# 1 *Introduction*

---

This chapter describes the environment and log files associated with Teamcenter Engineering command line utilities.

---



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 Engineering application root directory.

## Manually Setting the Teamcenter Engineering Environment

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

**IMAN\_ROOT** and **IMAN\_DATA** are the only environment variable settings required to run the core Teamcenter Engineering application. These variables can be set automatically at login. However, there are several standalone utilities such as **install** and **clearlocks** that also require that the entire Teamcenter Engineering environment be set.

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



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

## UNIX Systems

Manually configuring the Teamcenter Engineering environment on UNIX systems requires sourcing the **iman\_profilevars** and **iman\_cshvars** scripts. To manually set the Teamcenter Engineering environment, enter one of the following sets of commands:

Bourne/Korn shell:

```
IMAN_ROOT=/usr/iman2005; export IMAN_ROOT
IMAN_DATA=/usr/iman2005/imandata; export IMAN_DATA
. $IMAN_DATA/iman_profilevars
```

C shell:

```
setenv IMAN_ROOT /usr/iman2005
setenv IMAN_DATA /usr/iman2005/imandata
source $IMAN_DATA/iman_cshvars
```

Sourcing the **\$IMAN\_DATA/iman\_cshvars** file creates a **csh** subshell in which Teamcenter Engineering environment variables are set.

## Windows Systems

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

```
set IMAN_ROOT=c:\iman2005
set IMAN_DATA=c:\iman2005\imandata
call %IMAN_DATA%\iman_profilevars
```



This documentation consistently uses UNIX syntax, when referring to environment variable names, providing command line examples, and so on). Windows users must make the following adjustments:

Variable names	UNIX environment variable names are often prefixed with the dollar sign (\$); Windows NT environment variable names must be enclosed within two percent signs (%) in similar circumstances. For example:				
	<table><tr><td>UNIX</td><td>\$My_Environment_Variable</td></tr><tr><td>Windows</td><td>%My_Environment_Variable%</td></tr></table>	UNIX	\$My_Environment_Variable	Windows	%My_Environment_Variable%
UNIX	\$My_Environment_Variable				
Windows	%My_Environment_Variable%				
Path names	UNIX path names use forward slashes (/) for subdirectories; Windows NT path names use backslashes (\) for subdirectories. For example:				
	<table><tr><td>UNIX</td><td>My_Dir/My_Subdir/my_file</td></tr><tr><td>Windows</td><td>My_Dir\My_Subdir\my_file</td></tr></table>	UNIX	My_Dir/My_Subdir/my_file	Windows	My_Dir\My_Subdir\my_file
UNIX	My_Dir/My_Subdir/my_file				
Windows	My_Dir\My_Subdir\my_file				

## Log Files

Teamcenter Engineering 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 Engineering applications.



In this context, *application* means either a Teamcenter Engineering application (such as My Navigator or Product Structure Editor) or a Teamcenter Engineering utility (such as **clearlocks** or **schema\_browser**).

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 **.iman\_env** preference file stored in the **IMAN\_DATA** directory. User, group, and role preferences control application log files. These preferences can be set in copies of the **.iman\_env** preference file stored in the directory pointed to by the **IMAN\_USER\_PFILE**, **IMAN\_GROUP\_PFILE**, and **IMAN\_ROLE\_PFILE** site preferences, respectively.

### Best Practices

UGS 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 re-created after old log files are deleted. Log files can be deleted while users are logged into Teamcenter Engineering.

UGS 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 Engineering be able to write to the **\$IMAN\_LOG** directory if any system logging is enabled. Otherwise, the system repeatedly attempts to write to this directory and performance is affected.

## 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 Engineering system administration objects when they are created, modified, deleted, or released:

- **user**
- **group**
- **group member creation**
- **dataset type**
- **tool creation**
- **tool deletion**
- **volume creation**
- **volume deletion**
- **modification**
- **move**
- **grant access**
- **revoke access**

The protected **IMAN\_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 **IMAN\_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, the **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 **IMAN\_Security\_Logging** site preference enables security logging. Security logging creates a record of invalid access of Teamcenter Engineering objects and writes the data to the **\$IMAN\_LOG/security.log** file. Enabling security logging also requires creating a file named **security.log** in the **IMAN\_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 **IMAN\_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. Application log files are stored in the **\$IMAN\_TMP\_DIR** directory. The **IMAN\_Application\_Logging** preference controls the logging of journal and monitor 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 PSE log file names:

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

The following application log files are used by UGS support and development to troubleshoot and debug Teamcenter Engineering:Teamcenter Engineering

- Journal files (**.jnl**)  
Contains diagnostic information and is intended for UGS use only.
- Monitor files (**.mon**)  
Contains a record of user interface actions such as keystrokes and mouse clicks.
- Syslog files (**.syslog**)  
Contains diagnostic information and is intended for UGS use only.
- Object log files (**.log**)  
Contains a record of Teamcenter Engineering objects (users, groups, volumes, and so on) created, modified, or deleted during the application session.

## Application Logging Preferences and Variables

The following preferences and environmental variable control application logging:

- **IMAN\_Application\_Logging** preference  
Enables or suppresses application logging. This option can also be set in the rich client interface using the **Options** option on the **Edit** menu.  
This preference only enables or suppresses application logging for journal and monitor files; **syslog** files are always created and cannot be suppressed.
- **IMAN\_Journalling** preference  
Globally enables or suppresses creation of all journal files independently of monitor and **syslog** files. This option can also be set in the rich client interface using the **Options** option on the **Edit** menu.
- **IMAN\_Journal\_Modules** preference  
Determines which Teamcenter Engineering modules create application journal files when the **IMAN\_Journalling** preference is set to **On**.
- **IMAN\_KEEP\_SYSTEM\_LOG** preference  
Determines whether to retain new **syslog** files if they do not contain any errors.
- **IMAN\_POM\_JOURNALLING** preference  
Enables or suppresses extended application logging.
- **IMAN\_TMP\_DIR** preference  
Defines a temporary directory for storing application log files.
- **CLASSPATH** environment variable  
Defines the directory for storing the Portal 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
```

---

## Chapter

# 2 *Configuration Utilities*

Preferences Manager .....	2-1
preferences_manager .....	2-2
Access Manager .....	2-6
am_install_tree .....	2-7
Business Modeler .....	2-9
apply_naming_rule .....	2-10
business_rules_dtdxml2plmxml .....	2-16
deepcopyrules_migration .....	2-18
grm .....	2-19
import_export_business_rules .....	2-21
install_bmf_rules .....	2-24
install_type_display_rules .....	2-26
migrate_action_rules_to_bmf .....	2-27
migrate_alias .....	2-29
migrate_complex_property_rules .....	2-35
Change Management .....	2-36
create_change_types .....	2-37
ClearCase Integration With Teamcenter Engineering .....	2-39
install_scm_ClearCase .....	2-40
eIntegrator .....	2-41
tc_config_attr_mapping .....	2-42
Organization .....	2-44
ldapsync .....	2-45
Object Validation .....	2-47
create_validation_data_objects .....	2-48
Teamcenter Reporting .....	2-50
install_default_report_designs .....	2-51
rep_batch_report .....	2-53
Teamcenter Engineering Data Integration Services Adapter .....	2-56
runJtCacheClean .....	2-57
runJtCacheInit .....	2-58
runJtCachePopulator .....	2-59
Teamcenter Linking .....	2-60

---

TcEnggWolfAppRegistryUtil .....	2-61
---------------------------------	------



---

## Chapter

# 2 *Configuration Utilities*

---

This chapter describes the utilities used to configure Teamcenter Engineering.

---

## Preferences Manager

This section describes the utility used to migrate, import, and export preferences in to the database.

---

## preferences\_manager

---

**DESCRIPTION**

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-id -p=password -g=group
[-mode=import -file=file-name -scope=SITE | GROUP | ROLE | USER
-action=SKIP | OVERRIDE | MERGE] [-mode=import
-preference=preference-name -scope=SITE | GROUP | ROLE | USER
-values=comma-separated-values -action=SKIP | OVERRIDE | MERGE]
[-mode=export -out_file=output-file-name
-file=file-containing-preferences] [-mode=export
-out_file=output-file-name -scope=SITE | GROUP | ROLE | USER]
[-mode=generatexml -file=legacy-file-name
-scope=SITE | GROUP | ROLE | USER -out_file=output-file-name]
[-mode=remove -scope=SITE | GROUP | ROLE | USER
-preferences=comma-separated-preference-names] [-mode=clear
-scope=SITE | GROUP | ROLE | USER]
```

**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.



If Teamcenter 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 Engineering database. If you do not supply these arguments, the utility attempts 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-name* 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.

**-mode**

Specifies one of the following modes:

**=import**

Imports preferences in to the database using one of the follow command combinations:

```
[ -mode=import -file=file-name -scope=SITE|GROUP|ROLE|USER
  -action=SKIP|OVERRIDE|MERGE ]
```

```
[ -mode=import -preference=preference-name -scope=SITE|GROUP|ROLE|USER
  -values=comma-separated-values -action=SKIP|OVERRIDE|MERGE ]
```

**=export**

Exports preferences to a specified output file. When this mode is used, the **-out\_file** and either the **-file** or **-scope** option must be specified.

**=generatexml**

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

**=remove**

Removes the specified preferences from the database under the specified scope.

**=clear**

Clears all preferences for the specified scope. For example, if the scope is specified as **USER**, all preferences for the logged in user are cleared. Similarly, if the scope is specified as **ROLE**, all preferences for the logged in role are cleared.

**-file**

Specifies either a legacy preference file or an XML file when used with the **-mode=import** option. All preferences within the specified file are moved to the database.

Specifies an input file containing the list of preferences and their scopes when used with the **-mode=export** option. The preferences must be in the following format:

```
preference1
preference2
preference3
```

Specifies a legacy preference file if used with the **-mode=generatexml** option.

**-out\_file**

Specifies a file to which the preferences are exported.

This option must be used in conjunction with the **-mode=export** option.

**-preference**

Specifies the name of a preference to be imported.

**-values**

Specifies a comma-separated list of values for the preference being imported.

**-scope**

Specifies the scope of the preference being imported when used with the **-mode=import** option. Valid values for this option are **SITE | GROUP | ROLE | USER**.

Specifies the legacy preference file (site, group, role, or user preference file) when used with the **-mode=generatexml** option.

**-action**

Specifies the action taken in the event that a preference being imported already exists in the database. Valid values are **SKIP | OVERRIDE | MERGE**.

This option must be used in conjunction with the **-mode=import** option.

**-h**

Displays help for this utility.

**ENVIRONMENT**

As specified in [Manually Setting the Teamcenter Engineering Environment](#) in chapter 1, [Introduction](#).

**FILES**

As specified in [Log Files](#) in chapter 1, [Introduction](#).

**RESTRICTIONS**

The utility must be run by a user with administrative privileges when it is run under the following conditions.

- When the **-mode=migrate** option is used
- When using the **-mode=import** option with a specified scope of **SITE**, **GROUP**, or **ROLE**

**EXAMPLES**

- To migrate the preferences from the **tc\_preferences.xml** file in to the database, enter the following command on a single line:

```
preferences_manager -u=infodba -p=password -g=dba -mode=migrate
```

- To generate the site preferences XML file from the legacy site preference file, enter the following command on a single line:

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

- To generate the user preferences XML file from the legacy user preference file (for example, **smith.iman\_env**), enter the following command on a single line:

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

- To generate the group preferences XML file from the legacy group preference file (for example, **design.iman\_env**), 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
```

- To generate the role preferences XML file from the legacy role preference file (for example, **designer.iman\_env**), enter the following command on a single line:

```
preferences_manager -u=user-id -p=password -g=dba -mode=generatexml
-file=IMAN_DATA\rpfiles\designer.iman_env -scope=ROLE
-out_file=C:\temp\designer.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.iman_env -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 -preference=TestPreference -scope=SITE
-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
```



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

- To export all group preferences in the database for the default 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
```

## **Access Manager**

This section describes the utility used to install a default rule tree that can be used as the basis for access rule maintenance.

---

## am\_install\_tree

---

**DESCRIPTION**

Installs an AM rule tree at your site. Teamcenter Engineering supplies a default rule tree, which provides a starting point for creating rules at your site.

**SYNTAX**

```
am_install_tree -u=user-id -p=password -g=group -path=file-name
[-mode={replace_all | replace_tree | no_replace}]
```

**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.



If Teamcenter 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 Engineering database. If you do not supply these arguments, the utility attempts 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-name* 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.

**-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.

ENVIRONMENT	As specified in <i>Manually Setting the Teamcenter Engineering Environment</i> in chapter 1, <i>Introduction</i> .	
FILES	As specified in <i>Log Files</i> in chapter 1, <i>Introduction</i> .	
RESTRICTIONS	Requires Teamcenter Engineering administrator privileges.	
RETURN VALUES		
	<b>Return value upon success</b>	0
	<b>Return value upon failure</b>	1



## **Business Modeler**

The utilities used to apply naming rules, import and export business rules, and migrate complex property rules are described in this section.

## apply\_naming\_rule

### DESCRIPTION

Creates and maintains naming rules for the naming rules checker.

### SYNTAX

```
apply_naming_rule [-u=user-id -p=password -g=group] [-v]
-action=create_rule -rule=name-of-rule [-pattern=string]
[-autogen=y/n[-init=string] [-max=string]]
```

```
apply_naming_rule [-u=user-id -p=password -g=group] [-v]
-action=delete_rule -rule=name-of-rule
```

```
apply_naming_rule [-u=user-id -p=password -g=group] [-v]
-action=attach_rule -rule=name-of-rule -type=type-name
-property=property-name [-case=l | u | m]
```

```
apply_naming_rule [-u=user-id -p=password -g=group] [-v]
-action=detach_rule -rule=name-of-rule -type=type-name
-property=property-name
```

```
apply_naming_rule [-u=user-id -p=password -g=group] [-v] -action=list
```

```
apply_naming_rule [-u=user-id -p=password -g=group] [-v] -file=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.



If Teamcenter 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 Engineering database. If you do not supply these arguments, the utility attempts 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-name* 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**

Specifies one of the following actions:

**create\_rule**  
**attach\_rule**  
**detach\_rule**  
**delete\_rule**  
**list**

**-type**

Specifies the object type to which the rule is added.

**-property**

Specifies the object type property to which to add the rule.

**-case**

Specifies whether the object type/property field is converted to uppercase, lowercase, or mixed case letters.

**l**

Specifies lowercase letters.

**u**

Specifies uppercase letters.

**m**

Specifies mixed-case letters.

**-rule**

Specifies the rule name when attaching or creating rules.

**-pattern**

Specifies a pattern for the rule. The pattern can be repeated.

**-autogen**

Controls automatic generation based on valid patterns. Scans pattern fields for strings that can be incrementally increased and assigns counters. Valid values are **y** and **n**.

**-init**

Sets the initial value for a field. Used only when automatic generation is enabled. If a system variable is used in the first pattern for the field, the initial value should contain the current user's valid value, for example:

```
pattern=nn"${ROLE}"nn
ROLE=DBA
init=00DBA01
```

**-max**

Sets the maximum value for a field. Used only when automatic generation is enabled for the field. If a system variable is used in the first pattern of the field, the maximum value should contain the current user's valid value, for example:

```
pattern=nn"${ROLE}"nn
ROLE=DBA
init=99DBA99
```

**-file**

Reads the commands from a file instead of the command line. The file should contain lines with the same type of parameters as would be used on the command line. This allows multiple commands to be executed together.

**-h**

Displays help for this utility.

**-v**

Specifies verbose output.

**ENVIRONMENT**

As specified in [Manually Setting the Teamcenter Engineering Environment](#) in chapter 1, [Introduction](#).

**FILES**

As specified in [Log Files](#) in chapter 1, [Introduction](#).

**RESTRICTIONS**

Requires Teamcenter Engineering administrator privileges.

The Teamcenter Engineering preferences, **IMAN\_ugmaster\_name\_separator**, and **FLColumnCatIVFSeparatorPref** must be set to the same value or the default creation of **UGMASTER** dataset names are affected.

If the **IMAN\_ugmaster\_name\_separator** preference is set to (-), the dataset pattern must contain a dash (-). For example, **nnnnn"-nnn**. If the **IMAN\_ugmaster\_name\_separator** preference is set to (/), the dataset pattern must contain a forward slash character. For example, **nnnnn"/A**.

**EXAMPLES**

- The following example creates a naming rule:

```
apply_naming_rule -action=create_rule -rule=ItemId
-pattern="\CORP-\nnn\"-PART-\AANN" -pattern="\NONCORP\nnnn"
```

- The following example attaches the naming rule to the **item\_id** field of the item of type **Document**:

```
apply_naming_rule -action=attach_rule -rule=ItemId
-type=Document -property=item_id -case=upper
```

- The following example detaches the naming rule from the **item\_id** field of the **Document** item type:

```
apply_naming_rule -action=detach_rule -rule=ItemId
-type=Document -property=item_id
```

- The following example deletes a naming rule:

```
apply_naming_rule -action=delete_rule -rule=ItemId
```

- The following example creates an automatically generating naming rule:

```
apply_naming_rule -action=create_rule -rule=ComId
-pattern='nnn'-PART-'AA' -autogen=y
-init='001-PART-AA' -max='999-PART-ZZ'5.
```

- The following example specifies patterns using one of the following command formats:

```
-pattern="\CORP-\nnn\"-PART-\AANN" -pattern="\NONCORP\nnnn"
-pattern='CORP-nnn'-PART-AANN' -pattern='NONCORPnnnn'
```

Patterns may consist of any of the following:

Character	Matches
<b>&amp;</b>	Alphanumeric
<b>X</b>	Uppercase alphanumeric
<b>x</b>	Lowercase alphanumeric
<b>N</b>	Numeric
<b>n</b>	Numeric
<b>@</b>	Alpha
<b>A</b>	Uppercase alpha
<b>a</b>	Lowercase alpha
<b>"string"</b>	String delimiters
<b>?</b>	Any single character
<b>\</b>	Escape the next character to have literal meaning.
<b>\${system-variable}</b>	Substitute Teamcenter Engineering system variable value. Valid values for system variables are as follows:

#### **GROUP**

Specifies user's current group.

#### **ROLE**

Specifies user's current role.

#### **USERID**

Specifies user's ID.

#### **USERNAME**

Specifies user's name.

#### **SITENAME**

Specifies the site name.

#### **All:preference**

Specifies a preference.

#### **GROUP:preference**

Specifies a group preference.

#### **ROLE:preference**

Specifies a role preference.

Character	Matches
	<p><b>SITE:preference</b> Specifies a site preference.</p> <p><b>USER:preference</b> Specifies a user preference.</p>
<code>{var_type:var_name}</code>	<p>Delimiters for literal variables, for example:</p> <p><code>{Type:Name}</code></p> <p>Values in the list are treated as quoted strings, typically, <code>{LOV:Name}</code>.</p> <p>Valid values for the variable type are as follows:</p> <p><b>LOV</b> Specifies a list of values</p> <p><b>RULE</b> Specifies <b>NAME_rules</b>.</p>
<code>[var_type:pattern_name]</code>	<p>Delimiters for pattern variables, for example:</p> <p><code>{Type:name}</code></p> <p>Values in this list are treated as patterns, typically, <code>[RULE:Name]</code>.</p> <p>Valid values for the variable type are as follows:</p> <p><b>LOV</b> Specifies a list of values</p> <p><b>RULE</b> Specifies <b>NAME_rules</b>.</p>

- Explanation of backslash characters (\) and the use of braces and brackets.  
The backslash is used only if you use delimiters inside of delimiters, for example:  

```
"A\"Special\"Name"
```

which matches **A “Special” Name**.  
Otherwise, it is impossible to use a delimiter as part of a name. The same is true of brackets ([ ]) and braces ({}).  
**{RULE:Test\ \$Now\ {a\} \${ROLE}}** with the current role of **Designer** would look for a definition named:  

```
Test$Now{a}Designer
```

The brackets ([ ]) indicate pattern strings, such as **n** for 0 through 9. The braces ({} ) indicate literal strings, such as **abc** for "abc".

- Nested rules.

The following rules work together to define a number field with a required decimal point:

```
-action=create_rule -rule=RealNum -pattern=[RULE:IntegerNum}
"."[RULE:IntegerNum]

-action=create_rule -rule=IntegerNum -pattern=n
-pattern=n[RULE:IntegerNum]
```

- Valid property names.

Class	Property Name	Create	Modify	Comment
Item	<b>item_id</b>	<b>X</b>	<b>X</b>	Item ID
Item	<b>item_revision_id</b>	<b>X</b>	<b>X</b>	Item revision
Item	<b>object_name</b>	<b>X</b>	<b>X</b>	Item name
Item	<b>object_desc</b>		<b>X</b>	Item description
Dataset	<b>object_name</b>	<b>X</b>	<b>X</b>	Dataset name
Dataset	<b>object_desc</b>		<b>X</b>	Dataset description
Dataset	<b>pubr_object_id</b>		<b>X</b>	Dataset ID
Dataset	<b>rev</b>		<b>X</b>	Dataset revision
Form	<b>object_name</b>	<b>X</b>	<b>X</b>	Form name
Form	<b>object_desc</b>	<b>X</b>	<b>X</b>	Form description
Form	<i>form_field</i>		<b>X</b>	Form fields

## business\_rules\_dtdxml2plmxml

### DESCRIPTION

Converts an XML file conforming to a specified DTD into a valid PLM XML file pertaining to a specified PLM XML schema definition file.

This utility takes the location of the XML input file, generates the PLM XML file, and saves it to the specified location.

### SYNTAX

**business\_rules\_dtdxml2plmxml** [-u=*user-id* -p=*password* -g=*dba*  
-infile=*input-xml-file* -outfile=*outputfile-location* -dtdversion=*v9* | *v9.1* | *preV9*

### 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.



If Teamcenter 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 Engineering database. If you do not supply these arguments, the utility attempts 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-name* 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.

#### -infile

Specifies the XML file to be converted into PLM XML format.

#### -outfile

Specifies the path to the file to which the output PLM XML is generated.

#### -dtdversion

Specifies the version of the **BusinessRules** DTD file on which the input business rules XML file is based.

### ENVIRONMENT

As specified in [Manually Setting the Teamcenter Engineering Environment](#) in chapter 1, [Introduction](#).

### FILES

As specified in [Log Files](#) in chapter 1, [Introduction](#).



**RESTRICTIONS**

None.

**EXAMPLES**

Enter the following command on a single line to convert a business rules XML file into an equivalent PLM XML file:

```
%IMAN_ROOT\bin\business_rules_dtdxml2plmxml -u=infodba -p=password  
-g=dba -infile=c:\brules.xml -outfile=c:\brules.plmxml  
--dtdversion=prev9
```

**Sample Configuration File c:\data\business\_rules\_dtdxml2plmxmlcfg.xml**

```
<?xml version="1.0" encoding="UTF-8"?>  
<PLMXML xmlns=http://www.plmxml.org/Schemas/PLMXMLSchema  
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"  
xsi:schemaLocation="http://www.plmxml.org/Schemas/PLMXMLSchema  
../PLMXMLSchema.xsd"  
schemaVersion="3.1415926535897932384626433832795"  
author="Kavuru" time="14:20-05:00" date="">  
<UserData>  
<UserValue title="InputDataDefinitionFile"  
value="C:\imanv9\imandata\BusinessRules.dtd"/>  
<UserValue title="OutputDataDefinitionFile"  
value="C:\imanv9\imandata\PLMXMLBusinessSchema.xsd"/>  
</UserData>  
</PLMXML>
```

---

**deepcopyrules\_migration**

---

**DESCRIPTION**

Migrates deep copy rules existing in a Teamcenter Engineering 8.1 database preference file to 2005 SR1 database rules.



This utility must be run against a Teamcenter Engineering 8.1 database.

**SYNTAX**

**deepcopyrules\_migration** [-u=*user-id* -p=*password* -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.



If Teamcenter 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 Engineering database. If you do not supply these arguments, the utility attempts 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-name* 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.

**-h**

Displays help for this utility.

**ENVIRONMENT**

The **IMAN\_DATA** variable must be set, as specified in *Manually Setting the Teamcenter Engineering Environment* in chapter 1, *Introduction*.

**FILES**

As specified in *Log Files* in chapter 1, *Introduction*.

**RESTRICTIONS**

This utility must be run by an administrator and must be run when no users are logged in to the database.

**EXAMPLES**

To migrate deep copy rules from preference files to database storage, enter the following command:

```
deepcopyrules_migration -u=infodba -p=infodba -g=dba
```

---

**grm**


---

**DESCRIPTION**

Provides means to manipulate constraint rules.

**SYNTAX**

```
grm -help [-find primaryObj secondaryObj relationObj] [-find
primaryObj secondaryObj relationObj priCar secCar constraint]
[-match primaryObj secondaryObj relationObj] [-redun] [-redun
primaryObj secondaryObj relationObj] [-delete_all] [-delete primaryObj
secondaryObj relationObj] [-create primaryObj secondaryObj
relationObj priCar secCar constraint] [-create_secured_iman_rules]
[-file=file-name] -user=user-name -password=password
```

**ARGUMENTS****-find**

Lists all rules in the table.

**-find primaryObj secondaryObj relationObj priCar secCar constraint**

Lists a particular rule. Each parameter, with the exception of the **constraint** parameter, can accept **NULLTAG**.

**-match primaryObj secondaryObj relationObj****-redun[dancy]****inputFile**

Specifies an input file of predetermined rules.

**-delete priObj secObj relObj**

Deletes a particular rule.

**-delete\_all**

Deletes all entries in the rule table.

**-create\_secured\_item\_rules****-create priObj secObj relObj priCar secCar constraints**

Creates a new rule.

**-h**

Displays help for this utility.

**NOTES**

- The **relationObj** parameter for all options can accept **GRM\_match\_all**.
- When using the **-find** option, object names can accept **NULLTAG**.
- The **priCar** and **secCar** parameters can accept **n** or **N** to indicate unlimited cardinality.
- User and password parameters must be placed at the end of the command line.

EXAMPLES

- The following example, creates a new rule for the triplet **ItemRevision**, **Text**, and **IMAN\_specification**. The primary cardinality is **2**, the secondary cardinality is **3**, and the bitwise constraint is **41**.

```
grm -create ItemRevision Text IMAN_specification 2 3 41
```

- The following example, lists all rules where the primary object is **ItemRevision**, regardless of the secondary and relation objects.

```
grm -list Item Revision NULLTAG NULLTAG
```

- Uses the **ruleFile.txt** file to populate the table. The file contains the rules in the following table.

Item	POM_object	IMAN_specification	n	n	41
ItemRevision	UGMASTER	IMAN_specification	n	1	81
ItemRevision	PSBOMViewRevision	GRM_match_all	0	0	0
ItemRevision	POM_object	IMAN_specification	n	n	44

```
grm ruleFile.txt
```

Given the following constants, the constraint symbol (through the utility or through an ITK program) can be constructed according to the following formula:

Changeable + Restricted Attachability + Restricted Detachability →81  
 Add Only + Unrestricted Attachability + Unrestricted Detachability →42

<b>GRM_changeable</b>	<b>1</b>
<b>GRM_add_only</b>	<b>2</b>
<b>GRM_frozen</b>	<b>4</b>
<b>GRM_attach_unrest</b>	<b>8</b>
<b>GRM_attach_rest</b>	<b>16</b>
<b>GRM_detach_unrest</b>	<b>32</b>
<b>GRM_detach_rest</b>	<b>64</b>

## import\_export\_business\_rules

### DESCRIPTION

Imports and exports business rules to and from the Teamcenter Engineering database. To update different sites with business rules configured at one particular site, the business rules can be exported to an XML file that can be transferred to other sites where the business rules configuration can then be imported from the XML file.

Export operations can include all business rules configured at a site or a specific set of business rules. Business rules can be classified as naming rules, compound properties, canned methods, GRM cut/paste rules, and deep copy rules. Each classification has one or more corresponding rule classes. The rule class names are given as input. Naming rules have two rule classes: **NameField** and **NameRule**. When exporting rules, the rule class names can be specified as comma-separated value input to the **-rule** argument. To export all business rules **ALL** is accepted as a flag input to the **-rule** argument.

Import operations import rules information from an XML file and configure them at the site. Before performing the import operation, the current configuration is exported to a backup file which is stored in a temporary directory. Importing business rules overwrites the existing business rule configuration at a site unless one of the merge options are used.



UGS recommends that the current configuration be exported and merged with the XML file to be imported before importing the new rules.

This is necessary to avoid losing existing rules and to remove rules that do not need to be retained.



Importing an XML file containing business rules information without first merging it with the XML file containing the existing configuration could cause loss of data.

### SYNTAX

```
import_export_business_rules [-u=user-id] [-p=password]
[-g=group] -action=import | export | delete [-action=import
[-merge_add | -merge_replace | -replace]] [-type=type-name]
-file=complete-path-of-the-XML-file [-rule=valid-rule-class-name]
| -h | -list_business_rules
```

### 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.



If Teamcenter 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 Engineering database. If you do not supply these arguments, the utility attempts 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-name* 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**

Defines the action to be performed: **import**, **export**, or **delete**. The **delete** option deletes all unsecured business rules from the database.

**-file**

When exporting, specifies the full path of the XML file to which business rules are exported. When importing, specifies the full path of the XML file from which business rules are imported.

**-rule**

Export only. Specifies whether a specific set of rule types are to be exported or whether all the rules in the database are exported. A set of rule types can be specified as comma-separated values. Valid values are as follows:

**NameRule**

Naming rules

**NameField**

Naming rules

**DeepCopyRule**

Deep copy rules

**ImanGRM**

GRM cut/paste rules

**TypeCannedMethod**

Canned methods

**ImanCompoundPropDef**

Compound properties

**HideTypeRule**

Type display rules

**PropRule**

Property rules

**ALL**

All business rules

**-h**

Displays help for this utility.

**-merge\_add**

Import only. Merges imported rules into existing rules without replacing existing rules.

**-merge\_replace**

Import only. Merge imported rules into existing rules, replacing duplicates.

**-replace**

Import only. Replaces existing rules with imported rules.

**-type**

Specifies that rules referring to the specified type are exported. This argument is ignored with importing rules.

**-list\_business\_rules**

Displays a list of valid rule class names.

**ENVIRONMENT**

As specified in [Manually Setting the Teamcenter Engineering Environment](#) in chapter 1, [Introduction](#).

**FILES**

As specified in [Log Files](#) in chapter 1, [Introduction](#).

**RESTRICTIONS**

None.

**EXAMPLES**

- To import business rules from the XML file into the Teamcenter Engineering, enter the following command on a single line:  

```
import_export_business_rules -u=infodba -p=infodba -g=dba
-action=import -file=%IMAN_DATA%\BusinessRules.xml
```
- To import a few additional business rules from the XML file into Teamcenter Engineering, enter the following command on a single line:  

```
import_export_business_rules -u=infodba -p=infodba -g=dba
-action=import -merge_add -file=%IMAN_DATA%\BusinessRules.xml
```
- To export only business rules specific to **TypeCannedMethod**, enter the following command on a single line:  

```
import_export_business_rules -u=infodba -p=infodba -g=dba
-action=export -file=%IMAN_DATA%\BusinessRules.xml
-rule=TypeCannedMethod
```
- To export business rules specific to naming rules configured in the Teamcenter Engineering database, enter the following command on a single line:  

```
import_export_business_rules -u=infodba -p=infodba -g=dba
-action=export -file=%IMAN_DATA%\BusinessRules.xml
-rule=NameRule,NameField
```
- To export all business rules configured in the Teamcenter Engineering database, enter the following command on a single line:  

```
import_export_business_rules -u=infodba -p=infodba -g=dba
-action=export -file=%IMAN_DATA%\BusinessRules.xml -rule=ALL
```

---

**install\_bmf\_rules**

---

**DESCRIPTION**

Installs extensions and operations for all messages defined in the system.



This utility must be run as part of the installation or upgrade process.

**SYNTAX**

**install\_bmf\_rules** **-u**=*user-id* **-p**=*password* **-g**=*group-name* [**-h**] [**-dry\_run**]  
[**-print**] [**-upgrade** | **-refresh**] [**-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.



If Teamcenter 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 Engineering database. If you do not supply these arguments, the utility attempts 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-name* 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.

**-dry\_run**

Checks all extensions and operations defined in the database against the static lists defined in the core system and generates a report.

**-print**

Prints the contents of the database.

**-upgrade**

Updates the internal extensions and operations from static lists by updating only changes from the static lists. If new operations are added to the static list, the utility adds the operations and extensions.

**-refresh**

Deletes all internally defined extensions and operations and repopulates them from the static lists present in the code.



**-v**

Verbose mode. Prints output to the console.

**-h**

Displays help for this utility.

**ENVIRONMENT**

As specified in *Manually Setting the Teamcenter Engineering Environment* in chapter 1, *Introduction*.

**FILES**

As specified in *Log Files* in chapter 1, *Introduction*.

**RESTRICTIONS**

This utility can only be run by a user with **dba** privileges.

**EXAMPLES**

- To list all extensions and operations in the system, enter the following command on a single line:

```
install_bmf_rules -u=infodba -p=infodba -g=dba -print
```

- To upgrade all internal extensions and operations and print messages to the console, enter the following command on a single line:

```
install_bmf_rules -u=infodba -p=infodba -g=dba -upgrade -v
```

- To repopulate all internal extensions and operations, first deleting all the rules and then populating them from the static lists present in the code for the module, enter the following command on a single line:

```
install_bmf_rules -u=infodba -p=infodba -g=dba -refresh -v
```

---

**install\_type\_display\_rules**

---

**DESCRIPTION**

Creates system-level type display rules.

The display of the following object types can be controlled:

- **EngChange**
- Form
- Dataset
- **ChangeTypeData**

**SYNTAX**

**install\_type\_display\_rules** [-u=*user-name* -p=*password* -g=*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.



If Teamcenter 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 Engineering database. If you do not supply these arguments, the utility attempts 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-name* 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.

**-h**

Displays help for this utility.

**RESTRICTIONS**

This utility must be run by an administrator.

**EXAMPLES**

To install the system-level type display rules, enter the following command on a single line:

```
install_type_display_rules -u=user-name -p=password -g=dba
```

---

## migrate\_action\_rules\_to\_bmf

---

**DESCRIPTION**

Migrates action rules defined for a site and converts them to extension rules.



Registered custom canned methods must be migrated to extensions by creating the extension manually in the Business Modeler and then running the **migrate\_action\_rules\_to\_bmf** utility. If the utility is run without first creating the extensions in the database, the action rules related to custom canned methods remain in the database.

**SYNTAX**

**migrate\_action\_rules\_to\_bmf** **-u**=*user-name* **-p**=*password* **-g**=*group-name*  
**[-list]** **[-obsolete]** **[-log**=*log-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.



If Teamcenter 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 Engineering database. If you do not supply these arguments, the utility attempts 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-name* 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.

**-list**

Lists the action rules configured for a site.

**-obsolete**

Converts the action rules to extension rules.

**-log**

Writes a report to the specified log file.

**-h**

Displays help for this utility.

**ENVIRONMENT**

As specified in *Manually Setting the Teamcenter Engineering Environment* in chapter 1, *Introduction*.

**FILES**

As specified in *Log Files* in chapter 1, *Introduction*.

**RESTRICTIONS**

This utility can only be run by a user with **dba** privileges.

**EXAMPLES**

- To list the action rules defined for a site and write the log file to the **ar.txt** file, enter the following command on a single line:

```
migrate_action_rules_to_bmf -u=infodba -p=infodba  
-g=dba -list -log=ar.txt
```

- To list the action rules, convert the action rules to extension rules, and create a log file reporting which action rules were converted, enter the following command on a single line:

```
migrate_action_rules_to_bmf -u=infodba -p=infodba -g=dba  
-list -obsolete -log=ar.txt
```

## migrate\_alias

### DESCRIPTION

Migrates alias data to alias identifier data without modifying or deleting the existing data. Only alias data owned by the local site can be migrated. You can qualify what data is migrated.

Data migration from alias to alias identifier includes the following steps:

- Alias properties are mapped to alias identifier properties through a user-specified mapping file.
- Alias identifier objects are created and attached to the same owners as the qualified alias objects, using the **IMAN\_aliasid** relation.



This utility does not migrate alias objects to alternate identifier objects.

### SYNTAX

```
migrate_alias
-mode=map_filefull-path-to-mapping-file
-mode=migrate_contexts
-mode=define
-process_file=full-path-to-alias-process-file
[-scope_file=full-path-to-scope-file]
-mode=migrate
-map_file=full-path-to-mapping-file
-process_file=full-path-to-process-file
-error_file=full-path-to-error-reprocessing-file
[-u=user-id -p=password -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.



If Teamcenter 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 Engineering database. If you do not supply these arguments, the utility attempts 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-name* 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.

**-mode**

Specifies the type of processing the utility will perform. This argument is required.

**validate**

Reads and validates the contents of the mapping file. All errors are written to the **stderr** file. Requires the **-map\_file** parameter.

**migrate\_contexts**

Creates IdContext objects for every alias context in the alias LOV.

**define**

Creates a process file containing references to qualified alias objects that is used in **migrate** mode. Requires the **-process\_file** parameter. The **-scope\_file** parameter is optional, and all alias objects are migrated if it is not specified.

**migrate**

Migrates the alias data. Requires the **-map\_file**, **-process\_file**, and **-error\_file** parameters.

**-map\_file**

Specification required for **validate** and **migrate** modes; otherwise ignored. This argument identifies the full path to the input mapping file.

This text file contains from (**F**) / to (**T**) data pairs (and optionally a delimiter specification) that identify how alias data is mapped to alias identifier object data. All **prop** and **context** data pair records (one or more of each are allowed) and any optional delimiter record only apply to the type data pair record that they follow in the mapping file.

```
FTYPE=from-alias-type TTYPE=to-alias-identifier-type
DELIMITER=character-value
FPROP=from-alias-property-name TPROP=to-alias-identifier-property-name
FCONTEXT=from-alias-context TCONTEXT=to-alias-identifier-context
```

The following restrictions apply to mapping file records:

- A set of mapping file records is defined as all records following a type data pair record (inclusive).
- The type of any alias object that is qualified for migration must be defined in the mapping file.
- Only data for the properties specified for each type is migrated.

If the **alias\_id** property of the alias object for any type is not specified, the migration utility defaults the mapping of this property for that alias object type to the **idfr\_id** property of the alias identifier object unless a delimiter is specified.

- When a delimiter is specified, both the **id** and **rev\_id** values (alias objects) are combined together (with the delimiter) to form the identifier ID for the new alias identifier. The delimiter specification is mutually exclusive of the mapping of the **idfr\_id** property within the same type in the mapping file.

If the **-delimiter** parameter is a colon (:), the resulting **idfr\_id** property value is **alias\_id property-value:alias\_rev\_id property value**.

#### **-process\_file**

Specification required for **define** and **migrate** modes; otherwise ignored. This parameter identifies the full path to the alias process file. This file is created (or the existing file is overwritten) in **define** mode and read (input) in **migrate** mode. This text file contains unique persistent references to alias objects that are qualified by the **-scope\_file** parameter. Each record represents a single alias object, as follows:

```
TAGSTRING=alias-object-reference // ID=alias_id REVID=alias_rev_id
```

#### **-scope\_file**

Specification optional for **define** mode; otherwise ignored. If this parameter is not specified, the scope defaults to all local alias objects. This parameter specifies the full path to the input scope file which contains one or more type scope specification records. These scope specifications qualify the local alias objects by type (wildcards allowed) for migration.

```
TYPE=from-alias-type
```

#### **-error\_file**

Specification required for **migrate** mode; otherwise ignored. This parameter identifies the full path to the alias error process file. This file is created (or the existing file is overwritten) in **migrate** mode and contains a copy of any alias process file record (from the file specified by the **-process\_file** parameter) that encountered an error during processing. The file can be used as the (input) alias process file once the errors are resolved.

#### **-h**

Displays help for this utility. No parameters or any invalid combination of parameters will also display the help for this utility.

#### ENVIRONMENT

As specified in [Manually Setting the Teamcenter Engineering Environment](#) in chapter 1, [Introduction](#).

#### FILES

As specified in [Log Files](#) in chapter 1, [Introduction](#) and the **stderr** file to which all error messages and debugging trace messages are written.

#### RESTRICTIONS

- The user running this utility must have system administration privileges.
- A system administrator must create all Teamcenter Engineering types and custom properties that are used in the mapping file before running this utility. For more information, see *Type Help*.
- Only alias data owned by the local site can be migrated.

**EXAMPLES**

- Enter the following command on a single line to validate the mapping file:

```
migrate_alias -mode=validate -map_file=/home/infodba/mymap_file.txt
-u=infodba -p=mypass -g=dba
```

- Enter the following command on a single line to create IdContext objects for all of the existing alias contexts in the alias LOV:

```
migrate_alias -mode=migrate_contexts
```

- Enter the following command on a single line to create an alias process file containing all local alias references:

```
migrate_alias -mode=define
-process_file=/home/infodba/all_aliases_file.txt
```

- Enter the following command on a single line to migrate all of the local alias references:

```
migrate_alias -mode=migrate -map_file=/home/infodba/mymap_file.txt
-process_file=/home/infodba/all_aliases_file.txt
-error_file=/home/infodba/error_file.txt
```

- Enter the following command on a single line to create an alias process file containing a qualified set of local alias references:

```
migrate_alias -mode=define
-process_file=/home/infodba/scoped_aliases_file.txt
-scope_file=/home/infodba/scope_file.txt
```

- Enter the following command on a single line to migrate the qualified set of local alias references:

```
migrate_alias -mode=migrate -map_file=/home/infodba/mymap_file.txt
-process_file=/home/infodba/scoped_aliases_file.txt
-error_file=/home/infodba/error_file.txt
```

- Enter the following command on a single line to migrate a set of local alias references that encountered errors on a previous migration run:

```
migrate_alias -mode=migrate -map_file=/home/infodba/mymap_file.txt
-process_file=/home/infodba/error_file.txt
-error_file=/home/infodba/error2_file.txt
```

**MAPPING  
FILES**

The explanation for each set of records below refers only to the objects that match the type for that set. All **-TCONTEXT=** values are created as **IdContext** objects (by default) if they do not already exist. The type for each alias reference to be migrated (read from the file specified by the **-process\_file** argument) must be represented in the mapping file or the alias reference does not migrate but is written to the file specified by the **-error\_file** argument.

A type data pair record always begins a new set. The data pairs on each record can appear in any order, but each record must have the matching set in a data pair. Also notice that the first record must be a type data pair. The value for each title is case sensitive and must appear in quotes if the value contains any spaces; otherwise the quotes are optional.



**Set 1**

**-FTYPE=Alias -TTYPE=Identifier**

This record set maps all local alias objects of type **Alias** to alias identifier objects of type **Identifier**. The **alias\_id** property value of each alias object is used (by default) as the **idfr\_id** property value of the corresponding alias identifier object. The context value for each alias object is used as the identifier context value for the corresponding alias identifier object. No other properties are migrated.

**Set 2**

**-FTYPE=AliasAgain -TTYPE=IdentifierTwo**  
**-FPROP=alias\_id -TPROP=idfr\_id**

This record set maps all local alias objects of type **AliasAgain** to alias identifier objects of type **IdentifierTwo**. The **alias\_id** property value of each alias object is used for the **idfr\_id** property value of the corresponding alias identifier object. The context value for each alias object is used as the identifier context value for the corresponding alias identifier object. No other properties are migrated.

**Set 3**

**-FTYPE=AliasOld1 -TTYPE=IdentifierNew1**  
**-FPROP=alias\_custom1 -TPROP=idfr\_id**

This record set maps all local alias objects of type **AliasOld1** to alias identifier objects of type **IdentifierNew1**. The **alias\_custom1** property value of each alias object is used for the **idfr\_id** property value of the corresponding alias identifier object. The context value for each alias object is used as the identifier context value for the corresponding alias identifier object. No other properties are migrated.

**Set 4**

**-FTYPE=AliasOld2 -TTYPE=IdentifierNew2**  
**-DELIMITER=:**  
**-FPROP=alias\_custom1 -TPROP=idfr\_custom1**  
**-FPROP=name -TPROP=name**  
**-FCONTEXT=cxtA -TCONTEXT=idcontextB**  
**-FCONTEXT=cxtABC -TCONTEXT=idcontextXYZ**

This record set maps all local alias objects of type **AliasOld2** to alias identifier objects of type **IdentifierNew2**. The **alias\_id** and **alias\_rev\_id** property value of each alias object are combined together with the delimiter for the **idfr\_id** property value of the corresponding alias identifier object. The **alias\_custom1** property value of each alias object is used for the **idfr\_custom1** property value of the corresponding alias identifier object. The **name** property value of each alias object is used for the **name** property value of the corresponding alias identifier object. The **cxtA** context value for an alias object is converted to the **idcontextB** identifier context value for the corresponding alias identifier object. The **cxtABC** context value for an alias object is converted to the **idcontextXYZ** identifier context value for the corresponding **Alias Identifier** object. All other context values for each alias object are used as the identifier context value for the corresponding alias identifier object. No other properties are migrated.

**PROCESS/ERROR  
FILE  
CONTENTS**

The process and error files share the same format. Therefore, the error file can be used as input to the next run of the **migrate\_alias** utility once problems are corrected. Each record in these files represents a unique local persistent alias object. The **-TAGSTRING=** values are for internal use by the migration utility and must not be modified. The **ID=** and **REVID=** values are for reference only, so a user can identify the alias object referenced by the process record. The error file also has an additional reference, **ERROR\_ID=** that shows the Teamcenter Engineering error code reported.

```
-TAGSTRING=... // ID=alias_id-property-value REVID=alias-revid-property-value  
ERROR_ID=Teamcenter-Engineering-error-code
```

**SCOPE FILE  
CONTENTS**

Each record in this file represents an alias type to qualify (scope) into the alias process file specified by the **-process\_file** argument. Any alias object type not defined in this file will not be qualified and will not appear as an alias object reference in the alias process file. The value for each type is case sensitive and must appear in quotes if the value contains spaces or wildcard characters; otherwise the quotes are optional. The wildcard characters must match the wildcard preference corresponding to the ID of the user that is running the utility.

```
-TYPE=Alias  
-TYPE="AliasO*"
```

---

## migrate\_complex\_property\_rules

---

**DESCRIPTION**

Creates complex property rules in the database corresponding to those stored in the preference variables prior to Teamcenter Engineering 2005 SR1.

**SYNTAX**

**migrate\_complex\_property\_rules** **-u**=*user-name* **-p**=*password* **-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.



If Teamcenter 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 Engineering database. If you do not supply these arguments, the utility attempts 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-name* 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.

**-h**

Displays help for this utility.

**RESTRICTIONS**

This utility must be run by an administrator.

**EXAMPLES**

Enter the following command on a single line to migrate complex property rules from preferences to the database:

```
migrate_complex_property_rules -u=user-name -p=password -g=group-name
```

## **Change Management**

This section describes the Change Management utilities.

---

## create\_change\_types

---

**DESCRIPTION**

Provides a means for site administrators to define one or more change type definitions from the command line. This utility creates change types according to the information contained in an input file.

**SYNTAX**

**create\_change\_types -u=user-name -p=password -g=group -f=input-file**

**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.



If Teamcenter 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 Engineering database. If you do not supply these arguments, the utility attempts 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-name* 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.

**-f**

Specifies the input file path. A sample file, **default\_change\_types.dat**, is located in the **IMAN\_DATA** directory.

**-h**

Displays help for this utility.

**ENVIRONMENT**

As specified in [Manually Setting the Teamcenter Engineering Environment](#) in chapter 1, [Introduction](#).

**FILES**

As specified in [Log Files](#) in chapter 1, [Introduction](#).

**RESTRICTIONS**

This utility can only be used to create new change type definitions. It cannot modify existing change type definitions.

**EXAMPLES**

To invoke this utility, you must enter the following command:

```
create_change_types -u=user-name -p=password -group=group -f=input-file
```

**INPUT FILE  
FORMAT**

The input file must contain information in the following format:

```
start_change_type_def Change-Type-Name
ID=Max-Length | RANGE;VALUE;STATIC or RUNNING [ | next set]
REV_ID=Max-Length | Range;Value;Static or Running [|next set]
[EFFECT_SHARED=Y or N. Default is Y]
[FORMS=Formtype 1;Formtype 2;...]
[PROCESSES=process1;process2;...]
end_change_type_def
```

**SAMPLE FILE**

Following is the contents of the **default\_change\_types.dat** file used during installation/upgrade to create CMII change types:

```
# Definition of 'CR' Change type
start_change_type_def CR
  ID=6|1-2;CR;STATIC|3-6;0001;RUNNING
  REV_ID=1|1;A;RUNNING
  FORMS=CMII CR Form
  PROCESSES=CMII Change Request
end_change_type_def

# Definition of 'CN' Change type
start_change_type_def CN
  ID=6|1-2;CN;STATIC|3-6;0001;RUNNING
  REV_ID=1|1;A;RUNNING
  FORMS=CMII CN Form
  PROCESSES=CMII Change Notice
end_change_type_def

# Definition of 'WA' Change type
start_change_type_def WA
  ID=6|1-2;WA;STATIC|3-6;0001;RUNNING
  REV_ID=1|1;A;RUNNING
  PROCESSES=CMII WA
end_change_type_def
```

## **ClearCase Integration With Teamcenter Engineering**

This section describes the utility used to install the ClearCase integration with Teamcenter Engineering.

---

**install\_scm\_ClearCase**


---

**DESCRIPTION**

Installs the ClearCase integration with Teamcenter Engineering. The utility first installs all required storage classes. If any of these already exist in the system, the utility displays a warning on the console and adds a note to the system log file.

**SYNTAX**

**install\_scm\_ClearCase** [-u=*user-name*] [-p=*password*] [-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.



If Teamcenter 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 Engineering database. If you do not supply these arguments, the utility attempts 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-name* 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.

**-h**

Displays help for this utility.

**RESTRICTIONS**

None.



## **eIntegrator**

This section describes utilities related to eIntegrator.

## tc\_config\_attr\_mapping

### DESCRIPTION

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 **IMAN\_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.

For more information about configuring attribute sharing between external data sources and Teamcenter data, see *eIntegrator Help*.

### SYNTAX

**tc\_config\_attr\_mapping** [-u=*user-id* -p=*password* -g=*group-name*  
-action=**create\_mapping** | **delete\_mapping**=*type-name.property-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.



If Teamcenter 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 Engineering database. If you do not supply these arguments, the utility attempts 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-name* 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.

#### -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 Setting the Teamcenter Engineering Environment](#) in chapter 1, [Introduction](#).

#### **FILES**

As specified in [Log Files](#) in chapter 1, [Introduction](#).

#### **RESTRICTIONS**

- This utility can be run only by users with **DBA** privileges.
- The configuration file must be prepared and validated using the **xml\_validator** utility prior to running the **tc\_config\_attr\_mapping** utility. For more information, see [xml\\_validator](#) in chapter 16, [Volume and Database Management Utilities](#).

#### **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
```

For more information about the **-create\_mapping** option, see [Arguments](#), earlier in this section.

- 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**

This section describes utilities used to configure and maintain your Teamcenter organization.

## ldapsync

### DESCRIPTION

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 *Configuration Guide*.

### SYNTAX

```
ldapsync -u user-name -ppassword -ggroup-name -M ALL_ACTIONS |  
NO_ACTIONS | UPDATE_ONLY | CREATE_ONLY | DEACTIVATE_ONLY  
| UPDATE_AND_ADD -F -l LDAP-password [-t] [-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.



If Teamcenter 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 Engineering database. If you do not supply these arguments, the utility attempts 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-name* 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.

#### **-M**

Specifies one of the following modes:

#### **ALL\_ACTIONS**

Creates, updates, and deactivates user and person data after comparing the contents of the Teamcenter database with the contents of the LDAP database.

#### **NO\_ACTIONS**

Outputs a list of users in each of the three categories: create, update, and delete, but does not perform any action against the Teamcenter database.

<b>UPDATE_ONLY</b>	Updates only the data associated with existing user entries in the Teamcenter database. Does not create or deactivate user entries.
<b>CREATE_ONLY</b>	Creates user entries for users that exist in the LDAP database but not in the Teamcenter database. Does not update or deactivate existing user entries.
<b>DEACTIVATE_ONLY</b>	Deactivates Teamcenter users that are not listed in the LDAP database.
<b>UPDATE_AND_ADD</b>	Creates new user entries and updates existing user entries but does not deactivate users.

**-F**

Forces all synchronized objects to be updated. All update synchronizations are performed without first checking that the **modifyTimestamp** value in the external directory is newer than the **last\_sync\_date** in Teamcenter.

**-l**

Specifies the LDAP user password. This value overrides the value of the **LDAP\_admin\_pw\_value** preference.

**-t**

Runs the utility in trace mode for debugging or obtaining extra information.

**-h**

Displays help for this utility.

**RESTRICTIONS**

None.

**EXAMPLES**

To update existing user data and add new user entries, enter the following command on a single line:

```
$IMAN_ROOT/bin/ldapsync -u=administrative-user -p=administrative-password  
-g=dba -l=ldap-password -M=UPDATE_AND_ADD
```

**TROUBLESHOOTING**

The following error message may be encountered when running the **ldapsynch** utility:

```
Errors from LDAP server:  
LDAP Error code: 4, error message: No Message Set  
LDAP Error2: Sizelimit exceeded
```

This message indicates that the LDAP server has more records in the requested DN than it is allowed to return. The LDAP server size limit is set in the configuration options. This limit is the same limit that controls the number of users seen in the LDAP UI. The limit must be set to at least the number of users in (including nested users) the DN specified by the **LDAP\_base\_dn** option.

## **Object Validation**

This section describes the utility used to create validation data objects.

---

**create\_validation\_data\_objects**

---

Creates validation objects in Teamcenter Engineering based on specified closure rules.

**SYNTAX**

**create\_valiation\_data\_objects**

**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.



If Teamcenter 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 Engineering database. If you do not supply these arguments, the utility attempts 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-name* 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.

**-v\_name**

Specifies the name of the validation checker, for example **mgc\_exam\_geometry** in NX Check-Mate.

**-v\_desc**

Describes the validation data.

**-v\_app**

Specifies the application in which the validation is performed, for example NX.

**-v\_util\_name**

Specifies the name of the validation utility, for example, NX Check-Mate.

**-v\_util\_cmd**

Specifies the utility command string.

**closure\_rule**

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.



**-v\_can\_group**

Specifies whether this validation can be grouped with other validations performed in the same application. Valid values are **TRUE** and **FALSE**.

**-v\_args**

Specifies the arguments associated with the validation.

**-h**

Displays help for this utility.

**ENVIRONMENT**

Requires no special environment. Teamcenter environment is sufficient to run this utility.

**FILES**

**create\_validationdata.exe**

**RESTRICTIONS**

None.

**EXAMPLES**

The following example creates validation objects for the **mqc\_exam\_geometry** validation using the NX Check-Mate utility.

```
create_validationdata.exe -v_name=mgc_exam_geometry" -v_desc="testing"  
-v_app_name="NX" -v_can_group="true" -v_util_name="NX Check-Mate"  
-v_util_cmd="testing" -clr_name="NX" -v_args="'-pim=yes'
```

## **Teamcenter Reporting**

The utilities described in this section import report designs into the Teamcenter Engineering database and generate reports in batch mode.

## install\_default\_report\_designs

### DESCRIPTION

Creates the default report designs contained in the **default\_report\_design.xml** file as part of the Teamcenter Engineering 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\_designs -u=user-id -p=password [-g=group] -file=xml-file-list**

### 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.



If Teamcenter 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 Engineering database. If you do not supply these arguments, the utility attempts 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-name* 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.

#### **-file**

Specifies the name of the XML file containing report design definitions.

#### **-h**

Displays help for this utility.

#### **-v**

Displays detailed status information.

### ENVIRONMENT

As specified in [Manually Setting the Teamcenter Engineering Environment](#) in chapter 1, [Introduction](#).

### FILES

As specified in [Log Files](#) in chapter 1, [Introduction](#).

### RESTRICTIONS

None.

### RETURN VALUES

**Return value  
upon success**            0

**Return value  
upon failure**            >1, -1

### EXAMPLES

To create default report enter the following command on a single line:

```
install_default_report_designs -u=infodba -p=infodba -g=dba
-file=$IMAN_DATA\report_writer\default_report_designs.xml
```

### XML FILE FORMAT

The format required for the XML file is shown in figure 2-1.

```
<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>
```

**Figure 2-1. XML File Format**

---

## rep\_batch\_report

---

### DESCRIPTION

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 window.

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_____
GLOBAL DEFINITIONS
User ID      : demol
Password     : demol
IMAN_ROOT    : C:\IMAN00912
IMAN_DATA    : C:\IMAN00912\imandata\demo
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 : DemoTransferMode
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~SvNJlpuVl9P7VD~
(line continued)
Latest Working~~htNJmIl8l9P7VD~x5FJmIl8l9P7VD,BKKJmIl8l9P7VD,RaDJmIl8l9P7VD
~3~zwDJ2_1$19P7VD~Latest Working~~

end_data_definition
_____End of format_____

```



A sample batch file and shell script file are located in the *IMAN\_ROOT*web/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 *Configuration Guide*.
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-name -p=password -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.



If Teamcenter 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 Engineering database. If you do not supply these arguments, the utility attempts 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-name* 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.

**ENVIRONMENT**

As specified in *Manually Setting the Teamcenter Engineering Environment* in chapter 1, *Introduction*.

**FILES**

As specified in *Log Files* in chapter 1, *Introduction*.

**RESTRICTIONS**

Requires Teamcenter Engineering administrator privileges.

## **Teamcenter Engineering Data Integration Services Adapter**

The utilities described in this section are used to initialize the JT cache service, run the JT cache populator, and clean up the JT cache.



---

## runJtCacheClean

---

**DESCRIPTION**

Physically removes orphaned files from the cache used by the JtCache Service.

UGS recommends running this tool after shutting the adapter down or before starting it. The utility may be run periodically whenever the cache size on the disk nears the maximum size specified by the cache administrator.



For information about configuring the **JtCacheClean** utility, see the *Installation on UNIX and Linux Servers* or *Installation on Windows Servers* manuals.

The JtCache Service stores JT files in a server-side cache, making them available for fast retrieval. Occasionally, files are stored on the drive but the JtCache Service is unaware of their existence. This behavior is undesirable, because these orphaned files may cause the JtCache directory to exceed its specified capacity.

For example, the application creates orphaned files when an abnormal system shutdown occurs after a file has been stored in the cache but before the updated cache index file is written to the disk. When you restart the adapter, it reads the original cache index file and the new file is not listed. Running the **JtCacheClean** utility will remove the orphaned files.



Do not store files in the directory used by the JtCache Service, because this utility deletes them. The application specifies the JtCache directory using two properties in the adapter **web.xml** file: **JtCache.World.Path** and **JtCache.Limited.Path**.

**SYNTAX**

Windows:

**DataAccess/5\_0/Products/IMANAdapter/util runJtCacheClean web.xml**

UNIX:

**DataAccess/5\_0/Products/IMANAdapter/util runJtCacheClean.sh web.xml**

**ARGUMENTS**

None.

**ENVIRONMENT**

You must be logged in as the same user under which the adapter is running to grant the utility the permissions required to delete files.

**FILES**

None.

**RESTRICTIONS**

None.

**EXAMPLES**

None.

---

**runJtCacheInit**

---

**DESCRIPTION**

Initializes user sessions for the Teamcenter Engineering DIS Adapter JT cache populator.

**SYNTAX**

Windows:

**\$IMAN\_ROOT/VisDis/IMANAdapter/util/runJtCacheInit**

UNIX:

**\$IMAN\_ROOT/VisDis/IMANAdapter/util/runJtCacheInit.sh**

**ARGUMENTS**

None.

**ENVIRONMENT**

No special environment required.

**FILES**

None.

**RESTRICTIONS**

None.

**EXAMPLES**

To initialize the populator sessions on Windows, enter the following command:

```
$IMAN_ROOT/VisDis/IMANAdapter/util/runJtCacheInit
```

When prompted, initialize the world access cache with the special Teamcenter user who has been designated for this purpose.



The user name must match the user name provided in the **JtCache.World.PopulatorUser** context parameter set using the Web Application Manager.

When prompted, initialize the limited access cache with a master user who has access to most files.

---

## runJtCachePopulator

---

**DESCRIPTION**

Runs the Teamcenter Engineering Data Integration Services Adapter JT cache populator.

The JT cache populator can be run in two modes:

- As a one-time job that runs immediately
- As a recurring batch job



If you run the **JtCachePopulator** utility as a recurring batch job, you must specify the start time and interval length in the **config.properties** file.

**SYNTAX**

Windows:

**\$IMAN\_ROOT/VisDis/IMANAdapter/util/runJtCachePopulator**

UNIX:

**\$IMAN\_ROOT/VisDis/IMANAdapter/util/runJtCachePopulator.sh**

**ARGUMENTS**

*name-of-config.properties-file*

*user-name*

*password*



If user name and password are not provided, the utility prompts you to enter them.

**ENVIRONMENT**

No special environment required.

**FILES**

For information about the **config.properties** file format and XML ObjectList file format, see the *Installation on UNIX and Linux Servers* or *Installation on Windows Servers* manual.

**RESTRICTIONS**

None.

**EXAMPLES**

To run the Teamcenter Engineering Data Integration Services Adapter JT cache populator using the **config.properties** file, enter the following command:

```
runJtCachePopulator config.properties user-name password
```

## **Teamcenter Linking**

The utility in this section is used to verify the existence and availability of an application registry and to register and unregister an application instance with the registry.

---

## TcEnggWolfAppRegistryUtil

---

### DESCRIPTION

Communicates with the application registry, either as a standalone program or within an application (for example, an installation program), to perform the following tasks:

- Check the existence or availability of an application registry.
- Register an application instance as object chooser with the application registry.
- Unregister an application instance from the application registry.

### SYNTAX

#### TcEnggWolfAppRegistryUtil

### ARGUMENTS

#### **-mode**

Identifies the task to perform. The value can be **verify**, **register**, or **unregister**. If the mode is not specified, the program exits with an error.

#### **-ip**

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.

#### **-file**

Specifies a configuration file with the application (chooser) details and application registry URL. The values are details in this file and are available as *Name=Value* pairs. (See the **TceWolfConfig** file in the **\$IMAN\_DATA** directory for names and format.) If the value is specified as **default**, the program tries to use the **TcWolfConfig** file in the **\$IMAN\_DATA** directory.

#### **-guid**

Specifies the **guid** of the Chooser application instance.

#### **-h**

Displays help for this utility.

### ENVIRONMENT

If the **-file** option is used with a value of **default**, the utility expects the **IMAN\_DATA** environment to be available.

### RESTRICTIONS

The data in the **TceConfigFile** (or a file passed using the **-file** argument) must be defined as name value pairs separated by an equal sign (=).

## EXAMPLES

The examples in this section illustrate usage of the **TcEnggWolfAppRegistryUtil**.

### Verifying the Existence of the Application Registry

The examples in this section are used to verify the existence of the application registry.

- To verify the existence of the application registry:

```
TcEnggWolfAppRegistryUtil -mode=verify -ip=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:

```
TcEnggWolfAppRegistryUtil -mode=verify -file=absolute_path_of_config_file
```

Returns failure if the application registry is not running or is unreachable.

- To test whether the application registry identified by the **ApplicationRegistryUrl** property in the default **config** file in the **\$IMAN\_DATA/TceWolfConfig** directory is active:

```
TcEnggWolfAppRegistryUtil -mode=verify -file=default
```

Returns failure if the application registry is not active or unreachable.

### Registering an Application (Chooser) Instance With the Application Registry

This example is used to register a chooser with the application registry.

- To register an application instance defined in the given **config** file:

```
TcEnggWolfAppRegistryUtil -mode=register  
-file=absolute_path_of_config_file
```

or

```
TcEnggWolfAppRegistryUtil -mode=register -file=default
```

The details must be *Name=Value* pairs. If the value of the **-file** options is defined as **default**, the program expects the following:

- **\$IMAN\_DATA** environment
- The **TceWolfConfig** file in the **\$IMAN\_DATA** directory

The application registry information is expected from the **config** file identified by the **ApplicationRegistryUrl** property. The value of this property from the file can be overridden using the **-ip=application\_registry\_url** on the command line.

### Unregistering an Application (Chooser) Instance With the Application Registry

The example in this section is used to unregister a chooser with the application registry.

- To unregister an application instance defined in the given **config** file:

```
TcEnggWolfAppRegistryUtil -mode=register  
-file=absolute_path_of_config_file
```

or

```
TcEnggWolfAppRegistryUtil -mode=register -file=default
```

The details must be *Name=Value* pairs. If the value of the **-file** options is defined as **default**, the program expects the following:

- **\$IMAN\_DATA** environment
- **TceWolfConfig** file in the **\$IMAN\_DATA** directory

The application registry information is expected from the **config** file identified by the **ApplicationRegistryUrl** property. The value of this property from the file can be overridden using the **-ip=application\_registry\_url** on the command line.





---

## Chapter

# 3 *Product Configuration Utilities*

Appearance Configuration	3-1
appr_update_manager	3-2
appr_update_supervisor	3-3
appr_update_console	3-4
appearance_updater	3-7
create_appearances	3-10
purge_baselined_item_revisions	3-12
Product Structure Utilities	3-13
Product Structure Maintenance	3-14
generate_iman_ps_path	3-15
input_notetype_default	3-17
ps_rename_bvrs	3-19
ps_traverse	3-21
ps_upload	3-25
update_bomchanges	3-29
update_project_bom	3-31
upgrade_rev_rules	3-34
Effectivity Mode	3-35
effupgrade	3-36
Product Structure Clearance Analysis	3-38
batchmode_clearance_analysis.pl	3-39
Product Structure Comparison	3-40
compare_ug_and_iman_ps	3-41
ugmanager_compare_ps	3-43



---

## Chapter

# 3 *Product Configuration Utilities*

---

This chapter describes the utilities used to manage Teamcenter Engineering product and appearance configurations.

---

## Appearance Configuration

This section describes the utilities used to create appearances and manage appearance updates. For more information, see *Appearance Configuration Help*.

---

**appr\_update\_manager**

---

**DESCRIPTION**

Launches the Appearance Update Manager. Sets up the Teamcenter Engineering 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 Setting the Teamcenter Engineering Environment](#) in chapter 1, [Introduction](#).

**RESTRICTIONS**

None.

**EXAMPLES**

None.

---

## appr\_update\_supervisor

---

### DESCRIPTION

Runs as a background process, invoking Teamcenter Engineering processes to perform appearance updates, as required.



UGS 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 Engineering 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 in automatically. If autologin is not supported at your site, you can use the alternative log in mechanism documented in the **\$IMAN\_ROOT/data/.appr\_update\_env.default** file.

### SYNTAX

**appr\_update\_supervisor**

### ARGUMENTS

None.

### ENVIRONMENT

As specified in [Manually Setting the Teamcenter Engineering Environment](#) in chapter 1, [Introduction](#).

### FILES

- **\$IMAN\_ROOT/bin/.appr\_update\_env**  
Configuration file for the Appearance Update Manager.
- **\$IMAN\_ROOT/bin/.appr\_update\_env.default**  
Example configuration file.
- **\$IMAN\_ROOT/bin/appr\_update\_supervisor\*.log**  
Supervisor log file.

### RESTRICTIONS

None.

### EXAMPLES

None.

---

## appr\_update\_console

---

### DESCRIPTION

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 -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]
```

### 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.



If Teamcenter 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 Engineering database. If you do not supply these arguments, the utility attempts 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-name* 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.

**-host**

Connects to supervisor on the specified host. Overrides setting in the **.appr\_update\_config** file.

**-port**

Connects to the supervisor through the specified communications port. Overrides 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 primary queue.

**-dump\_secondary**

Prints 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 supervisor logging level. This command is only available to system administrators.

**-clear\_log**

Requests that the Update Manager clear the log file. This command is only available 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 only available to system administrators.

**-restart**

Requests that the Update Manager restart immediately. This command is only available to system administrators.

**-restart\_now**

Restarts update manager immediately. This command is only available to system administrators.

**-prod\_queue**

Prods the supervisor with a dummy update request.

**-h**

Displays help for this utility.

**ENVIRONMENT**

As specified in *Manually Setting the Teamcenter Engineering Environment* in chapter 1, *Introduction*.

**FILES**

As specified in *Log Files* in chapter 1, *Introduction*, along with the following files:

- **\$IMAN\_ROOT/bin/.appr\_update\_env**  
Configuration file for the Appearance Update Manager.
- **\$IMAN\_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

---

**DESCRIPTION**

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 -g=group]
[-quiet]
[-nolog]
[-supervisor=host,port1,port2]
[-task=cmd [,cmd...]]
[-show_uids]
[-show_blocked]
[-since=date -item_id=item-id] ]
[-at=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.



If Teamcenter 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 Engineering database. If you do not supply these arguments, the utility attempts 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-name* 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.

**-login**

Forces the program to search for the **.appr\_update\_info** file in the **\$IMAN\_DATA**, **\$IMAN\_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 in. 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:

<b>query_update</b>	Queries the POM UID of the current appearance update.
<b>query_primary_size</b>	Queries the size of the primary appearance update queue.
<b>query_sets</b>	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.
<b>query_packages</b>	Queries the number of update packages in the current update process.
<b>query_unprocessed_packages</b>	Queries the number of update packages in the current update that have not yet been processed.
<b>process_primary</b>	Requests the processing of the next update in the primary queue. This fails if there is already an update in process.
<b>process_secondary</b>	Requests the processing of a single appearance set for the current update. This fails if there are no appearance sets that require processing.
<b>process_all_secondary</b>	Requests that the program loops until all appearance sets are processed for the current update.
<b>finish_primary</b>	Completes the final processing of an update once all secondary sets have been processed.
<b>dump_primary</b>	Dumps information about the primary queue to standard output ( <b>stdout</b> ).
<b>dump_secondary</b>	Dumps information about the secondary queue to standard output ( <b>stdout</b> ).

**-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 Setting the Teamcenter Engineering Environment](#) in chapter 1, [Introduction](#).

**FILES**

As specified in [Log Files](#) in chapter 1, [Introduction](#).

**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
```

---

**create\_appearances**

---

Creates appearances representing an initial product structure in a particular context. Also creates an appearance root, if required.

**SYNTAX**

**create\_appearances** **-u**=*user-name* **-p**=*password* **-g**=*group-name*  
**-item\_id**=*item-id* **-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*]

**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.



If Teamcenter 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 Engineering database. If you do not supply these arguments, the utility attempts 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-name* 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.

**-item\_id**

Specifies the item to track.

**-config\_rule**

Specifies the revision rule to use.

**-view\_type**

Specifies the view to use.

**ENVIRONMENT**

As specified in *Manually Setting the Teamcenter Engineering Environment* in chapter 1, *Introduction*.

**FILES**

None.

**RESTRICTIONS**

None.

**EXAMPLES**

None.

---

**purge\_baselined\_item\_revisions**

---

**DESCRIPTION**

Purges baseline revisions when the automatic purge process fails.

**SYNTAX**

**purge\_baselined\_item\_revisions** **-u**=*user-name* **-p**=*password*  
**-g**=*group-name* **-status**=*release-status* [**-date**=*yyyy MM dd hh mm ss* | **now** | **today**]

**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.



If Teamcenter 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 Engineering database. If you do not supply these arguments, the utility attempts 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-name* 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.

**-item\_id**

Specifies the appearance root item.

**-status**

Specifies the release status of the baselined item revisions.

**-date**

Specifies the date of the baselined item revisions.

**ENVIRONMENT**

As specified in [Manually Setting the Teamcenter Engineering Environment](#) in chapter 1, [Introduction](#).

**FILES**

None.

**RESTRICTIONS**

None.

**EXAMPLES**

None.

## Product Structure Utilities

This section describes utilities used to manage the following tasks related to Teamcenter Engineering product structure:

- Activate the Teamcenter Engineering 7.0 effectivity functionality.
- Convert effectivity data created in Teamcenter Engineering versions prior to 7.0 into the new effectivity model used in 7.0 and higher.
- Report the differences between a NX cached part occurrence structure in the top-level part file and the corresponding Teamcenter Engineering product structure.
- Input default values for note types.
- Rename BOMViews and BOMView revisions using the current naming scheme.
- Traverse a product structure and report BOMline attributes and workspace attribute values in a file in delimited format.
- Create imprecise product structures based on an input file.
- Update BOM change records corresponding to all existing change objects.
- Upgrades revision rules so that they can safely use the modified revision rule implementation introduced in Teamcenter Engineering 7.0.
- Perform clearance analysis on a product structure and store the results data in the clearance database.
- Update items in a BOM structure with projects.

## **Product Structure Maintenance**

This section describes the utilities used to create, update, and maintain Teamcenter Engineering product structure.



---

## generate\_iman\_ps\_path

---

**DESCRIPTION**

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\_iman\_ps\_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.



If Teamcenter 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 Engineering database. If you do not supply these arguments, the utility attempts 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-name* 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.

**-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 Engineering environment is set.

#### **FILES**

As specified in [Log Files](#) in chapter 1, [Introduction](#).

#### **RESTRICTIONS**

None.

#### **EXAMPLES**

To generate a path for the item revisions specified in a text file:

```
$IMAN_ROOT/bin/generate_iman_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
```

## input\_notetype\_default

### DESCRIPTION

Inputs default values for note types.

### SYNTAX

**input\_notetype\_default** **-u**=user-name **-p**=password **-g**=group [**-n**=note-type-name **-d**=default-value] [**-f**=file-name **-c**=comment-line-identifier]

### 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.



If Teamcenter 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 Engineering database. If you do not supply these arguments, the utility attempts 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-name* 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.

#### **-f**

Modifies a list of note type default values simultaneously by writing the entries to a file. The note type and default value is written on a new line in the file in the following syntax:

```
Note Type Name: Note Type Default Value
```

#### **-c**

Identifies the beginning of a comment line in the file. The comment line identifier is a character entry used in conjunction with the **-f** argument. By default, the identifier is **#**, but you can define another character as the identifier. This argument is optional.

### ENVIRONMENT

As specified in [Manually Setting the Teamcenter Engineering Environment](#) in chapter 1, [Introduction](#).

### FILES

As specified in [Log Files](#) in chapter 1, [Introduction](#).

### RESTRICTIONS

This utility requires Teamcenter Engineering system administration privileges.

**EXAMPLES**

- To add a default value of **Active** to the **BomUsage** note, enter:

```
input_notetype_default -u=user-name -p=password -g=group  
-n-BomUsage -d=Active
```

- To add a list of default values from a file named **notetype\_def.txt**, enter:

```
input_notetype_default -u=user-name -p=password -g=group  
-f=notetype_def.txt
```

- To add a list of default values from a file named **notetype\_def.txt** with the comment line identifier as **@**, enter:

```
input_notetype_default -u=user-name -p=password -g=group-name  
-f=notetype_def.txt -c=@
```

---

**ps\_rename\_bvrs**

---

**DESCRIPTION**

Renames BOMViews and BOMView revisions using the current naming scheme. The new name can differ from the old name either because the naming scheme has changed or because the name of the view type has changed.

By default, the utility runs on all BOMViews and BOMView revisions in the database, but it can accept an item ID argument (which can include wildcards) defining a set of objects to rename.

**SYNTAX**

**ps\_rename\_bvrs** [**-item**=*item\_pattern*] [**-view**=*view-type*] [**-h**]

**ARGUMENTS****-item**

Specifies which BOMViews or BOMView revisions to rename by a pattern match on the parent item ID. The default is to rename the BOMViews and BOMView revisions of all items.

**-view**

Specifies which BOMViews or BOMView revisions to rename by BOMView type. The default is to rename BOMViews or BOMView revisions of all types.

**-v**

Verbose mode.

**-h**

Displays help for this utility.

**ENVIRONMENT**

As specified in [Manually Setting the Teamcenter Engineering Environment](#) in chapter 1, [Introduction](#), and the following preferences:

**IMAN\_ignore\_case\_on\_search**

**IMAN\_pattern\_match\_style**

For more information about these preferences, see *Configuration Guide*.

**FILES**

As specified in [Log Files](#) in chapter 1, [Introduction](#).

**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 BOMViews and BOMView revisions (BVRs) in the database, enter the following command:  
  
`$IMAN_ROOT/bin/ps_rename_bvrs`
- Consider a site where the **Manufacturing** BOMView type is renamed to **M Site 1**. To rename all BOMViews and BOMView revisions with parent item IDs beginning **pbx** to agree with the new name, enter the following command on a single line:

```
$IMAN_ROOT/bin/ps_rename_bvrs -view="M Site 1" -item="pbx*"
```

---

## ps\_traverse

---

### DESCRIPTION

Traverses a product structure and reports BOMline 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 **\$IMAN\_ROOT** directory.

### Sample Configuration File

Figure 3-1 shows the contents of the **ps\_traverse.cfg** sample configuration file, which is located in the **\$IMAN\_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
```

**Figure 3-1. Sample Configuration File**

**SYNTAX**

```
ps_traverse [-u=user-id]  
[-p=password]  
[-g=group-name]  
-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]  
[-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.



If Teamcenter 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 Engineering database. If you do not supply these arguments, the utility attempts 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-name* 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.

**-itemid=**

Specifies the ID of the item for which the associated BOMView revision (BVR) is traversed. (BOMView 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**.

**-cfg=**

Specifies the input configuration file. The default file is **te.cfg**.

#### CONFIGURATION FILE ENTRIES

The configuration file is a text file to which entries must be made under the following separate headings:

**bomreport=**

Valid values for this attribute are the display names of columns in PSE, 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**



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.

---

## ps\_upload

---

**DESCRIPTION**

Creates imprecise product structures based on an input file.

**SYNTAX**

**ps\_upload -u=user-id -p=password -g=group [-o] [-h] -i= | input-file**

**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.



If Teamcenter 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 Engineering database. If you do not supply these arguments, the utility attempts 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-name* 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.

**-o**

Turns off overwrite mode. Default setting is **on**.

**-i**

Specifies the input file name.

**-h**

Displays help for this utility.

**ENVIRONMENT**

As specified in [Manually Setting the Teamcenter Engineering Environment](#) in chapter 1, [Introduction](#).

**FILES**

As specified in [Log Files](#) in chapter 1, [Introduction](#).

**RESTRICTIONS**

- The default mode, **overwrite**, replaces any existing structures with the information contained in the input file.
- Input files must use the following format:

**Items**

The input file consists mainly of item lines, each defining an item that the **ps\_upload** utility creates. The file can also contain comments and directive lines, but these 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.

Do not show more than one occurrence of the same expanded assembly or you duplicate its contents.

**A** is assigned as the initial revision ID.

**Columns**

The pound sign (#) in the first column indicates a comment that is ignored, with the exception of lines beginning with **#DELIMITER** or **#COL**. Completely blank lines are also ignored. The default order for column headings is as follows:

<b>item</b>	Item ID.
<b>name</b>	Item name.
<b>level</b>	Determines the structural hierarchy.
<b>seq</b>	Specifies the sequence number of the item in the BOMView.
<b>occs</b>	Specifies the number of occurrences of the item.
<b>qty</b>	Specifies the quantity of an item in the structure.
<b>uom</b>	Specifies the symbol representing the unit of measure.
<b>alt</b>	Specifies alternate parts.
<b>type</b>	Specifies the item type. Note that the type must already exist in the database.

**rev**

Allows you to specify a revision letter other than the default, which is **A**.



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.

**option**

Allows an option and set of allowed values to be defined and attached to the item being created by that line. The format is as follows:

*Option-name;Value;Value...*



The delimiter must be that defined for Alternates (**#ALT\_DELIMITER**). In the previous example, the delimiter was a semicolon (;).

**loadif**

Allows a simple variant condition to be specified using an option that has been defined in the previously created items. The format is as follows: The variant condition is limited to one option/value expression.

*Owning-ItemID;Option-name== or != Value*



The delimiter must be that defined for Alternates (**#ALT\_DELIMITER**). In the previous example, the delimiter was a semicolon (;).

**Alternates**

An entry in the alternates column should consist of a delimiter-separated list of item IDs, where each alternate has been individually defined as a **level 0** structure. The default column delimiter is a semicolon (;).

The **#ALT\_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**, the delimiter is set to a blank space (' '). The alternate delimiter cannot be the same as the column delimiter.

**EXAMPLES**

The following input file (figure 3-2) 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 alternates 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 Alt
#   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   Alt 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 Alternate Delimiter to /
#ALT_DELIMITER /
#       2,   40, b005,      -,   bolt,                      -, b101 / b100, 2
#       1,   50, b011,      -,   Saddle,
# Change Delimiter to allow commas in the name
#DELIMITER $
#       1$   60$ 1001$      ml$ Oil, lubricating$      -$   -$   100
#       1$   70$ 1002$      m$  Paint, red$            -$   -$   2.4
-----
```

**Figure 3-2. Product Structure Input File**

## update\_bomchanges

### DESCRIPTION

Updates BOM change records corresponding to all existing change objects. Customers moving to Teamcenter Engineering 8.1 from 7.0.3 or 8.0 must run this utility to create **Move** and **Reshape** changes that were tracked as common parts until Teamcenter Engineering 8.1. 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

**update\_bomchanges** [**-h**] **-u**=*user-name* **-p**=*password* **-g**=*group-name*  
**-c**=**{0|1|2}** **-e**=*change-id* **-l**=*log-file*

### 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.



If Teamcenter 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 Engineering database. If you do not supply these arguments, the utility attempts 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-name* 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.

#### **-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.

**-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 Setting the Teamcenter Engineering Environment](#) in chapter 1, [Introduction](#).

**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:

```
$IMAN_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:

```
$IMAN_BIN/update_bomchanges -u=infodba -p=infodba -g=dba -c=2
```



## update\_project\_bom

### DESCRIPTION

Allows you to update all items in a BOM structure within specified projects.

### SYNTAX

```
update_project_bom -u=user-id -p=password -g=group [-f={add | remove}]  
[-type={item | rev}] -item=item-id [-rev_id=revision-id] [-rev_rule=revision-rule]  
[-unit_no=unit-number] [-date=date] [-end_item=end-item-id]  
[-var_rule=variant-rule] -projects=project-lists [-level=level-number] [-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.



If Teamcenter 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 Engineering database. If you do not supply these arguments, the utility attempts 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-name* 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.

#### **-f**

Specifies one of the following types of operation for this utility:

**=add** Adds item objects to projects.

**=remove** Removes item objects from projects.



If this argument is omitted, the default action is to add items to 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.



If this argument is omitted, the default type is **item**.

**-item**

Specifies the ID of the root item of the BOM to update.

**-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.

**-var\_rule**

Specifies the variant rule to be applied to the BOM structure. If omitted, no variant rule is applied.

**-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. More than one project can be specified. If there is more than one project in the list, each project name is separated by a comma (,).

**-level**

Specifies to what depth the BOM tree is traversed. The default value is **all**, which indicates the entire BOM structure is traversed.

**-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 -level=3
```

---

**upgrade\_rev\_rules**

---

**DESCRIPTION**

Upgrades revision rules so that they can safely use the modified revision rule implementation introduced in Teamcenter Engineering 7.0. 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** [-h] [-u -p [-g]] [-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.



If Teamcenter 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 Engineering database. If you do not supply these arguments, the utility attempts 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-name* 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.

**-h**

Displays help for this utility.

**-v**

Verbose mode. Shows additional messages.

**ENVIRONMENT**

As specified in [Manually Setting the Teamcenter Engineering Environment](#) in chapter 1, [Introduction](#).

**FILES**

As specified in [Log Files](#) in chapter 1, [Introduction](#).

**RESTRICTIONS**

User must be logged on as a member of a system administration group or use the **-u -p -g** arguments to run the utility as a member of a system administration group.

**EXAMPLES**

None.

## **Effectivity Mode**

The utility used to update effectivity mode are described in this section.

---

**effupgrade**

---

**DESCRIPTION**

Converts effectivity data created in Teamcenter Engineering versions prior to 7.0 into the new effectivity model used in version 7.0 and later. 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/end date/unit values on the release status.

**SYNTAX**

**effupgrade [-h] [-u -p [-g]] [-s] [-d] [-v [-e]]**

**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.



If Teamcenter 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 Engineering database. If you do not supply these arguments, the utility attempts 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-name* 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.

**-h**

Displays help for this utility.

**-s**

Specifies a single run. Upgrade runs once and ignores locked statuses.

**-i**

Specifies rerun interval as minutes between reruns when invoked without the **-s** argument. The default values 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.

**ENVIRONMENT**

As specified in *Manually Setting the Teamcenter Engineering Environment* in chapter 1, *Introduction*.

**FILES**

As specified in *Log Files* in chapter 1, *Introduction*.

**RESTRICTIONS**

The user must be logged in as a member of a system administration group or use the **-u -p -g** switches to run the utility as a member of a system administration group.

**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 in), and then run repetitively until complete.  

```
effupgrade -u=infodba -p=password -g=dba
```

## **Product Structure Clearance Analysis**

This section describes the utilities used to perform clearance analysis on product structures.



---

**batchmode\_clearance\_analysis.pl**


---

**DESCRIPTION**

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**

**IMAN\_ROOT/clearanceDB/scripts/**

**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.



If Teamcenter 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 Engineering database. If you do not supply these arguments, the utility attempts 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-name* 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.

**-bookmark**

Generates a bookmark file of the product.

**-h**

Displays help for this utility.

**ENVIRONMENT**

As specified in [Manually Setting the Teamcenter Engineering Environment](#) in chapter 1, [Introduction](#), and in the setup document provided in the **IMAN\_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
```

## **Product Structure Comparison**

This section describes the utilities used to compare NX and Teamcenter Engineering product structures.

---

## compare\_ug\_and\_iman\_ps

---

**DESCRIPTION**

Runs three utilities (**find\_released\_item\_rev**, **generate\_iman\_ps\_path**, and **ugmanager\_compare\_ps**) in sequence and reports the differences between a NX cached part occurrence structure in the top-level part file and the corresponding Teamcenter Engineering product structure.

Use this utility to simplify the procedure for determining such differences. This utility can be modified to use the **find\_recently\_saved\_item\_rev** utility rather than the **find\_released\_item\_rev** utility.

**SYNTAX**

**compare\_ug\_and\_iman\_ps**

**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.



If Teamcenter 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 Engineering database. If you do not supply these arguments, the utility attempts 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-name* 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.

**-change\_obj\_type**

Defines the type of object to be searched for a change of release status since January 1, 1970. If this argument is not specified, objects of all types are searched.

This argument is optional. If defined, this setting is used for the **find\_released\_item\_rev** utility.

**-top\_assem\_type**

Determines the type of top-level assembly for which the path is generated. If this argument is not specified, the path is generated for all item types.

This argument is optional. If defined, this setting is used for the **generate\_iman\_ps\_path** utility.

**-rev\_rule\_file**

Defines the file that contains the list of revision rules.

This argument is required unless it is used with the **-rev\_rule** option, in which case it is optional. This argument is used with the **generate\_iman\_ps\_path** utility.

**-configure\_top\_level\_revs**

Defines the top-level revision configuration flag. If set to **Yes**, all top-level revisions are configured. If set to **No**, or not defined, top-level revisions are not configured.

This argument is used in conjunction with the **generate\_iman\_ps\_path** utility.

**-rev\_rule**

Defines the revision rule for the top-level item.

This argument is required unless it is used with the **-rev\_rule\_file** option, in which case it is optional. This argument is used with the **generate\_iman\_ps\_path** utility.

**-top\_item\_id**

Defines the item ID of the top-level assembly for which the path should be compared.

This argument is optional. If defined, it is used with the **ugmanager\_compare\_ps** utility.

**-top\_item\_rev\_id**

Defines the item revision ID of the top-level assembly for which the path should be compared.

This argument is optional. If defined, it is used with the **ugmanager\_compare\_ps** utility.

**-log\_file**

Defines the log file to which the differences are logged.

This argument is optional; if not defined, the output is written to the standard output.

**-h**

Displays help for this utility.

**ENVIRONMENT**

This utility must be run from a shell where the Teamcenter Engineering and NX environments are set.

**FILES**

None.

**RESTRICTIONS**

This utility can be used only with NX 1.0.5.2 and later, NX 1.0.2 and later, and NX 2.0.

**EXAMPLES**

- To list the differences of all the top-level assemblies present in the database:

```
$IMAN_ROOT/bin/compare_ug_and_iman_ps.pl -log_file=path_diff.log
```

- To list the differences all the way down to the leaf node of a given top-level assembly:

```
$IMAN_ROOT/bin/compare_ug_and_iman_ps.pl -change_obj_type="Item Revision"
-top_item_id=ABC0001 -log_file=path_diff.log
```

---

## ugmanager\_compare\_ps

---

**DESCRIPTION**

Compares the NX cached part occurrence structure in a top-level part file with the corresponding Teamcenter Engineering product structure and reports any differences. Use this utility to ensure that the NX cache and Teamcenter Engineering product structure are synchronized.

**SYNTAX**

**\$UGII\_BASE\_DIR/ugmanager/ugmanager\_compare\_ps**

**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.



If Teamcenter 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 Engineering database. If you do not supply these arguments, the utility attempts 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-name* 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.

**-path\_list**

Defines the text file containing all the paths in the Teamcenter Engineering product structure. Typically, this is the output file from the **generate\_iman\_ps\_path** utility.

This argument is required.

**-top\_item\_id**

Determines the item ID of the top-level assembly. This argument is optional and is used in combination with the **-top\_item\_rev\_id** and **-rev\_rule** arguments.

If the **-top\_item\_id**, **-top\_item\_rev\_id** and **-rev\_rule** arguments are all specified, the three arguments are validated to determine whether they form a valid combination; a top-level item revision can be configured with only one revision rule. If the combination is not valid, an error is displayed.

If only the **-top\_item\_id** and **-top\_item\_rev\_id** arguments are specified, the defined item revision is processed. If only the **-top\_item\_id** and **-rev\_rule** arguments are specified, the corresponding item revision is processed.

If only the **-top\_item\_rev\_id** argument is specified (the **-top\_item\_id** is not), an error is raised.

If only the **-top\_item\_id** argument is specified, all revisions of the defined item are processed.

If only the **-rev\_rule** argument is specified, all top-level item revisions corresponding to the defined revision rule is processed.

If none of the three arguments: **-top\_item\_id**, **-top\_item\_rev\_id**, or **-rev\_rule** are specified, all top-level **Item Revisions** in the **path\_file** are processed.

#### **-top\_item\_rev\_id**

Defines the item revision ID of the top-level assembly. This argument is optional; it is only used with the **-top\_item\_id** argument.

#### **-rev\_rule**

Defines the revision rule for the top-level item. This argument is optional; it is used in conjunction with the **-top\_item\_id** argument.

#### **-log\_file**

Defines the log file to which the differences are logged. 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 Engineering and NX environments are set.

#### **FILES**

As specified in [Log Files](#) in chapter 1, *Introduction*.

#### **RESTRICTIONS**

This utility can only be used with NX 1.0.5.2 and later , NX 1.0.2 and later, and NX 2.0. The **path\_list** file should conform to the format generated by the **generate\_iman\_ps\_path** utility.

#### **EXAMPLES**

- To list the differences of all the top-level assemblies present in the **path\_list** file to a log file:

```
$UGII_BASE_DIR/ugmanager/ugmanager_compare_ps  
-path_list=path_list.txt -log_file=path_diff.log
```

- To list the differences for top-level item **ABC0001**:

```
$UGII_BASE_DIR/ugmanager/ugmanager_compare_ps  
-path_list=path_list.txt -top_item_id=ABC0001 -log_file=path_diff.log
```

---

## Chapter

# 4 *Workflow Utilities*

clear_process_stage_list	4-2
delete_namedrevisions	4-3
find_processes	4-4
global_transfer	4-6
tc_workflow_postprocess	4-8
purge_processes	4-12
install_handlers	4-14
release_man	4-16
upgrade_motif_workflow_templates	4-18
upgrade_workflow_objects	4-20
verify_tasks	4-22





---

## Chapter

# 4 *Workflow Utilities*

---

This chapter describes the command line utilities related to Teamcenter Engineering workflow.

---

---

**clear\_process\_stage\_list**

---

**DESCRIPTION**

Clears the **process\_stage\_list** field of the workspace object.



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 -g=group-name*

**ARGUMENTS****-u**

Specifies the user ID.

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

**-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.

**-g**

Specifies the group associated with the user. The group value must be **dba** to run this utility. See restriction #1.

**-folder**

Specifies the folder name where the target workspace objects should be placed whose process stage lists are to be cleared. See restriction #2.

**-h**

Displays help for this utility.

**ENVIRONMENT**

As specified in [Manually Setting the Teamcenter Engineering Environment](#) in chapter 1, [Introduction](#).

**FILES**

As specified in [Log Files](#) in chapter 1, [Introduction](#).

**RESTRICTIONS**

- You must be logged in as a member of the **dba** group.
- The folder must be a single folder directly inside the executing user's **Home** folder.

**EXAMPLES**

To clear the **process\_stage\_list** fields of all the objects in the **my\_folder** object, enter the following command:

```
$IMAN_ROOT/bin/clear_process_stage_list -u=user-id -p=password  
-g=dba -folder=my_folder
```

## delete\_namedrevisions

### DESCRIPTION

Queries the database to retrieve tasks that have **null\_tags** for a **parent\_process** attribute. Once it finds the task, it looks for the **releaseLevels** that reference the task. The **releaseLevels** are then deleted and subsequently the tasks are deleted.

### SYNTAX

**delete\_namedrevisions [-h] -u=user-name -p=password -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.



If Teamcenter 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 Engineering database. If you do not supply these arguments, the utility attempts 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-name* 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.

#### **-h**

Displays help for this utility.

### ENVIRONMENT

As specified in [Manually Setting the Teamcenter Engineering Environment](#) in chapter 1, [Introduction](#).

### FILES

As specified in [Log Files](#) in chapter 1, [Introduction](#).

### RESTRICTIONS

None.

---

**find\_processes**

---

**DESCRIPTION**

Queries the database, returning the names of all process objects still in-process. It also references the handlers in question so they can be tracked and completed, if necessary. This utility is helpful in determining which processes must be completed before deprecated handlers are no longer supported.

Use the **find\_processes** batch utility whenever a handler is made obsolete.

In some cases, this utility may take a while to complete because it has to load all the processes that are in-process.

**SYNTAX**

```
find_processes [-u=user-id -p=password -g=group [-f=input-file |  
-handler=handler1[,handler2...]> ] [-rpt=report-file-name] [-send_mail] [-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.



If Teamcenter 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 Engineering database. If you do not supply these arguments, the utility attempts 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-name* 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.

**-f=*input-file***

Specifies a text file containing handler names. Each handler name must be specified in a new line. This argument should not be used with the **-handler** argument.

**-handler=*handler1*[,*handler2*...]> ]**

Specifies names of handlers. This option must not be used with the **-f** argument.

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

Specifies the output file. If not specified, output is written to standard output.

**[-send\_mail]**

Specifies that references to the process objects are sent as mail to the user who executes this utility. A report file, in the form of a text dataset, is attached to the mail.

The mail option allows the user to see the status of the processes. This argument is optional. If the number of process objects in the output exceeds 512, the output is split into units of 512 objects and each is sent in a separate E-mail.

**-h**

Displays help for this utility.

**ENVIRONMENT**

This utility must be run in the Teamcenter Engineering shell environment.

**FILES**

As specified in [Log Files](#) in chapter 1, [Introduction](#).

**RESTRICTIONS**

None.

**EXAMPLES**

None.

---

**global\_transfer**

---

**DESCRIPTION**

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* -g=*group*] -f= -t=

**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.



If Teamcenter 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 Engineering database. If you do not supply these arguments, the utility attempts 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-name* 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.

**-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 Setting the Teamcenter Engineering Environment](#) in chapter 1, [Introduction](#).

**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 of user Mike to user Kevin:

```
_transfer -f=mike -t=kevin
```

---

**tc\_workflow\_postprocess**

---

**DESCRIPTION**

Executes a specific action on a specific task in the workflow process from which the related background process was initiated.

For example, the utility could evoke the **complete** action on the review task in the workflow process from which a background tessellation process was initiated. The 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 -status_xfer_type=transfer-type itemid=item-id
-rev-id=rev-id -dsname=dataset-name -dataset_tag=dataset-tag
-task_tag=tag [-u=user-id -p=password [-g=group]] [-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 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.

**-g**

Specifies the group associated with the user.

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

If the utility is invoked via the **invoke-system-action** or **invoke-system-rule** handlers, the utility inherits the session authentication and the **-u/-p** switches are not needed, irrespective of the autologin setting.



**-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.

**-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 **invoke-system-rule** handler or the **invoke-system-action** handler.

**-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**

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**

Comment for the signoff specified with the **-signoff=decision** argument.

**-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.

**-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.

**-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.

**-h**

Displays help for this utility.

**ENVIRONMENT**

As specified in *Manually Setting the Teamcenter Engineering Environment* in chapter 1, *Introduction*.

**FILES**

As specified in *Log Files* in chapter 1, *Introduction*.

**RESTRICTIONS**

None.

**EXAMPLES**

- The example in figure 4-1 checks the group/role defined by the **-member\_group** and **-member\_role** arguments (**Body/Designer**) against the group/role the user is currently logged in 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.

```
$IMAN_ROOT/bin/tc_workflow_postprocess -member_group=Body
-member_role=Designer -task_tag=AZwszeaegnHvqDAAAAAAAAAAAAAA -action=Complete
-trigger_comment="Tessellation completed successfully"
```

**Figure 4-1. tc\_workflow\_postprocess Utility**

- The example in figure 4-2 checks the group/role defined by the **-member\_group** and **-member\_role** arguments (**Body/Designer**) against the group/role the user is currently logged in 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.

```
$IMAN_ROOT/bin/tc_workflow_postprocess -member_group=Body
-member_role=Designer -task_tag=AZwszeaegnHvqDAAAAAAAAAAAAAA -signoff=NoDecision
-signoff_comment="Tessellation failed - disk full"
```

**Figure 4-2. tc\_workflow\_postprocess Utility**

- 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.

```
$IMAN_ROOT/bin/tc_workflow_postprocess
-dataset_tag=GAXwszeagnHvqDAAAAAAAAAAAAAA
```

- The following example transfers the release status from a **UGMASTER** dataset to its corresponding **CAEMesh** dataset:

```
$IMAN_ROOT/bin/tc_workflow_postprocess
-dataset_tag=QZPBK4_6x4$kbDAAAAAAAAAAAAAA
-status_xfer_type=cae_mesh -itemid=000266 -revid=A -dsname=000266/A
```

---

**purge\_processes**

---

**DESCRIPTION**

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. The **-force** argument can be used to delete the processes and sever the references between objects and the processes.

**SYNTAX**

**purge\_processes -u=infodba -p=password -g=dba -d[ate]=MM/DD/YYYY  
-f[orce] -r[eport] -h[elp]**

**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.



If Teamcenter 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 Engineering database. If you do not supply these arguments, the utility attempts 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-name* 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.

**-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.

**-force**

Deletes specified processes from the system and severs references between objects and the processes being deleted. Unless this argument is specified, only processes that have no referenced objects are purged.

**-report**

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 Setting the Teamcenter Engineering Environment* in chapter 1, *Introduction*.

**FILES**

As specified in *Log Files* in chapter 1, *Introduction*.

**RESTRICTIONS**

This utility must be run by an administrator.

**EXAMPLES**

Enter the following command on a single line to remove all processes that have been modified on or before 15th April 1998:

```
purge_processes -u=user-id -p=password -g=dba -d=04-15-1998
```

---

## install\_handlers

---

### DESCRIPTION

Defines action handlers. It can also configure new action handlers and modify the definition of existing handlers.

### SYNTAX

```
install_handlers [-h] -u=user-name -p=password -g=group-name  
-f=function {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
```

### 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.



If Teamcenter 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 Engineering database. If you do not supply these arguments, the utility attempts 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-name* 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.

#### -f

Specifies the mode in which the utility must execute. The mode must be one of the following:

- **install**
- **create**
- **modify**
- **delete**
- **listall**

**-functype**

Specifies whether the handler is a library function or a standalone executable. The value for this argument must be one of the following:

- 1 Library function
- 2 Standalone 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.

**-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.

**ENVIRONMENT**

As specified in [Manually Setting the Teamcenter Engineering Environment](#) in chapter 1, [Introduction](#).

**FILES**

As specified in [Log Files](#) in chapter 1, [Introduction](#).

**RESTRICTIONS**

None.

**EXAMPLES**

- The following command installs the default action handlers:  

```
install_handlers -f=install
```
- The following command creates an action handler with the specified attribute values to set the execution time for the handler to 6.00 p.m.:  

```
install_handlers -f=create -id=MyActionHandler -funcname=Myfunc  
-functype=1 -execmode=1 exectime=1800
```
- The following command sets the retry count value to **5** for the **MyActionHandler** action handler:  

```
install_handlers -f=modify -id=MyActionHandler -retryCount=5
```
- The following command deletes the **MyActionHandler** action handler:  

```
install_handlers -f=delete -id=MyActionHandler
```
- The following command lists all the action handlers defined in the database:  

```
install_handlers -f=listall
```

---

**release\_man**

---

**DESCRIPTION**

Releases objects in batch mode without creating jobs and audit files.

**SYNTAX**

**release\_man** **-u**=*user-id* **-p**=*password* **-g**=*dba* [**-spec**] [**-unrelease**]  
**-retain\_release\_date** **-status**=*status-type* **-folder**=*folder-name* **-acl**=*{user ::::: | group ::: | group::role: | role: | pseudo-role:}* **read** | **write** | **copy** | **delete** | **change**

**ARGUMENTS****-u**

Specifies the user ID.

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

**-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.

**-g**

Specifies the group. Must be a **dba** group.

See restriction #1.

**-spec**

Indicates that specifications and BOMView revisions of an item revision in the release folder are released along with the item revision.

**-unrelease**

Removes the specified status type. See restriction #2.

**-retain\_release\_date**

Specifies that if the object to be released is already released, the original release date is retained.

**-status**

Specifies the status type to be applied to all objects.

See restriction #2.

**-folder**

Specifies the name of the release folder. See restriction #3.

**-acl**

Specifies the object protections to apply. Accessors are separated by commas. To apply the same object protections to more than one accessor, simply supply the appropriate colons. If the **-acl** argument is not supplied, the default protection is read-only for the executing user. See restriction #4.



**ENVIRONMENT**

As specified in *Manually Setting the Teamcenter Engineering Environment* in chapter 1, *Introduction*.

**FILES**

As specified in *Log Files* in chapter 1, *Introduction*.

**RESTRICTIONS**

- The user must be a member of the **dba** group.
- The status type must be a valid status type defined for your site.
- The release folder must be a single folder directly inside the executing user's workspace **Home** folder.
- The **release\_man** utility does not release invalid objects or objects locked by other processes.

**EXAMPLES**

- To apply the **Released** status type to all objects in the **my\_folder** folder (including ItemRevision specifications and BOMView revisions), enter the following command:
- To remove the **Released** status type from all objects in the **my\_folder** folder (including ItemRevision specifications and BOMView revisions), enter the following command:

```
$IMAN_ROOT/bin/release_man -u=user-id -p=password -g=dba -spec  
-status=Released -folder=my_folder
```

```
IMAN_ROOT/bin/release_man -u=user-id -p=password -g=dba -spec  
-unrelease -status=Released -folder=my_folder
```

---

**upgrade\_motif\_workflow\_templates**

---

**DESCRIPTION**

This utility upgrades templates from Teamcenter Engineering versions prior to 2005 SR1 to Teamcenter Engineering 2005 SR1. Once the upgrade is complete, this utility is obsolete.

Modifies process templates created using the Motif interface to be usable in the rich client interface. This utility upgrades the latest available process template, as well as all versions of workflow procedure definition files referenced by all in-process workflow processes. This utility can be executed multiple times to upgrade to the most recently Motif created templates.



Prior to Teamcenter Engineering 9.0, this utility was called **upgrade\_process\_template**.

**SYNTAX**

**upgrade\_motif\_workflow\_templates -u=infodba -p=password -g=dba**

**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.



If Teamcenter 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 Engineering database. If you do not supply these arguments, the utility attempts 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-name* 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.

**ENVIRONMENT**

As specified in [Manually Setting the Teamcenter Engineering Environment](#) in chapter 1, [Introduction](#).

**FILES**

As specified in [Log Files](#) in chapter 1, [Introduction](#).

**RESTRICTIONS**

Requires Teamcenter Engineering administration privileges.

**EXAMPLES**

To upgrade all process template definition files that have been created using the Motif interface, enter the following command on a single line:

```
upgrade_motif_workflow_templates -u=infodba -p=infodba -g=dba
```

---

**upgrade\_workflow\_objects**

---

**DESCRIPTION**

Allows users to convert all file-based workflow templates to database workflow template objects, convert any transient objects associated with the workflow templates to persistent objects, and update the referencing workflow processes.

The following handlers are affected by the upgrade:

- **EPM-set-job-type**

Removes this handler from templates and source code. Also removes the handler's registration and documentation.

- **modify-status**

Removes this handler from templates and source code. Also removes the handler's registration and documentation.

- **check-completion**

Replaces usage of this handler with the new attributes **dependency\_task\_templates** and **dependency\_task\_actions** on the new **EPMTaskTemplate** objects. Removes this handler from templates and source code. Also removes the handler's registration and documentation.

- **EPM-add-task-to-targets**

Replaces usage of this handler with the new attribute **show\_in\_process\_stage** on new **EPMTaskTemplate** objects. Removes this handler from templates and source code. Also removes the handler's registration and documentation.

**SYNTAX**

**upgrade\_workflow\_objects -u=infodba -p=password -g=dba [-report] -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.



If Teamcenter 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 Engineering database. If you do not supply these arguments, the utility attempts 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-name* 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.

**[-report]**

Generates a report without purging. The report names the processes to be purged and displays information about the number of processes to be purged.

**-h**

Displays help for this utility.

**ENVIRONMENT**

No special environment variables or settings are required.

**FILES**

A log file is created in the general log file directory. The format for the log file name is as follows:

```
purge_process_number.log
```

This file records the time the utility begins and ends, how many processes have been purged, and any error messages that are generated.

**RESTRICTIONS**

None.

**EXAMPLES**

Enter the following command on a single line to upgrade workflow objects:

```
upgrade_workflow_objects -u=user-id -p=password -g=dba
```

---

**verify\_tasks**

---

**DESCRIPTION**

Finds all corrupted CM 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* **-g**=*group* [**-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.



If Teamcenter 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 Engineering database. If you do not supply these arguments, the utility attempts 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-name* 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.

**-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 Setting the Teamcenter Engineering Environment](#) in chapter 1, [Introduction](#).

**FILES**

As specified in [Log Files](#) in chapter 1, [Introduction](#).

**RESTRICTIONS**

Teamcenter Engineering administration privileges are required to run this utility.

**EXAMPLES**

None.





---

## Chapter

# 5 *Data Sharing Utilities*

batch_export_translate_import	5-2
database_verify	5-4
dsa_util	5-6
EMBulkloader	5-9
export_recovery	5-11
data_share	5-15
idsminetd	5-24
data_sync	5-25
import_file	5-35
item_export	5-38
item_import	5-44
item_relink	5-47
item_rename	5-51
plmxml_export	5-54
plmxml_import	5-57
step_export	5-60
step_import	5-62
upgrade_nx_cam_templates	5-64



---

## Chapter

# 5 *Data Sharing Utilities*

---

This chapter describes the utilities used to manage data sharing activities, such as importing and exporting data and mass publishing objects to remote sites.

---

---

**batch\_export\_translate\_import**

---

**DESCRIPTION**

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 Engineering.
- **Import mode (-i)**  
Imports datasets to Teamcenter Engineering.
- **Export, translate, import mode (-eti)**  
Exports, translates, and imports the dataset back in to Teamcenter Engineering.

**SYNTAX**

```
batch_export_translate_import -u=user-name -p=password  
-g=group-name -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.

**-p**

Specifies the user's password.



You can set the **IMAN\_USER\_PASSWORD** environment variable rather than specifying the **-p** option.

**-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 Setting the Teamcenter Engineering Environment](#) in chapter 1, [Introduction](#). 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](#) in chapter 1, [Introduction](#).

**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
```

---

**database\_verify**

---

**DESCRIPTION**

Compares database schema, Teamcenter Engineering types, tools, release statuses, and units of measure between two specified Multi-Site Collaboration sites and generates a report of any database discrepancies.

**SYNTAX**

**database\_verify** **-u**=*user-id* **-p**=*password* **-g**=*group* **-from**=*site-name1* **-to**=*site-name2* [**-schema**] [**-type**] [**-tool**] [**-status**] [**-uom**] [**-all**] [**-output**=*file-name*] [**-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.



If Teamcenter 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 Engineering database. If you do not supply these arguments, the utility attempts 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-name* 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.

**-from**

Specifies a Teamcenter Engineering database site to be verified.

**-to**

Specifies a Teamcenter Engineering 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.

**-v**

Runs utility in verbose mode. Display maximum amount of information. Typically, nonverbose utility sessions only display error messages.

**-h**

Displays help for this utility.

**ENVIRONMENT**

As specified in [Manually Setting the Teamcenter Engineering Environment](#) in chapter 1, [Introduction](#).

**FILES**

As specified in [Log Files](#) in chapter 1, [Introduction](#).

**RESTRICTIONS**

The **-from** and **-to** arguments must be specified.

**EXAMPLES**

None.

---

**dsa\_util**

---

**DESCRIPTION**

Distributes system administration data, such as users, groups, types, and lists of values (LOVs), 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.



This utility should be used only for the initial migration of system objects. It is not recommended for use in maintaining system objects.

If the data you are distributing includes note types with attached LOVs, the utility must be run separately to distribute the LOVs. Otherwise, the attached LOVs are not distributed to the new site.

Propagates Teamcenter Engineering 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-name -p=password -g=group -f={distribute(dist) |
export(exp) | import(imp) | list_classes(lc) | list_sites(ls) | check_sites(cs)}
[-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} [-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.



If Teamcenter 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 Engineering database. If you do not supply these arguments, the utility attempts 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-name* 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.

**-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** option. The output file can be edited and used with the **distribute** function in conjunction with the **-filename** option. Equivalent to exporting system objects.
- import (imp)** Imports system object information from the text file identified by the **-filename** option 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.

**-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.



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, UGS 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**

Displays help information for a specific topic.

**-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 -f**

Displays detailed help information for the given function.

**-h**

Displays help information on basic usage.

**RETURN  
VALUES**

**Return value**    0  
**upon success**

**Return value**    >1  
**upon failure**

## EMBulkloader

### DESCRIPTION

Imports multiple Solid Edge files into a Teamcenter Engineering database without starting the Solid Edge application. The utility consists of the **EMBulkloader.exe** file that works in conjunction with the existing Solid Edge Manager installation.

Run this executable from a Teamcenter Engineering MS-DOS command window and specify the path to the Solid Edge files that you want to import into the Teamcenter Engineering database.

### SYNTAX

**EMBulkloader.exe** **-path**=*path-name* [**-u**=*user-name*] [**-p**=*password*]  
 [**-g**=*group-name*] [**-auto**] [**-nofolder**] [**-delete**] [**-i**=*item-type*]  
 [**-uom**=*unit-of-measure*] [**-rev**=*revision*] [**-precise**] [**-dryrun**]

### 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.



If Teamcenter 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 Engineering database. If you do not supply these arguments, the utility attempts 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-name* 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.

#### **-path**

Specifies the path to the files to be imported. This argument is required.

#### **-delete**

Specifies that the local version of each file is deleted to the recycle bin after it has been imported into Teamcenter Engineering. If not specified, the original version of the file remains in the directory.

**-nofolder**

Specifies that files are not to be imported into a specific folder.

**-auto**

Specifies that the **USER\_exit** routine is used to generate missing item IDs. If not specified, the file name is used as the project ID.

**-i**

Specifies the item type used for the import operation. If not defined, the **EM\_item\_type** preference determines the item type. The default is **item**.

**-uom**

Specifies the unit of measure used for the import operation. If not defined, no unit of measure is assigned to the item.

**-rev**

Specifies the revision used for the import operation when a file does not have the document revision property specified. If not defined, the **USER\_exit** routine is used.

**-precise**

Specifies the precise BOMViews/BOMView revisions to be created. If not defined, the **IMAN\_BOM\_Precision\_Preference** preference determines how the BOMViews are constructed.

**-dryrun**

Starts a practice run to identify any possible errors that Solid Edge Manager might find without actually importing the files. Note that errors may still occur when the actual import takes place due to problems with the import operation.

**FILES**

The **EM\_Iportunique-id.log** log file is generated in the master connection directory and contains detailed information about each of the file transfer transactions. The unique id is a generated string that will make the file unique.

**EXAMPLES**

- The following example imports all of the files in the **Work** directory:

```
>EMBulkloader.exe -path=d:\Solid Edge\Work\*.*
```

- The following example imports all assemblies contained in the **Work** directory.

```
>EMBulkloader.exe -path=d:\Solid Edge\Work\*.asm
```

## export\_recovery

### DESCRIPTION

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 **IMAN\_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 **IMAN\_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.

### SYNTAX

```
export_recovery [-u=user -p=password -g=group] -mode={ full |
min | auto | auto_noexport | find } { [-item_id=item-id-to-restore]
| [-folder=folder-name] | [-filename=file-name] } [-dir=directory]
[-report=report-file] [-remote_site=last-transfer-site] [-include_bom]
[-new_owning_site=desired-owning-site-for-auto_noexport mode]
[-exclude=relation-type1 -exclude=relation-type2 ...] [-include=relation-type3
-include=relation-type4 ...] [-ignore_am_rules] [-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.



If Teamcenter 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 Engineering database. If you do not supply these arguments, the utility attempts 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-name* 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.

**-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. Only valid if the metafile was generated with transfer of ownership.

**auto**

Restores ownership on the specified item without reimporting. Only valid if the **-itemid** and **-remote\_site** options are specified.

**find**

Searches for items with inconsistent site ownership and generates a report.

**auto\_noexport**

Changes the owning site of specified objects to the new site. Only valid with the **-new\_owning\_site** option. Use only as a last resort if the **-auto** option fails to restore full site ownership.

**-dir**

Defines the path of the directory containing the exported metafile and the data files. Required only with the **-mode=full** and **-mode=min** options.

**-item\_id**

Specifies the ID of the item to process. Wildcards are allowed.

**-folder**

Defines the name of the Teamcenter Engineering folder containing the list of items to process.

**-filename**

Defines the full path of the file that contains the list of items to process.

**-remote\_site**

Defines the last site for which a transfer of ownership was attempted. This argument is only valid with the **-auto** option.

**-h**

Displays help for this utility.

**-report**

Specifies the full path of the report file. Valid only with **-find** mode.

**-new\_owning\_site**

Designates the desired owning site used with the **-auto\_noexport** mode. This may be the local site or a remote site.

**-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 option to force the inclusion of a relation type that is not specified by your **IMAN\_relation\_export** preference.

**-ignore\_am\_rules**

Ignores AM rules for recovery purposes.

**-bp**

Displays best practice recommendations.

**-h**

Displays help for this utility.

**ENVIRONMENT**

As specified in [Manually Setting the Teamcenter Engineering Environment](#) in chapter 1, [Introduction](#).

**FILES**

As specified in [Log Files](#) in chapter 1, [Introduction](#).

**RESTRICTIONS**

1. At least one primary mode must be specified.
2. Not more than one primary mode can be specified.
3. 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 restore site ownership after the **-mode=auto** option fails to restore ownership:

```
export_recovery -mode=auto_noexport -item_id=xyz
               -new_owning_site=Site1
```

- To restore ownership of an entire assembly:

```
export_recovery -mode=auto -item_id=Assyl -remote_site=Site2
               -include_bom
```



---

## **data\_share**

---

### **DESCRIPTION**

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.

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 all the ODS sites to which an object is published.

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** options, 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 -g=group] -f={ send | publish | unpublish | delete_pubrec | register | unregister | delete_exprec | list_ods | check_ods | list_pub_info | find_duplicates | list_remote_co | list_replica_co | cancel_remote_co | cancel_replica_co} [-user=user-id] [-group=group-name] [-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=wso-name | | -filename=file-name } {-class=wso-class-name | | -classoffile=class-name} [-exclude=relation-type1 -exclude=relation-type2...] [-batch_size=number-of-objects-per-batch] [ {-revision-selector | -rev=rev-id}] [-include_bom] [-transfer] [-attach] [-exclude_files] [-latest_ds_version] [-exclude_folder_contents] [-include_bc] [-include_supercedures] [-report=report-file-name] [-h]
```

#### ARGUMENTS



Entries in parentheses are accepted abbreviations for arguments.

##### **-u**

Specifies the Teamcenter Engineering user ID. If the ID given is a privileged user, bypass privilege is set; otherwise, bypass is not set and only those objects accessible to the user are processed.

##### **-p**

Specifies the Teamcenter Engineering password associated with the user ID.

##### **-g**

Specifies the Teamcenter Engineering group associated with the user ID.

**-function**

Specifies the function to be performed; define one of the following options:

**publish (pub)**

Publishes objects to the given ODS sites. The objects to publish are determined by the **-item\_id**, **-folder** or **-filename** arguments.

**unpublish (unp)**

Unpublishes objects from the given ODS sites. The objects to unpublish are determined by the **-item\_id**, **-folder** or **-filename** arguments.

**send**

Sends objects to the specified remote sites.

The objects to send are determined by the **-item\_id**, **-folder** or **-filename** 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** option with specific item ID; no wildcards or other arguments are supported with this function.



To be used only if the master object has been deleted but publication records still exist at the ODS site.

**-delete\_exprec (dxr)**

Deletes export records for the specified sites for objects listed in the text file identified by the **-filename** and **-classoffile** options. Does not traverse item structure. Only privileged users may use this function.



To be used only as a last resort after attempting to delete export records using the **-verify** option 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** site 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.

**register**

Registers item IDs to the central item ID registry.

**unregister**

Unregisters item IDs from the central item ID registry. The register and unregister functions must be supplied with the **-item\_id** or **-filename** option.

**find\_duplicates**

Compares all of the item IDs at the remote site specified by the **-site** switch. The item IDs searched for may be filtered with the **-item\_id**, **-created\_after**, , and **-created\_before** switches. The output may be directed to a file using the **-report** option. The output is formatted to **csv** style, using comma-separated values.

**list\_remote\_co**

Lists master objects that are checked out by remote users based on the specified user ID, group name, and site name.

**list\_replica\_co**

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**

Cancels all remote checkouts based on the specified user ID, group name, and site name.

**cancel\_replica\_co**

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.

**-user**

Specifies the user id.

**-group**

Specifies the group name.

**-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)**

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](#).

**-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.



The list of relation types to be included is determined by the **IMAN\_relation\_export** preference. The **-exclude** option overrides the preference setting. For more information, see the *Configuration Guide*.

**-exclude\_files (exf)**

Excludes dataset files.

**-item\_id (item)**

Specifies the item ID or template of items to process. Mutually exclusive with **-folder**, **-filename**, and **-name** options. Required for the **-delete\_pubrec** option.

**-folder (fl)**

Specifies the name of a Teamcenter Engineering folder containing the list of objects process. Mutually exclusive with the **-name**, **-filename**, and **-item\_id** options.

**-name**

Specifies the name of a single workspace object to be preprocessed. 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 the list of IDs or names of objects to process. File entries are treated as IDs for Items and ItemRevisions and as names for other classes of objects. Mutually exclusive with the **-name**, **-folder** and **-item\_id** options. If the input file contains names, the **-classoffile** argument is required.

**-rev**

Specifies the ID of a specific item revision to be sent to a remote site. Valid only with the **-item\_id** option and with the **-send** function. Mutually exclusive with revision selectors.

**-class (cl)**

Specifies the Teamcenter Engineering class of the object specified by the **-name** argument. This option is only valid with the **-name** option. The default class is **Item**.

**-classoffile**

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**. Required if input file has names instead of IDs.

**-include\_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**.

**-transfer (tf)**

Transfers site ownership when sending objects. Site ownership is not transferred by default.

**-attach (att)**

Attaches to 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.

**-revision-selector**

If no revision selector is specified, the default selector is **all\_revisions**.

Identifies the item revisions to send. Also used as the revision rule for identifying components when processing assemblies. When used with the **-include\_bom** option 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. 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.

**-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=release-status-type**

Processes only the latest released revision with the given release status.

**-all\_released\_revs**

Sends all revisions with a release status; not valid when publishing.

**-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.

**-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\_pftemplate**

Identifies the related part family template to be exported when handling part family members.

**-part\_family\_bom\_treatment**

Identifies the part family objects associated with the assemblies to be exported. The option must be used in conjunction with the **-include\_bom** option. 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.

**-report (rep)**

Specifies to output a report to the specified file.

**-created\_after**

Restricts searches for duplicate items to those created at the target site after a specified date.

**-created\_before**

Restricts searches for duplicate items to those created at the target site before the specified date.

**-latest\_ds\_version (ldv)**

Sends only the latest dataset version. Unless this argument is specified, all dataset versions are sent.

**-continue\_on\_error**

Outputs the error in a report file and continues processing the other items. The **-report** switch must be given in order to see the error.

**-h**

Displays help for this utility.

**ENVIRONMENT**

As specified in *Manually Setting the Teamcenter Engineering Environment* in chapter 1, *Introduction*.

**FILES**

As specified in *Log Files* in chapter 1, *Introduction*.

**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:

- 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.

**EXAMPLES**



Required login 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 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:



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
```

---

**idsminetd**

---

**DESCRIPTION**

Serves as the iMAN Directory Services Manager (IDSM) launching program on UNIX systems. Located in the **\$IMAN\_ROOT/bin** directory, it is run at system startup and services all inbound requests for a new IDSM. For more information on IDSM, see *Multi-Site Collaboration Help*.

**SYNTAX**

**idsminetd [-dt] [-p=tcp\_port\_number] [-r=idsm-start-script]**

**ARGUMENTS**

**-d**

Specifies debug mode for standalone testing. The server runs in the foreground.

**-t**

Enhances logging.

**-p**

Specifies the port number on which the IDSM should run. Default is the system-assigned port number.

**-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 Setting the Teamcenter Engineering Environment* in chapter 1, *Introduction*.

**FILES**

As specified in *Log Files* in chapter 1, *Introduction*.

**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
```

## data\_sync

### DESCRIPTION

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 **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, via 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, UGS 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 **data\_sync** utility supports part family templates and members.

### SYNTAX

```
data_sync -u=user-id -p=password -g=group {-class=class-name
  -filename=file-name -item_id=template} -site=site-name | {-sync
  | -republish | -verify} [-update] [-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]
[-assert_extinct_site] [-assert_extinct_ods] [-latest_ds_version]
[-exclude_folder_contents] [-since=YYYY-MM-DD:HH:NN] [-bp]
[-batch_size=number-of-objects-per-batch] [-verbose] [-log] [-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.



If Teamcenter 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 Engineering database. If you do not supply these arguments, the utility attempts 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-name* 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.

**-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 will be synchronized; only those that were modified since the last time they were exported to the given site(s) will be synchronized. See restriction [1](#).

**-filename (fn)**

Specifies the name of the input file containing IDs or names of objects to update. See restriction [1](#)

**-classoffile (cof)**

Specifies the class of objects in the input file. This option is valid only with the **-filename** argument. If not specified, **Item** is the default class. **Folder** is a valid class for synchronization.

**-item\_id (item)**

Specifies the ID or template of items to update. See restriction [1](#).

**-sync**

Initiates the update process. See restriction [2](#).

**-republish (repub)**

Republishes objects that have been modified since last published. See restriction [2](#).

**-verify (veri)**

UGS recommends that you always use the **-item\_id=\*** option with the **-verify** option. If you use the **-class=item** option with the **-verify** option, it processes only the items that have been modified after their last export.

When used with the **-update** option, deletes the IXRs of objects for which replicas do not exist at the remote sites. When used without the **-update** option, generates a report. The report returns the following verification verdict codes:

- |          |                                    |
|----------|------------------------------------|
| <b>0</b> | Object does not exist.             |
| <b>1</b> | Object exists as a master copy.    |
| <b>2</b> | Object exists as a replica.        |
| <b>3</b> | Object was replaced by a POM stub. |

**-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 **-verify** option. See restriction 5.

**-assert\_extinct\_ods (aao)**

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 **-verify** option. See restriction 5.

**-update**

Performs a database update. Must be given in order for the **-sync**, **-republish**, or **-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 Engineering 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. 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 Help*.

**-revision-selector**

Valid only if both the **-sync** and **-update** options 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.

**-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**.



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\_pftemplate**

Identifies the related part family template to be exported when handling part family members.

**-part\_family\_bom\_treatment**

Identifies the part family objects associated with the assemblies to be exported. The option must be used in conjunction with the **-include\_bom** option. 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.

**-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. When used with the **-verify** option, the IXRs for the selected objects are destroyed without verifying if the object exists at the remote site. 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.

**-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.

**-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 option to analyze the performance of the utility.

**-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**.

**-h**

Displays help for this utility.

**-bp**

Displays best practices information.



## ENVIRONMENT

As specified in *Manually Setting the Teamcenter Engineering Environment* in chapter 1, *Introduction*.

## FILES

As specified in *Log Files* in chapter 1, *Introduction*.

## RESTRICTIONS

1. One of the following arguments must be supplied: **-class**, **-filename**, or **-item\_id**.
2. One of the following arguments must be supplied: **-sync**, **-republish**, or **-verify**.
3. The **-force** option can only be used along with the **-filename** or the **-item\_id** option. It does not function when used in combination with the **-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 **-verify** option.
6. The **-exclude\_files**, **-exclude=**, **-include\_bom**, **-disable\_modified\_only** and **-latest\_ds\_version** options can be used if both the **-sync** and **-update** arguments are supplied.
7. The **-classoffile** option currently supports only the **Item**, **ItemRevision**, **Dataset**, **Form**, and **Folder** classes.

## EXAMPLES



Required login information is omitted from the following examples.

- To generate a report of items that must be synchronized for a given site:

```
data_sync -class=Item -site=Site1 -sync
```

The report is output to **stdout**. No synchronization is performed.

- To synchronize all items copied to a site and output a report to a file:

```
data_sync -class=Item -site=Site1 -sync -update -report=report.lst
```



The default revision selector, **-same\_as\_last\_export**, is used.

- To synchronize the latest released revisions of items:

```
data_sync -class=Item -site=Site1 -sync -update -latest_released
```

- To synchronize all forms and datasets:

```
data_sync -class=Form -class=Dataset -site=Site1 -sync -update
```

- To republish all previously published items to the **Mfg\_ODS** ODS :

```
data_sync -class=Item -site=Mfg_ODS, -republish -update
```

- 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 -verify -update
```

- 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 -sync -update -force -report=report.lst
```

- To force synchronization of a single item or items that match a template:

```
data_sync -item_id=Eng* -site=Site1 -sync -update -force  
-report=rep.lst
```

- To destroy all the export records to a known extinct site:

```
data_sync -u=infodba -p=infodba -site=XSite -verify  
-update -assert_extinct_site
```

- To destroy all the publication records to a known extinct ODS site:

```
data_sync -u=infodba -p=infodba -site=XSite -verify  
-assert_extinct_ods
```

- To destroy export records, BVRs, and attachments of specific deleted replica item revisions :

```
data_sync -site=S1 -verify -update -fn=mylist -cof=ItemRevision
```

The **mylist** file has item revision names in the following format: **item123/A**

- To destroy export records of specific deleted replica datasets:

```
data_sync -site=S1 -verify -update -filename=mylist  
-classoffile=Dataset
```

The **mylist** file has dataset names in the following format: **dataset123**

- 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 -verify -update
```

- 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 -verify -update -report=rep.lst
```

- To generate a report of the IXRs, enter the following command on a single line:

```
data_sync -item_id=* -site=Site1 -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.

### Important Notes

1. When synchronizing items, all item revisions, BOMView revisions, BomViews, 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
```



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** option.
4. Because the **data\_sync** utility never involves any transfer of ownership, there is no need to perform export recovery if the utility terminates prematurely.
5. 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.
6. The **-verbose** option can be used to analyze the performance of the **data\_sync** utility. The **-verbose** option prints the system times at important stages during the process of synchronization.
7. UGS 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:

```
$IMAN_BIN/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** option.

**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 -u=infodba -p=infodba -g=dba -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.

## import\_file

### DESCRIPTION

Imports files into the Teamcenter Engineering 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.



The **import\_file** utility does not support the creation of custom item types.

### SYNTAX

```
import_file -u=user-id -p=password -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] [-revision=item-rev-num] [-ie={n
| y}] [-desc=string] [-v=volume-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.



If Teamcenter 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 Engineering database. If you do not supply these arguments, the utility attempts 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-name* 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.

#### -f

Imports a single file into Teamcenter Engineering. 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 Engineering 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 Engineering, 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. Each dataset type defines one or more named references to be associated with it. For more information, see *Type Help*. See restriction #2.

**-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.

**-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 Engineering volume where the imported file is placed.

**ENVIRONMENT**

As specified in [Manually Setting the Teamcenter Engineering Environment](#) in chapter 1, [Introduction](#).

**FILES**

As specified in [Log Files](#) in chapter 1, [Introduction](#).

**RESTRICTIONS**

1. When importing a file, the file name cannot be longer than 31 characters.
2. To create a dataset in Teamcenter Engineering, 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 Engineering as a **UGPART** dataset named **my\_bike\_dataset**, enter the following on a single line:

```
$IMAN_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 Engineering, 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=bikeN.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:

```
$IMAN_ROOT/bin/import_file -user=user-id -password=password
-group=group -i=input-file
```

---

**item\_export**

---

**DESCRIPTION**

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 **IMAN\_relation\_required\_on\_export** and **IMAN\_relation\_required\_on\_transfer** preferences.



The **-xcl\_refs**, **-xcl\_manifs**, **-xfr\_xcl\_refs**, and **-xfr\_xcl\_manifs** arguments for this utility are obsolete as of Teamcenter Engineering 2005 SR1. To exclude references and manifestations from the export operation, use the **-exclude** argument.

**SYNTAX**

```
item_export -u=user-id -p=password -g=group -dir=directory
{ -item_id=item-id |
[ -rev=revision-selector ] | -filename=input-file }
{ -owning_site=site-name | -target_site=site-name1, site-name2, ... }
[-exclude= relation-type1 -exclude=relation-type2...]
[ -reason=export-reason ] [ -latest_ds_version ] [ -include_bom ] [ -preview ]
[-report=file-name] [ -continue_on_error ] [ -xfr_top_lvl_only ] [ -xfr_top_asm_only ]
[ -xcl_files ]
[ -status=release-status ]
[-exclude_folder_contents ] [-dont_exclude_protected] [-classoffile=class-name]
[-email=email-address ] [-script=script-name]
[revision-selector] [-include_bc] [-include_supercedures][ -v ] [ -help ]
```

**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.



If Teamcenter 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 Engineering database. If you do not supply these arguments, the utility attempts 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-name* 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.

**-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** option.

**-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** or **USE\_STATUS**. If the **USE\_STATUS** keyword is given, you must specify a release status using the **-status** option. This is valid only when you are not transferring site ownership.

If used with an input file (**-i** option), 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.



The revision ID cannot be specified when using an input file.

**-filename**

Specifies the name of an input file that contains the list of item IDs to be exported.



This replaces the **-i** option, which is still supported for backward compatibility.

**-classoffile**

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.

**-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)**

Excludes the specified relation type from the export operation. 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\_requirement**, **IMAN\_specification**, or **IMAN\_master\_form** relation types.



Run the **install\_types** utility with the **list** argument to obtain the relation type names.

**-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. Only valid when site ownership is not being transferred.

**-include\_bom (bom)**

Exports all components if the given item is an assembly.

**-preview**

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**

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**

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\_pftemplate**

Identifies the related part family template to be exported when handling part family members.

**-part\_family\_bom\_treatment**

Identifies the part family objects associated with the assemblies to be exported. The option must be used in conjunction with the **-include\_bom** option. 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**

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 Engineering user account.

**-script**

Specifies the name of the script in the **IMAN\_BIN** directory that is executed after a successful export. If a script is already defined by the **IMAN\_post\_export\_script** site preference, the specified script will override the site preference entry. For more information, see the *Configuration Guide*.

**-revision\_selector**

Determines which item revision is exported with the item. The valid selectors are as follows:

<b>latest_revision (lt)</b>	Exports the latest revision only, regardless of whether it is a working or released revision.
<b>latest_working (lw)</b>	Exports the latest working revision only.
<b>latest_released (lr)</b>	Exports the latest released revision only with any release status.
<b>latest_working_or_any (lwoa)</b>	Exports the latest working revision. If no working revision exists, it exports the latest released revision with any release status.
<b>status (stat)</b>	Specifies the release status to be exported.

If no revision selector is given, all revisions are exported.



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.

**-h**

Displays help for this utility.

**ENVIRONMENT**

As specified in *Manually Setting the Teamcenter Engineering Environment* in chapter 1, *Introduction*.

**FILES**

As specified in *Log Files* in chapter 1, *Introduction*.

**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**

None.

---

## item\_import

---

### DESCRIPTION

Imports multiple items (in batch mode) into the Teamcenter Engineering 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 -g=group -dir=directory  
[-folder=folder-name] [-preview][-report [=file-name] [-continue_on_error]  
[-filename=file-name] [-classoffile=class-name] [-list_metafile]  
[-email=email-address] [-report=file-name] [-script=pre-import-script  
] [-continue_on_error] [-verbose] [-help]
```

### 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.



If Teamcenter 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 Engineering database. If you do not supply these arguments, the utility attempts 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-name* 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.

#### **-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 option must be used in conjunction with the **-include\_bom** option. 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 **IMAN\_post\_export\_script** site preference, this argument overrides preference setting. If specified as **NONE**, the script defined in the preference file will not be 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 Engineering user profile is used.

**-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.

**-h**

Displays help for this utility.

**ENVIRONMENT**

As specified in *Manually Setting the Teamcenter Engineering Environment* in chapter 1, *Introduction*.

**FILES**

As specified in *Log Files* in chapter 1, *Introduction*.

**RESTRICTIONS**

None.

**EXAMPLES**

None.



---

## item\_relink

---

### DESCRIPTION

Allows you to replace the external references for a duplicate item and its corresponding replica. The **item\_relink** utility works in conjunction with the **item\_rename** utility.

The **Bypass** option must be enabled to use this utility. The **Bypass** option enables or disables special bypass object protections for administrators. Bypass privileges are typically configured so that administrators can freely access any object in the database in order to perform maintenance. Therefore, **infodba** is the Teamcenter Engineering user ID 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*.



The **item\_relink** utility is used only with Multi-Site Collaboration to process production data. UGS recommends that a full backup of your database is performed. This will allow 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 TCFS 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.

### 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 BOMView, match the view type name.

To find the corresponding BOMView 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 in the report file for review.

### Exception

If unexpected Teamcenter Engineering 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 -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
-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.



If Teamcenter 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 Engineering database. If you do not supply these arguments, the utility attempts 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-name* 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.

**-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.

**-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.

**-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.

**-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\_rename** utility also uses this file.

**-verify**

Requests verification of compatibility between duplicates and replicas.

**-update**

Performs the link replacement.

**-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.

**-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 [Manually Setting the Teamcenter Engineering Environment](#) in chapter 1, [Introduction](#).

**FILES**

As specified in [Log Files](#) in chapter 1, [Introduction](#).

**RESTRICTIONS**

The following restrictions must be understood and adhered to when using the **item\_relink** utility:

1. This utility has to run with the **bypass** option on. Therefore, the **infodba** account should be used.
2. The **-replace** and **-refile** arguments must be supplied.
3. Either the **-rename** or **-verify** arguments must be supplied.
4. 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 should 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.

- Enter the following command on a single line to verify the items in the replacement folder and generate a report called **relink.rpt**. 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
```

- Check the **relink.rpt** file for errors and/or warning messages and examine the duplicates that are related to these messages. These duplicates may need to be corrected or have the ownership changed. To replace the links, enter the same command (after step 1), but replace the **-verify** argument with the **-update** argument.

---

## item\_rename

---

### DESCRIPTION

Allows you to change the item IDs for duplicate part numbers in a naming pattern. 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 must be enabled to use the **item\_rename** utility. The **Bypass** option enables or disables special bypass object protections for Teamcenter Engineering administrators. Bypass privileges are typically configured so that administrators can freely access any object in the database in order to perform maintenance. Therefore, **infodba** is the Teamcenter Engineering user ID.

### 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 Engineering 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 -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.



If Teamcenter 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 Engineering database. If you do not supply these arguments, the utility attempts 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-name* 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.

**-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 [Manually Setting the Teamcenter Engineering Environment](#) in chapter 1, [Introduction](#).

**FILES**

As specified in [Log Files](#) in chapter 1, [Introduction](#).

**RESTRICTIONS**

The following restrictions must be understood and adhered to when using the **item\_rename** utility:

1. This utility has to run with the **bypass** option on. Therefore, **infodba** should be used as the Teamcenter Engineering user ID.
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 in 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.

---

**plmxml\_export**

---

**DESCRIPTION**

Exports objects from Teamcenter Engineering 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.

**SYNTAX**

```
plmxml_export [-u=user-id -p=password -g=group ] -xml_file=xml-file-name  
-transfermode=transfermode_name {-item=item-id [-rev=item-revision-id ]  
[-rev_rule=revision-rule] [-svrule=saved-variant-rule ] | -class=class-name |  
-type=type-name | -imantypedef=iman-type -uid=uid-of-object} -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.



If Teamcenter 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 Engineering database. If you do not supply these arguments, the utility attempts 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-name* 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.

**-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.

**-item**

Specifies the ID of the item to be exported.

**-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.

**-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.

**-type**

Export all instances of a given type.

**-imantypedef**

Exports the definition of specified type.

**-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.

**-h**

Displays help for this utility.

**ENVIRONMENT**

As specified in *Manually Setting the Teamcenter Engineering Environment* in chapter 1, *Introduction*.

**FILES**

As specified in *Log Files* in chapter 1, *Introduction*.

**RESTRICTIONS**

None.

**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 Engineering:

```
plmxml_export -u=infodba -p=infodba -g=dba -xml_file=myobj.xml  
-uid="QRw4LZ0g1YomJAAAAAAAAAAAAA"
```

## plmxml\_import

### DESCRIPTION

Imports objects to Teamcenter Engineering from a PLM XML file. In case there are files for import as determined by transfer mode, the utility looks for files in the path specified by the **xml\_file** parameter. This utility is also used to import workflow templates.

### SYNTAX

```
plmxml_import [-u=user-id -p=password -g=group]  
-xml_file=name-of-xml-file -transfermode=transfermode_name  
[-type=object-type-to-import] [-log=log-file-name] [-h] [-s=TRUE |  
FALSE] [-import_mode=overwrite/ignore]
```

### 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.



If Teamcenter 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 Engineering database. If you do not supply these arguments, the utility attempts 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-name* 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.

#### **-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 template. A new "version" of the existing template is created in the database. If the template does not already exist, a new template is created.

**-type**

Specifies the name of the PLM XML element type. Only objects of this element type would be imported from the file.

**-log**

Specifies the name of a file to be used as the log file. If you do not specify a file name, a file is created using the name of the file specified by the **-xml\_file** option. The log file name format is *file-name\_log.txt*.

**-s**

Indicates that all imported objects are placed in a newly created folder within the user's **Newstuff** folder. The new folder is named using the right-most 32 characters of the XML file name. The default value for this argument is **FALSE**, and the behavior is not to save imported objects in the newly created folder unless this argument is set to **TRUE**.



Imported objects are saved in the system regardless of this argument. This argument controls the placement of imported objects into a folder. Multiple imports of the same object with the **-s** flag set results in multiple folders containing references to the same object.

**-h**

Displays help for this utility.

**-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 PLM XML Import/Export configuration objects for which this option is applicable are as follows:

**TransferMode**  
**ClosureRule**  
**PropertySet**  
**FilterRule**

**ENVIRONMENT**

As specified in *Manually Setting the Teamcenter Engineering Environment* in chapter 1, *Introduction*.

**FILES**

As specified in *Log Files* in chapter 1, *Introduction*.

**RESTRICTIONS**

None.

**EXAMPLES**

- To import all objects in the 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 Organization elements from the XML file that correspond to groups in Teamcenter Engineering, enter the following command on a single line:

```
plmxml_import -u=infodba -p=infodba -g=dba -xml_file=abc.xml  
-type=Organization -log=mylog.txt
```

- To import all objects in the XML file using the default context and save them in a newly created folder within the user's **Newstuff** folder, enter the following command on a single line:

```
plmxml_import -u=bob -p=password -g=group-name  
-xml_file=myxml.xml -s=TRUE
```

- To import a new 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 a workflow template and overwrite an existing template, creating a new 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
```

---

**step\_export**

---

**DESCRIPTION**

Exports Teamcenter Engineering 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] [-g=group] [-i=input-file |  
-item=item-id [-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.



If Teamcenter 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 Engineering database. If you do not supply these arguments, the utility attempts 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-name* 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.

**-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 Engineering relation containing the dataset.

**-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 Setting the Teamcenter Engineering Environment](#) in chapter 1, [Introduction](#).

**FILES**

As specified in [Log Files](#) in chapter 1, [Introduction](#) 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:



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:

```
$IMAN_ROOT/bin/step_export -u=infodba -p=password -g=dba -i=input-file
```

---

**step\_import**

---

**DESCRIPTION**

Imports product information from STEP-compliant physical files into the Teamcenter Engineering database.

**SYNTAX**

**step\_import** [-u=*user-id*] [-p=*password*] [-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.



If Teamcenter 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 Engineering database. If you do not supply these arguments, the utility attempts 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-name* 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.

**-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 Setting the Teamcenter Engineering Environment* in chapter 1, *Introduction*.

**FILES**

As specified in *Log Files* in chapter 1, *Introduction* 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.

---

**upgrade\_nx\_cam\_templates**

---

**DESCRIPTION**

Imports NX/CAM templates in to the Teamcenter database.



The **UGII\_BASE\_DIR** variable must be set to use this utility.

**SYNTAX**

**upgrade\_nx\_cam\_templates -u=infodba -p=password -g=dba**

**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.



If Teamcenter 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 Engineering database. If you do not supply these arguments, the utility attempts 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-name* 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.

**-h**

Displays help for this utility.

**ENVIRONMENT**

Requires the standard Teamcenter environment for running Integration Toolkit programs and also requires the **UGII\_BASE\_DIR** environment variable to be set.

**FILES**

None.

**RESTRICTIONS**

None.

**EXAMPLES**

- To display usage information for the utility, enter the following command on a single line:

```
upgrade_nx_cam_templates -h
```

- To import NX/CAM templates from the file system in to the Teamcenter database, enter the following command on a single line:

```
upgrade_nx_cam_templates -u=infodba -p=password -g=dba
```



---

Chapter

6     *Customization Utilities*

convert\_forms ..... 6-2  
tc\_erp\_schema ..... 6-8  
taxonomy ..... 6-10



---

**Chapter**

# **6**     *Customization Utilities*

---

This chapter describes utilities used to customize Teamcenter Engineering.

---

---

**convert\_forms**

---

**DESCRIPTION**

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

- Determine whether a given form type or all form types to be converted.
- Define attribute mapping between the file-based and storage-based forms.
- 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.
- Run the utility in batch mode without user intervention.
- Restart the process without data corruption in the event that the process is stopped or terminates abnormally.
- 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_info=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\_info** 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.



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 validate the XML files that you generate.



## The Process Options File

The **-process\_info** argument specifies information related to specific runs of the utility. The following example illustrates a sample **C:\temp\process\_info.txt** XML 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>
```

<b>InputFile</b>	Specifies either the file generated as output when the utility is run with the <b>-identify</b> option or the <b>ErrorFile</b> file generated during a conversion run.
<b>StartLine</b>	The line number in the <b>InputFile</b> that specifies the beginning of the block of forms to be converted. If the line number is not specified, the default value is <b>1</b> .
<b>EndLine</b>	The line number in the <b>InputFile</b> 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.
<b>LogFile</b>	Specifies the name of the file that logs information about dropped or truncated attribute values.
<b>ErrorFile</b>	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 <b>ErrorFile</b> file can be used as the input file when the utility is rerun to convert those forms.
<b>SuccessFile</b>	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.

### The Input Options File

The **input\_options** XML file is common to all runs of the utility, unlike the **process\_options** XML file, which is specific to a particular run of the utility. The following example shows the format of the **input\_options** XML 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 bold type in the example above. If all default values are assumed, the input options file can be omitted.

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

The following example shows a sample **C:\temp\input\_options.txt** XML 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 -identify -output_file=file-name[ -type=form-type] -convert
-process_info=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.



If Teamcenter 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 Engineering database. If you do not supply these arguments, the utility attempts 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-name* 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.

**-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\_info** and **-input\_options** arguments.

**-process\_info**

Specifies the process info 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 *The Process Options File*, earlier in this section.

**-input\_options**

Specifies the name of an input file containing information about how to convert forms. For more information about this file, see *The Input Options File*, earlier in this section. This argument is optional.

**-h**

Displays help for this utility.

**ENVIRONMENT**

As specified in *Manually Setting the Teamcenter Engineering Environment* in chapter 1, *Introduction*.

**FILES**

As specified in *Log Files* in chapter 1, *Introduction*.

**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
```

- 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_info=C:\temp\process_info.txt  
-input_options=C:\temp\input_options.txt
```

---

**tc\_erp\_schema**

---

**DESCRIPTION**

Updates the database with new classes and attributes. Once made, these updates cannot be reversed; therefore, you should carefully analyze what form types and attributes are required to send data to SAP. Once attributes are created for a form type, they can not be removed or modified. However, it is possible to create new form types to correct the attributes required, but this prevents previously created and populated form types from being correctly read or sent to ERP, because they do not correspond to the new definitions in the mapping file.



Before running this utility, check that there are no users on the system by running the **clearlocks** utility. For more information, see [clearlocks](#) in chapter 10, *Maintenance Utilities*.

This utility must be run if changes are made to the mapping file that require a database update, for example:

- New form types defined
- New attributes added to a form type
- LOV created/changed
- New occurrence note types are mapped to **BomComponent**
- New view type created

The following changes to the mapping file are picked up when you start a new session and do not require the **tc\_erp\_schema** program to be run:

- Default value
- Mandatory entry
- Runtime properties added



The **tc\_erp\_schema** utility creates LOV and default values for PSE note types. If the default is changed for a note type in Teamcenter Engineering, you must manually edit the mapping file to update the change. If the default is changed manually in the mapping file, you can run the **tc\_erp\_schema** utility to update Teamcenter Engineering.

**SYNTAX**

```
tc_erp_schema -u=infodba -p=password -g=dba -l=log-file-name
```

**ARGUMENTS**

**-u**

Specifies the user name. The **infodba** user account must be used to run this utility.

**-p**

Specifies the password associated with the **infodba** account.

**-g**

Specifies the group associated with the **infodba** account, which is **dba**.

**-l**

Optionally specifies a log file name.

**ENVIRONMENT**

As specified in *Manually Setting the Teamcenter Engineering Environment* in chapter 1, *Introduction*.

**FILES**

As specified in *Log Files* in chapter 1, *Introduction*.

If the mapping file fails to parse, syntax errors are reported in the following file:

```
/tmp/mapping_file_errors.txt
```

**RESTRICTIONS**

This utility must be run from the **infodba** account.

**EXAMPLES**

When the program completes successfully, you should see text similar to that in figure 6-1.

```
GOING TO DEFINE THE FORM DEFINITION CLASS "SAP_Basic_Data_FSC" FOR
THE FORMTYPE "SAP_Basic_Data"

ADDED THE ATTRIBUTE "Industry" TO THE CLASS "SAP_Basic_Data_FSC"
ADDED THE ATTRIBUTE "mat_grp" TO THE CLASS "SAP_Basic_Data_FSC"
ADDED THE ATTRIBUTE "Mat_type" TO THE CLASS "SAP_Basic_Data_FSC"
ADDED THE ATTRIBUTE "division" TO THE CLASS "SAP_Basic_Data_FSC"
ADDED THE ATTRIBUTE "basic_mat" TO THE CLASS "SAP_Basic_Data_FSC"
ADDED THE ATTRIBUTE "Sent_to_ERP" to the class "SAP_Basic_Data_FSC"
DEFINED THE CLASS "SAP_Basic_Data_FSC"
DEFINED THE FORMTYPE "SAP_Basic_Data"
ADDED LOV TO THE PROPERTY "Industry" OF THE FORMTYPE "SAP_Basic_Data"
ADDED LOV TO THE PROPERTY "Mat_type" OF THE FORMTYPE "SAP_Basic_Data"
ADDED LOV TO THE PROPERTY "division" OF THE FORMTYPE "SAP_Basic_Data"

, etc. for each Form Type, Attribute, LOV created

EXITING "tc_erp_schema"
```

**Figure 6-1. tc\_erp\_schema Output**

---

**taxonomy**

---

**DESCRIPTION**

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* **-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.



If Teamcenter 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 Engineering database. If you do not supply these arguments, the utility attempts 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-name* 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.

**-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 Setting the Teamcenter Engineering Environment](#) in chapter 1, [Introduction](#).

**FILES**

As specified in [Log Files](#) in chapter 1, [Introduction](#).

**RESTRICTIONS**

None.

**EXAMPLES**

None.



---

## Chapter

# 7 *Repeatable Digital Validation (RDV) Utilities*

batchmode_clearance_analysis.pl . . . . .	7-2
bomwriter . . . . .	7-3
cc_writer . . . . .	7-7
harvester.pl . . . . .	7-9
harvester_jt.pl . . . . .	7-12
rdv_import_usage . . . . .	7-15
rdv_run_audit_report . . . . .	7-17
tess_server . . . . .	7-19
RDV Cache Maintenance . . . . .	7-20
get_qpl_harvester_assemblies . . . . .	7-21
released_parts_collector . . . . .	7-26



---

## Chapter

# 7 *Repeatable Digital Validation (RDV) Utilities*

---

This chapter describes the utilities used to configure and maintain Teamcenter Engineering RDV and the RDV cache.

---

---

**batchmode\_clearance\_analysis.pl**

---

**DESCRIPTION**

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**

**IMAN\_ROOT/clearanceDB/scripts/**

**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.



If Teamcenter 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 Engineering database. If you do not supply these arguments, the utility attempts 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-name* 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.

**-bookmark**

Generates a bookmark file of the product.

**-h**

Displays help for this utility.

**ENVIRONMENT**

As specified in [Manually Setting the Teamcenter Engineering Environment](#) in chapter 1, [Introduction](#), and in the setup document provided in the **IMAN\_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
```

## bomwriter

### DESCRIPTION

Emits a bill of materials (BOM), in a variety of file formats, to a nominated file optionally restricted to selected areas of the BOM.

### SYNTAX

**bomwriter -u=user-id -p=password -g=group [-noprompt]**

Exactly one of the following:

**-bookmark=bookmark-file**

**-item\_list=input-file-name**

**-item=item-name**

Followed by one or more of the following:

**[-rev=revision]**

**[-selected=input-file-name]**

**[-subselected=input-file-name]**

**[-output\_file=output-file-name]**

**[-revision\_rule=config-rule]**

**[-show\_alternates=true | false]**

**[-show\_unconfigured=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]**

And exactly one of the following:

**-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 user with administration privileges. If this argument is used without a value, the operating system user name is used.



If Teamcenter 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 Engineering database. If you do not supply these arguments, the utility attempts 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-name* 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.

**-noprompt**

Specifies that if autologin fails, the system will not prompt for an interactive login. This is a standard autologin option.

**-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.

**-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, as follows:

**-format=index**

Format used with the **-selected** option. No modifiers.

**-format=ajt**

**AJT** format with the following modifiers:

**+nt**

**AJT** file attribute in Windows format.

**+unix**

**AJT** file attribute in UNIX format.

**+native**

**AJT** file attribute in machine-native (UNIX/Windows) format.

**+uidtag**

**AJT** file attribute in **uidtag** format.

**+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.

**-format=psup**

**psup** format with the **+prop=xxx** modifier representing comma-separated BOM line properties.

**-format=plmxml**

**plmxml** format with the following modifiers:

**+strict**

Minor errors are fatal.

**+type=xxx**

Use nondefault builder.

**+ua=xxx**

User attributes specifier.

**-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.

**-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**

(TEST) Use a packed BOM window. Valid values are **true** and **false**. The default value is **false**.

**-transient\_unpack**

(TEST) Use transient unpacking. Valid values are **true** and **false**. The default value is **false**.

**-smstring**

(TEST) 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\_alternates**

Specifies that alternates be shown. Defaults to the site default value.

**-show\_unconfigured**

Specifies that unconfigured lines (occurrence effectivity) be shown. Defaults to the site default value.

**-show\_variants**

Specifies that unconfigured variants be shown. Defaults to the site default value.

**-h**

Displays help for this utility.

**ENVIRONMENT**

As specified in [Manually Setting the Teamcenter Engineering Environment](#) in chapter 1, [Introduction](#).

**FILES**

As specified in [Log Files](#) in chapter 1, [Introduction](#).

**RESTRICTIONS**

None

**EXAMPLES**

- Run the **bomwriter** utility with the following arguments to write an **AJT** file for compilation (using **asciitot**) 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
```



## cc\_writer

### DESCRIPTION

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 Visualization Mockup from within Teamcenter.

### SYNTAX

```
cc_writer -u=user-name -p=password -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.



If Teamcenter 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 Engineering database. If you do not supply these arguments, the utility attempts 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-name* 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.

#### **-cc\_name**

Specifies the name of the Collaboration Context object. Either this argument or the **-cc\_uid** argument must be specified.



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. 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**(optional):*preference-name*,**target**  
**t**:*targetName1*,**key**:*keyName1*,...,**targ**  
**et**:*targetNameN*,**key**:*keyNameN*,**prop**:*propNameN*

**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 PLM XML file, **cc\_process\_cache.plmxml**, 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 Engineering Visualization Mockup.

## harvester.pl

### DESCRIPTION

Updates the QPL database with the latest changes in the product structure. The QPL database is used by the DesignContext application for 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.



To access the Teamcenter Engineering volumes containing JT data, this utility, the TCFS service, 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 Unigraphics 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**

### 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.



If Teamcenter 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 Engineering database. If you do not supply these arguments, the utility attempts 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-name* 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.

**-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](#) in chapter 1, [Introduction](#).

**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**

---

**DESCRIPTION**

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 by DesignContext for performing spatial and attribute queries.



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 visible in the Product Structure Editor application. 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**

**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.



If Teamcenter 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 Engineering database. If you do not supply these arguments, the utility attempts 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-name* 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.

**-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](#) in chapter 1, [Introduction](#).

#### 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
```



## rdv\_import\_usage

### DESCRIPTION

Imports usages from a PLM XML file to Teamcenter Engineering usage representations. The Teamcenter Engineering usage representations are occurrences of type **ArchPartUsage** that are created under the top level of a specified architecture breakdown. The architecture breakdown is associated with a product object, as specified by the DesignContext preference settings.

### SYNTAX

```
rdv_import_usage -u=user-name -p=password -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.



If Teamcenter 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 Engineering database. If you do not supply these arguments, the utility attempts 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-name* 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.

#### **-input\_file**

Specifies the XML input file from which the utility imports usages.

#### **-h**

Displays help for this utility.

### ENVIRONMENT

As specified in [Manually Setting the Teamcenter Engineering Environment](#) in chapter 1, [Introduction](#).

### FILES

As specified in [Log Files](#) in chapter 1, [Introduction](#).

**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.xml
```

The utility uses the **RDV\_IMPORT\_USAGE\_TM** transfer mode to convert the XML file to PLM XML format, via a style sheet.

## rdv\_run\_audit\_report


### DESCRIPTION

Generates an audit report for all BOM Lines of assemblies specified in an input file.

### SYNTAX

```
rdv_run_audit_report -u=user-id -p=password -g=group-name
                    -product_id=item-id -product_rev_id=revision-id
                    -v_rule=saved-variant-rule -rev_rule=revision-rule -option=usage | design
                    -input_file=text-file-containing-item-ids-of-assemblies-to-be-audited
```

### ARGUMENTS

- |                        |   |
|------------------------|---|
| <b>-u</b>              | <p>Specifies the user ID.</p> <p>This is generally <b>infodba</b> or another user with administration privileges. If this argument is used without a value, the operating system user name is used.</p> <div style="display: flex; align-items: flex-start;">  <div> <p>If Teamcenter Security Services single sign-on (SSO) is enabled for your server, the <b>-u</b> and <b>-p</b> arguments are authenticated externally through SSO rather than being authenticated against the Teamcenter Engineering database. If you do not supply these arguments, the utility attempts to join an existing SSO session. If no session is found, you are prompted to enter a user ID and password.</p> </div> </div> |
| <b>-p</b>              | <p>Specifies the password.</p> <p>If used without a value, the system assumes a null value.</p> <p>If this argument is not used, the system assumes the <i>user-name</i> value to be the password.</p>  |
| <b>-g</b>              | <p>Specifies the group associated with the user.</p> <p>If used without a value, the user's default group is assumed.</p>   |
| <b>-product_id</b>     | Specifies the item ID.  |
| <b>-product_rev_id</b> | Specifies the revision ID.  |
| <b>-v_rule</b>         | Specifies the saved variant rule by which the assembly is configured.   |
| <b>-rev_rule</b>       | Specifies the revision rule by which the assembly is configured.  |
| <b>-option</b>         | Specifies an option for the audit report, either <b>usage</b> or <b>design</b> .  |
| <b>-input_file</b>     | Specifies the name of the text file containing the item IDs of the assemblies to be audited.  |
| <b>-h</b>              | Displays help for this utility.   |

### ENVIRONMENT

- As specified in [Manually Setting the Teamcenter Engineering Environment](#) in chapter 1, [Introduction](#).

- **RDV\_debug** environment variable. If set, the Teamcenter Engineering **syslog** file contains additional debug information.
- **GMO\_enable\_audit\_score**, **GMO\_RDV\_enable\_audit\_criteria**, and **GMO\_enable\_new\_pnum\_match\_criteria** preferences. For more information, see the *Configuration Guide*.

**RESTRICTIONS**

None.

**EXAMPLES**

To generate an audit report for all the BOM Lines of assemblies mentioned in the input file, enter the following command on a single line:

```
$IMAN_ROOT/bin/ rdv_run_audit_report -product_id=TL109375  
-product_rev_id=001 -rev_rule="Latest Working" -option=design  
-input_file=c:\temp\ia_file.txt
```

---

## tess\_server

---

### DESCRIPTION

Starts the tessellation server which translates the UGMASTER/UGALTREP datasets to JT datasets.

### SYNTAX

**tess\_server**

### ARGUMENTS

**-s**

Starts the tessellation server.

**-c**

Stops the tessellation server.

**-h**

Displays help for this utility.

### ENVIRONMENT

As specified in [Manually Setting the Teamcenter Engineering Environment](#) in chapter 1, [Introduction](#).

### FILES

As specified in [Log Files](#) in chapter 1, [Introduction](#).

### 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
```

## **RDV Cache Maintenance**

This section describes the suite of utilities used to maintain the RDV cache.

---

## get\_qpl\_harvester\_assemblies

---

**DESCRIPTION**

Generates an XML file with the list of IAs/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.

The utility also finds new, modified, or deleted IAs/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 AIs/special assemblies that have been changed, released, deleted from, or added to the structure.
- Providing AI 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 IA), 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 powertrain modules

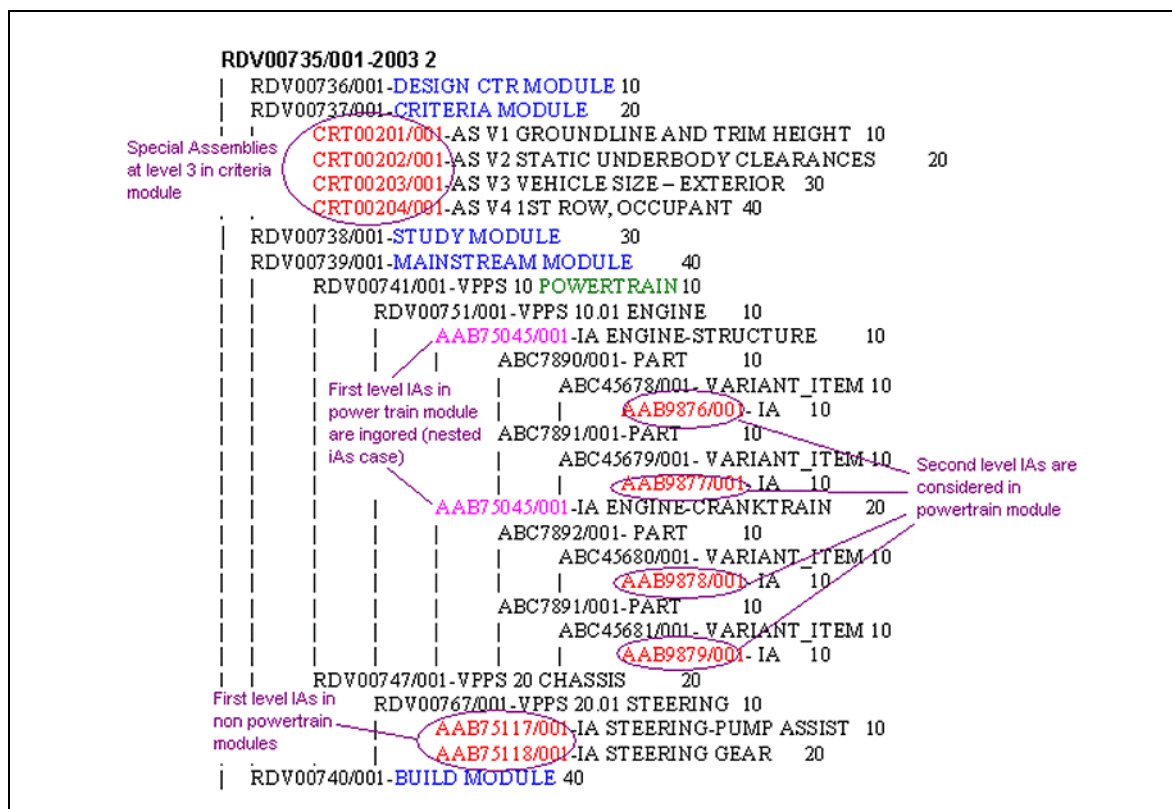
The cache administrator must specify the level in such a way that the assemblies selected at that level belong to the product assembly structure site. The administrator can specify different levels for different branches, for example:

```
--select_by_level=4:CRITERIA MODULE"
--select_by_level=3:STUDY MODULE"
```

The **get\_qpl\_harvester\_assemblies** utility traverses the assembly structure of the vehicle assembly by walking down from the top-node. Because IAs do not exist in all the branches of the vehicle assemblies, it needs to identify the special assemblies in those branches that do not have IAs. The **get\_qpl\_harvester\_assemblies** utility identifies the branch by its item name. The second-level assemblies in the vehicle structure represent the branch/modules and they are of type **CORP\_Proc\_Plan**. This utility is provided with the list of branches/modules, which do not have the IAs. 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 IAs (with item type **CORP\_install**). If this branch is integrated with powertrain modules, then it requires special handling, as these branches may contain nested IAs. Therefore, the program walks further down from the IA level to find lower level IAs for these powertrain modules.

To determine whether the branch is integrated with the power train module, it must check whether the names of the items of type **CORP\_Proc\_Plan** matches the powertrain module names specified as input to the program.

Figure 7-1 illustrates a typical vehicle assembly structure. Red nodes in the structure are those identified as IAs or special assemblies by the **get\_qpl\_harvester\_assemblies** program.



### Figure 7-1. Vehicle Assembly Structure



## SYNTAX

```
get_qpl_harvester_assemblies -u=user-id -p=password -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 options are specified.)
-item_id, -item_rev, and -rev_rule (If -bookmark option 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.



If Teamcenter 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 Engineering database. If you do not supply these arguments, the utility attempts 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-name* 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.

### -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.



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 IA list or the special list) even though they are not newly released.



If the assemblies do not satisfy one of the following requirements, they are not added to the IA or special list:

1. Assembly items must belong to a specific item type, such as the IA 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 powertrain product 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/new IAs 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 Engineering 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**

•

```
$IMAN_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
```

•

```
$IMAN_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
```

---

**released\_parts\_collector**

---

**DESCRIPTION**

Socket-based server application that receives requests from the **EPM-add-released-parts-queue** workflow handler via the Teamcenter Engineering 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* [**Command**]

Client mode:

**released\_parts\_collector\_server C** *host-name:port-number* [**Command**]

Command 1: **Cf** *host-name:port-number client-IA-file(XML)-with-absolute-path*

Command 2: **C** *host-name:port-number stop*

Command 3: **C** *host-name:port-number empty-master-list*

**ARGUMENTS**

**-S**

Specifies that the application functions as a server.

*hostname:port*

Indicates the port number at which this server should listen for incoming requests.

*path-to-released\_master\_list.xml*

Specifies the absolute path to the file to be created/updated with the released parts information. The released parts information is received from either the **EPM-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.

*server\_name:port*

Specifies the name and port number of the server. This is required when running the utility in client mode.

*fparts-list-file.xml*

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.

**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
```



---

Chapter

8     *Classification Utilities*

smlutility ..... 8-2  
icsxml ..... 8-6  
icsutility ..... 8-10





---

## Chapter

# 8 *Classification Utilities*

---

This chapter describes the utilities used to define, maintain, import, and export Classification classification hierarchy data.

---

---

**smlutility**

---

**DESCRIPTION**

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 have modified 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.



The **smlutility** program overwrites all attributes when run in **-update** mode. This program does not support the update of individual attributes.

**SYNTAX**

```
smlutility [-install | -import | -export | -delete | -list |  
-sync | -reassign_to_rev]  
smlutility -subclassesToStorageClasses sa-user-name sa-password  
sa-group[SML-Class-ID[, SML-Class-ID...]]
```

**ARGUMENTS****-install**

For example:

```
smlutility -install SA user SA password SA group module module::ICS
```

**-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, or ICO ID. 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)
```

**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).

**-list**

For example:

```
smlutility -list SA user SA password SA group [class|view|instance]
```

**-sync**

Synchronizes Classification definition data with remote sites, for example:

```
smlutility -sync SA user SA password SA group [definitions] [sites]
```

**definitions=**

**(VIEW | CLASS | GROUP | DICTIONARY):ID [,ID[,...]]**

**VIEW: ID=**

*Class Id::View Id*

**DICTIONARY: ID=**

*Attribute Id*

**CLASS: ID=**

*Class Id*

**KEYLOV: ID=**

*KeyLOV Id*

**GROUP: ID=**

*Group Id*



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=SITENAME[,SITENAME[,...]]
```

**Examples**

```
-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.

**-recurse**

Reassigns all ICOs of all subclasses of the identified class.

**view**

Reassigns all ICOs of the identified view.

**ico**

Reassigns the identified Classification instances (ICOs).

**-h**

Displays help for this utility.

**ENVIRONMENT**

As specified in *Manually Setting the Teamcenter Engineering Environment* in chapter 1, *Introduction*.

**FILES**

As specified in *Log Files* in chapter 1, *Introduction*.

**RESTRICTIONS**

1. This utility can only be run by users with system administration privileges.
2. 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 happens if you execute the following command:

```
smlutility -sync SA user SA password SA group
-definition=CLASS:myClass -sites=Paris
```

**EXAMPLES**

See Arguments, earlier in this section.

---

**icsxml**

---

**DESCRIPTION**

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 table 8-1.

**Table 8-1. Replacement Characters**

Symbol	Replacement Characters
&	<b>&amp;amp;</b>
<	<b>&amp;lt;</b>
>	<b>&amp;gt;</b>
,	<b>&amp;apos;</b>
“	<b>&amp;quote</b>

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**

**icsxml**

## 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.



If Teamcenter 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 Engineering database. If you do not supply these arguments, the utility attempts 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-name* 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.

**-export**

Exports the given subset of Classification data into an XML file.

**-import**

Imports Classification data from a given XML file into the Teamcenter Engineering database.

**-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.

**-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.

**-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 of the following values can be specified for the **-objtype** argument:

<b>A</b>	Attributes
<b>K</b>	Key-LOV
<b>C</b>	Classes
<b>I</b>	Instances

**-objid=object-id**

Specifies the ID of the object. Used in conjunction with the **-objtype** argument.



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 must be run in the iMAN shell environment.

**FILES**

As specified in [Log Files](#) in chapter 1, [Introduction](#).

**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:

```
ics_xml_utility -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:

```
ics_xml_utility -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:

```
ics_xml_utility -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:

```
ics_xml_utility -file=-2005.xml -filter=AK -export -objtype=A  
-objid=-2005 -u=infodba -p=pwd -g=dba
```

## icsutility

### DESCRIPTION

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.
- Manufacturing 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.

The icsutility also allows you to import class-specific and instance-specific files.

### SYNTAX

**icsutility**

### 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.



If Teamcenter 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 Engineering database. If you do not supply these arguments, the utility attempts 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-name* 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.

**-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 Engineering shell environment.

**FILES**

As specified in *Log Files* in chapter 1, *Introduction* and the **filetypeDefaults.txt** configuration file.

The entries in the **filetypeDefaults.txt** configuration file map the extensions of all imported files to specific Teamcenter Engineering data structures: GRM relationship type, dataset type, named reference, optional default tool, and optional subdirectory name. The mapping of file extensions to specific Teamcenter Engineering 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 Teamcenter Engineering ManufacturingResource 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
```

---

## Chapter

# 9 *Query Utilities*

build_fts_index . . . . .	9-2
default_queries . . . . .	9-8
find_appearances . . . . .	9-9
find_modified_datasets . . . . .	9-13
find_recently_saved_item_rev . . . . .	9-15
find_released_item_rev . . . . .	9-17
query_xml . . . . .	9-19



---

## Chapter

# 9 *Query Utilities*

---

This chapter describes the utilities used to reinstall default query forms, create queries to find recently saved or released item revisions, build keyword indexes, and maintain and run queries from an XML formatted file.

---

---

**build\_fts\_index**

---

**DESCRIPTION**

Builds keyword indexes for the Autonomy search engine on an object-by-object basis. These indexes enable the Teamcenter Engineering full-text keyword search and can index both the properties of Teamcenter Engineering 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.



Before running this utility, add the **TCENG\_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 adding preferences, see *My Navigator Help*.

The following file types are supported by Autonomy:

**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 above
- Microsoft Word Mac V4.x to V6.x
- Microsoft Word PC V2.0 to V5.5
- Quark QXD

**Spreadsheet**

The following spreadsheet file types are supported:

- Microsoft Works V3.x and later
- Microsoft Excel V3.x and later

**Presentation Graphics Formats**

The following presentation graphics file formats are supported:

- Shockwave Flash (with Autonomy Flashslave)
- Microsoft PowerPoint V4 and later



## SYNTAX

**build\_fts\_index.exe** **-u**=*user-name* **-p**=*password* **-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.



If Teamcenter 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 Engineering database. If you do not supply these arguments, the utility attempts 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-name* 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.

**-type**

Specifies the dataset type to index. Specified dataset types must be defined by the **TCENG\_fts\_indexed\_types** preference.



The **WSOM\_find\_list\_separator** preference must be set when using the **TCENG\_fts\_indexed\_types** preference.

If this argument is not specified, all dataset types defined by the **TCENG\_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**.

**-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.



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.

If not specified, the default is the current directory.

**-slavedir**

Specifies the full path to the directory where Autonomy import slaves are stored. If this argument is not specified, the directory specified by the **TCENT\_fts\_slave\_dir** preference is used.

**-idx**

Specifies an Autonomy intermediate IDX file that is created and stored in the **Current** directory. This file is removed after the utility runs. If not specified, the default IDX file is **tceng\_autonomy.idx**.

**-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**.

**-lang**

Defines the language type of the dataset files being indexed.

If this argument is not specified, the language type specified by the **TCENG\_default\_language\_type** preference is used.

If the **-lang** input differs from the value of the **TCENG\_default\_language\_type** preference, the existing index (based on the default language) is appended with the index based on the **-lang** argument. This behavior can be overridden using the **-f=index** option.

**-query**

Defines the name of the Teamcenter Engineering saved query that can be run to find all dataset objects that need to be indexed.

This argument is useful when you need to select a few individual datasets to be indexed with special options, such as a different language type.



This option works only with the **-f=index** option, **-f=append** option, and **-f=update** option. Also, the datasets returned by the query are indexed only if their dataset types are defined by the **TCENG\_fts\_indexed\_types** preference. Invalid dataset types are ignored.

**-entry**

Specifies the user entry name for the saved query.

**-value**

Specifies the value corresponding to the entry name. The **-entry** and **-value** arguments must be supplied in pairs.

**-log**

Specifies the name of the file into which the import statistics are written. The file is created and stored in the **IMAN\_TMP\_DIR** directory. 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 **tceng\_index\_processId.log**.

**-f**

Specifies one of the following operations:

- **index**  
Indexes datasets.
- **delete**  
Deletes invalid index entries from the Autonomy database. This option works only in conjunction with the **-type** argument.
- **append**  
Appends additional dataset index according to the language defined by the **-lang** argument. This option works only in conjunction with the **-lang** 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. UGS 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 **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.

If this argument is not specified, the default maximum number is **5000**.



This option works only in conjunction with the **-f=delete** argument.

**-maxbatch**

Defines the number of batch indexes or deletions that can be performed before the utility is automatically restarted.

The utility indexes the dataset files in batch mode. The number of dataset files per batch is defined by the **-filenumber** argument. Similarly, the utility also deletes the invalid index entries in batch mode. The number of query results per batch deletion is defined by the **-maxresults** argument.



The **-maxbatch** argument works in conjunction with the **-filenumber** and **-maxresults** arguments. The default value for the **-maxbatch** argument is **no limit**; therefore, the utility will not terminate and restart.

**-report**

Prints all the datasets that are indexed when the utility is run using the **-update** option. The **-report** option can be used to report datasets that were not indexed during the last indexing run.

**-db**

Defines the Autonomy database where index data is stored. If not specified, the data is stored in the database defined by the **TCENG\_fts\_database\_name** preference.

**ENVIRONMENT**

As specified in [Manually Setting the Teamcenter Engineering Environment](#) in chapter 1, [Introduction](#).

**FILES**

As specified in [Log Files](#) in chapter 1, [Introduction](#).

**RESTRICTIONS**

None.

**EXAMPLES**

- The following example indexes all dataset types defined by the **TCENG\_fts\_indexed\_types** preference and creates a **lastmoddate.txt** file that is saved in the **\$IMAN\_DATA/fts** directory. This utility can be stopped and restarted. Upon being restarted, the utility processes only the datasets that were created or modified after the last-modified date of the last object that was indexed.

```
$IMAN_BIN/build_fts_index
-u=user-name -p=password -g=group-name\
-ext=*.\\
-filenumber=500
```

- The following example indexes **Text** type datasets. Only the dataset files with **txt** extensions are indexed. The **TextTypeUID\_lastmoddate.txt** file is created and saved in the **\$IMAN\_DATA/fts** directory. This utility can be stopped and restarted. Upon being restarted, the utility processes only the datasets that were created or modified after the last-modified date of the last object that was indexed.

```
$IMAN_BIN/build_fts_index
-u=user-name -p=password -g=group-name\
-type=Text -ext=*txt.\
-filenumber=500
```

- The following example indexes the datasets returned by the specified query. Only dataset files with the **txt** extension are indexed. The **QueryUID\_lastmoddate.txt** file is created and saved in the **\$IMAN\_DATA/fts** directory.

Notice that no value is entered for the **Modified After** entry. This allows the utility to assign to the **Modified After** entry the last-modified date of the last object indexed, which in turn allows the utility to be stopped and restarted. Upon being restarted, the utility processes only the datasets that were created or modified after the last-modified date of the last object that was indexed. Without this input, the utility always processes all datasets returned by the query.

```
$IMAN_BIN/build_fts_index
-u=user-name -p=password -g=group-name\
-ext=*txt.\
-filenumber=500
-query=query-name\
-entry="Owning User"
-value=user-name
-entry="Modified After"
-value=\
```

- The following example deletes invalid index entries from the Autonomy database for all dataset types defined by the **TCENG\_fts\_indexed\_types** preference. The **lastindex.txt** file is created and saved in the **\$IMAN\_DATA/fts** directory. This command can be stopped and restarted. Upon being restarted, the utility processes only the query results that start from the last index of the last query result.

```
$IMAN_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 and creates a **TextTypeUID\_lastindex.txt** file that is stored in the **\$IMAN\_DATA/fts** directory. This command can be stopped and restarted. Upon being restarted, the utility processes only query results that start from the last index of the last query result.

```
$IMAN_BIN/build_fts_index
-u=user-name -p=password -g=group-name\
-f=delete
-type=Text
```

---

**default\_queries**

---

**DESCRIPTION**

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.



UGS recommends that you run this utility as the default Teamcenter Engineering 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* **-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.



If Teamcenter 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 Engineering database. If you do not supply these arguments, the utility attempts 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-name* 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.

**ENVIRONMENT**

As specified in [Manually Setting the Teamcenter Engineering Environment](#) in chapter 1, [Introduction](#).

**FILES**

As specified in [Log Files](#) in chapter 1, [Introduction](#).

**RESTRICTIONS**

None.

**EXAMPLES**

None.

## find\_appearances

### DESCRIPTION

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 or other attribute.



The output of this utility can be configured to print effectivity values.

### SYNTAX

```
find_appearances -u=user-name -p=password
-g=group-name [-item_id=item-id] [-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]
```

### 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.



If Teamcenter 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 Engineering database. If you do not supply these arguments, the utility attempts 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-name* 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.

**-item\_id**

Tracks the given item to find appearances for the appearance roots.

**-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.



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.

**ENVIRONMENT**

As specified in *Manually Setting the Teamcenter Engineering Environment* in chapter 1, *Introduction*.

**FILES**

As specified in *Log Files* in chapter 1, *Introduction*.

**RESTRICTIONS**

None.

**EXAMPLES**

Figure 9-1 shows the invocation of the **find\_appearances** 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 Pro
earances)), 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 13:56:03 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
--> QJPFI87ZAAGcRA      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
--> QNLFI8qKAAGcRA      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
=====
```

**Figure 9-1. find\_appearances Utility**

---

## find\_modified\_datasets

---

**DESCRIPTION**

Queries the database for a set of datasets. The utility finds the list of datasets, based on the following criteria:

- Dataset type
- Reference name
- Attached to a specific item revision type
- Attached to an item revision by a specific relation type
- Modified after a given date
- Having a given owning group
- Contained in the latest working revisions of items.

**SYNTAX**

```
find_modified_datasets -u=user-name -p=password -g=group-name  
-d=modified-date -revtype=item-revision-type -relation=relation-type  
-datasettype=dataset-type -reftype=reference-type -f=output-text-file-path  
[-owning_group=owning-group-of-dataset] [-working] [-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.

**-p**

Specifies the user's password. If this option is not specified, the utility looks for the password defined by the **IMAN\_USER\_PASSWORD** environment variable.

**-g**

Specifies the group associated with the user.

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

**-d**

Specifies the date after which objects were created or modified. Format for this value is **dd-mm-yyy hh:mm:ss**.

**-revtype**

Specifies the item revision type to which the required datasets are attached.

**-relation**

Specifies the relation with which the dataset is attached to the specified item revision type.

**-datasettype**

Specifies the dataset type for the query.

**-f**

Specifies the full path name of the CSV text file containing the item ID and revision ID that have the required datasets satisfying the search criteria.

**-owning\_group**

Specifies the owning group of the required datasets. Setting the owning group option restricts the search to datasets owned by users in the group.

**-working**

Specifies that the query look for the latest working revisions. If not specified, the query returns all the revisions that have been modified after the specified date. Released and working revisions are included in the output file. This argument is optional.

**-reftype**

Specifies the reference name of the dataset that should be used by the export program. The string passed to this parameter is printed to the output file.

**-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 Setting the Teamcenter Engineering Environment](#) in chapter 1, [Introduction](#). 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](#) in chapter 1, [Introduction](#).

**RESTRICTIONS**

None.

**EXAMPLES**

- To find the list of **UGMASTER** datasets with a **UGPART** named reference attached to the latest working **CORP\_Criteria Revisions** with the **IMAN\_specification** relation modified after 02-Jan-2005 10:10 a.m. and whose owning group is **dba**, enter the following command on a single line:

```
find_modified_datasets -u=infodba -p=password -g=dba
-d="02-Jan-2005 10:10"
-revtype="CORP_Criteria Revision" -relation=IMAN_specification
-datasettype=UGMASTER -reftype=UGPART -f=c:\temp\ug.txt
-owning_group=dba -working -nolog
```

- To find the list of **ALIAS\_PROJECT** datasets with an **Alias\_Wire** named reference attached to any **CORP\_Criteria Revisions** with the **IMAN\_specification** relation modified after 02-Jan-2005 10:10 a.m. with any owning user group, requiring a log file and without specifying the **-p** option, first set the **IMAN\_USER\_PASSWORD** variable and then enter the following command on a single line:

```
find_modified_datasets -u=infodba -g=dba -d="02-Jan-2005 10:10"
-revtype="CORP_Criteria Revision" -relation=IMAN_specification
-datasettype=UGMASTER -reftype=UGPART -f=c:\temp\ug.txt
```

---

## find\_recently\_saved\_item\_rev

---

**DESCRIPTION**

Allows you to search for item revisions with a **UGMASTER** dataset or BOMView 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**

**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.



If Teamcenter 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 Engineering database. If you do not supply these arguments, the utility attempts 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-name* 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.

**-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.

**-h**

Displays help for this utility.

**ENVIRONMENT**

This utility should be run from a shell where the Teamcenter Engineering environment is set.

**FILES**

As specified in [Log Files](#) in chapter 1, [Introduction](#).

**RESTRICTIONS**

None.

**EXAMPLES**

To list the item revisions saved after January 01, 2002:

```
$IMAN_ROOT/bin/find_recently_saved_item_rev  
-start_date="01-Jan-2002 00:00:00" -out_file=saved_items.txt
```

---

## find\_released\_item\_rev

---

**DESCRIPTION**

Allows you to create a query based on date and object type. The query is performed on the Teamcenter Engineering 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**

**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.



If Teamcenter 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 Engineering database. If you do not supply these arguments, the utility attempts 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-name* 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.

**-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.

**-h**

Displays help for this utility.

**ENVIRONMENT**

This utility should be run from a shell where the Teamcenter Engineering environment is set.

**FILES**

As specified in [Log Files](#) in chapter 1, [Introduction](#).

**RESTRICTIONS**

None.

**EXAMPLES**

To list all the item revisions released after January 01, 2002:

```
$IMAN_ROOT/bin/find_released_item_rev -start_date="01-Jan-2002 00:00:00"  
-out_file=released_items.txt
```



---

## query\_xml

---

**DESCRIPTION**

Creates, modifies, writes, deletes, and runs queries from an XML formatted file.

**SYNTAX**

**query\_xml**

**ARGUMENTS**

**-h**

Displays help for this utility.

**-f=command-file**

Specifies the fully qualified name of the XML file used to control processing. This argument is mandatory.

**-u=user-name**

Specifies the user under which you are logging in. This argument is optional but must be used with the **-p** argument.

**-p=password**

Specifies the password corresponding to the user specified by the **-u** argument. This argument is optional but must be used with the **-u** argument.

**-g=group**

Specifies the group corresponding to the user. This argument is optional and is used only if the **-u** and **-p** arguments are specified.

**-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.

**ENVIRONMENT**

As specified in [Manually Setting the Teamcenter Engineering Environment](#) in chapter 1, [Introduction](#). If the **IMAN\_TMP\_DIR** variable is not set, set it to a temporary location.

**FILES**

As specified in [Log Files](#) in chapter 1, [Introduction](#) 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**    0  
**upon success**

**Return value**    0  
**upon failure**

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
```

XML FILE  
FORMAT AND  
EXAMPLES

The XML file must conform to the format shown in figure 9-2.

```

<?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)>
  <!-- ATTLIST clauses_real value CDATA #REQUIRED -->
  <!ELEMENT clauses_display (#PCDATA)>
  <!-- ATTLIST clauses_display value CDATA #REQUIRED -->
  <!ELEMENT uniqueid EMPTY>
  <!-- ATTLIST uniqueid value CDATA #REQUIRED -->
  <!ELEMENT iflag EMPTY>
  <!-- ATTLIST iflag value CDATA #REQUIRED -->

  <!-- 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 -->
  <!-- ATTLIST query_pff_post outputFileFileName CDATA #REQUIRED -->

<!-- the pff to use (must be in db, and file to write data -->

<!-- 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 -->
<!-- full definition of the query will work. -->
  <!-- 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 -->
  <!-- ImanQueryCommandFile (ImanQueryCommand)* -->
  <!-- ATTLIST ImanQueryCommandFile site_name CDATA #IMPLIED -->
  <!-- ATTLIST ImanQueryCommandFile site_id CDATA #IMPLIED -->

]>

```

Figure 9-2. XML File Format (Continued)

```

<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>

```

**Figure 9-2. 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>

```

**Figure 9-3. 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>

```

**Figure 9-4. XML File Example**



---

## Chapter

# 10 *Maintenance Utilities*

install	10-2
site_util	10-13
tem.sh/.bat	10-16
uih_to_xml	10-17
Archive/Restore	10-19
auto_archive	10-20
auto_restore	10-23
object_archive	10-26
object_restore	10-28
Audit Manager	10-30
audit_archive	10-31
combine_audit_files	10-33
define_auditdefs	10-36
Backup and Recovery	10-38
backup_modes	10-39
backup_xmlinfo	10-42
object_backup	10-44
object_recover	10-46
sfr_instances	10-48
Engineering Translation Services	10-50
ETSDbConfig (.bat/sh)	10-51
etsServiceRegistry	10-52
etsServiceManager (.bat/sh)	10-54
ets_create_rqst	10-57
Migration	10-59
convert_distribution_lists	10-60
migrate_type_display_prefs	10-62
move_mso_forms	10-64
Project	10-65
create_project	10-66
update_project_bom	10-69
update_project_data	10-72
Subscription Manager	10-75
purge_invalid_subscriptions	10-76
System Maintenance	10-78

---

clearlocks . . . . .	10-79
imanhelp . . . . .	10-83
tc_mail_smtp . . . . .	10-84
install_event_types . . . . .	10-86
list_users . . . . .	10-90
purge_file_cache . . . . .	10-91
reset_user_home_folder . . . . .	10-93



---

## Chapter

# *10 Maintenance Utilities*

---

This chapter describes the utilities used to maintain your Teamcenter installation and database.

---

---

**install**

---

**DESCRIPTION**

Performs limited Teamcenter Engineering maintenance and Oracle database administration.



Some of the **install** utility arguments are reserved for UGS use only and should not be used by customers. Using the **install** utility with arguments designated as being for UGS use only can result in unpredictable behavior and may cause loss of data.

**SYNTAX**

```
install [-eim] [-encrypt] [-upgrade_v2_4] [-add_mes_class] [-make_user]
[-db_name_length] [-verify] [-regen_schema_file user-id password group]
[-regen_db_schema] [-header] [-index] -add_index infodba password dba
index_name unique_option class attr1 attr2...[-add_attr user-id password
group class name type length descriptor {strlen | ref_class}] [-mod_attr user-id
password group class name token {+ | -}] [-mod_class user-id password group
class token {+ | -}] [-ask_db] [-ayt] [-ask_version] [-set_version] [-lock_db
user-id password group ] [-unlock_db user-id password group ] [-vrf {-b | -l}]
[-stats] [-gen_xmit_file user-id password group ] [-priv name user-id password
group [-h]][-alter_str_len user password group class attr new_length]
```

**ARGUMENTS**

**-alter\_str\_len** *user password group class attr new\_length*

Specifies the class, attribute, and new length used to alter the string length of a given attribute. The attribute must be of type **POM\_string datatype**. The length value must be greater than the existing length and cannot exceed the 256-character maximum string length allowed by POM.

This argument only changes the Oracle table column definition and the corresponding POM data dictionary attribute definition. Therefore, care must be taken when increasing the length of an attribute to ensure that the calling code has allocated enough memory for the new length.

**-eim**

UGS use only.

**-encrypt**

Generates encrypted database connection string and modifies the **IMAN\_DB\_CONNECT** environment variable setting in the **\$IMAN\_DATA/iman\_profilevars** file.

**-upgrade\_v2\_4**

UGS use only.

**-add\_mes\_class**

UGS use only.

**-make\_user**

UGS use only.

**-db\_name\_length**

UGS use only.

**-verify**

UGS use only.

**-regen\_schema\_file**

Generates a new POM schema file. Requires Teamcenter Engineering 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 associated with the *user-id* value.

*group*

Specifies the group associated with the *user-id* value. In most cases, the group is **dba**. See restrictions #1 and #3.

**-regen\_db\_schema**

UGS use only.

**-header**

Examines the POM schema file header for information (without logging in to the database) and logs it to the **system.log** file. This can be useful for diagnosing problems when multiple databases exist.

**-index**

UGS use only.

**-add\_index**

Generates a new POM schema file. Requires Teamcenter Engineering 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 associated with the *user-id* value.

*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.



The order of attributes is important. Take care when creating indexes on a group of attributes.

**-add\_attr**

Adds attributes to classes after the class has been saved and populated. Requires two iterations of this utility: the first iteration uses the **-add\_attr** argument, the second iteration uses the **-regen\_schema\_file** argument. See restriction #1. 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 associated with the *user-id* value.

*group*

Specifies the group associated with the *user-id* value. In most cases, the group is **dba**.

*class*

Specifies the POM class name that contains the new attribute.

*name*

Specifies the name of the new attribute.

*type*

Specifies the attribute type using one of the following numeric codes:

**2001**

**POM\_char**

**2002**

**POM\_date**

**2003**

**POM\_double**

**2004**

**POM\_float**

**2005**

**POM\_int**

**2006**

**POM\_logical**

**2007**

**POM\_short**

Equivalent to **int**. Oracle does not save or lose any space in database storage because it uses a BCD-like representation. Therefore, small **ints** take less space than the long ones.

**2008****POM\_string****2009****POM\_typed\_reference**

The following conditions apply when referencing by tag:

If the ID of the referenced object changes, the reference to the object is maintained.

The referenced object cannot be deleted while it is referenced; therefore, referential integrity is maintained.

**2010****POM\_untyped\_reference**

Similar to a typed reference, but uses the root class POM object. Therefore, an untyped reference can point to any legitimate POM object.

**2011****POM\_external\_reference**

Similar to untyped references; however, external references do not impose referential integrity. Therefore, the use of external references is not recommended.

**2012****POM\_note**

Strings that can exceed the 256-character limit imposed on strings. Notes are used by the same ITK as strings. However, high-level form ITK may contain local string buffers that can cope with strings but not with the longer notes.

*length*

Specifies the array length. The VLA and large and small fixed-length arrays are accessed using the **POM\_ask/set/insert/delete/etc\_attr\_type[s]** ITK function. For more information, see *Integration Toolkit Function Reference*.

Specifies the array length using one of the following numeric codes:

**-1****VLA**

Variable length arrays can contain zero, one, or many values. It is not necessary for all instances to have the same number of entries in a VLA.

Adding a VLA attribute to an existing class causes all existing instances to have the VLA with no entries.

**1****Scalar****2 through 6****Small**

Small and large are internal terms for fixed-length arrays, which must have exactly the number of values in the array before being saved; otherwise, an error results.

7 and higher

Large

The internal difference between small and large arrays is that small arrays are stored as a set of in-line columns in the classes table and large arrays are stored in their own table.

*descriptor*

Describes the property using one of the following numeric codes:

**64**

**POM\_null\_is\_valid**

**256**

**POM\_public\_read**

**512**

**POM\_public\_write**

Public write makes an exception to the protected attributes of a class.

If the class is application protected but the attribute is public write, ITK functions such as **POM\_set\_attr\_type[s]** can be used on the attribute without first identifying the application.

**1024**

**POM\_transient**

Transient values are not saved in the database nor are they updated when refreshed.

*strlen | ref\_class*

Optional value. If *type* is **POM\_string** or **POM\_note**, this value is a string length. If *type* is **POM\_typed\_reference**, this value is the class name of the **POM\_typed\_reference**.

**-mod\_attr**

Sets attribute name properties **ON (+)** or **OFF (-)**. 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 associated with the *user-id* value.

*group*

Specifies the group associated with the *user-id* value. In most cases, the group is **dba**.

*class*

Specifies the parent class of the attribute.

*name*

Specifies the name of the new attribute.

*token*

The plus sign (+) sets attribute properties to **ON**. The minus sign (–) sets attribute properties to **OFF**.

See restrictions #1 and #2.

### **–mod\_class**

Sets class properties **ON** (+) or **OFF** (–). 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 associated with the *user-id* value.

*group*

Specifies the group associated with the *user-id* value. In most cases, the group is **dba**.

*class*

Specifies the affected classes.

*token*

The plus sign (+) sets properties to **ON**. The minus sign (–) sets properties to **OFF**.

See restrictions #1 and #2.

### **–ask\_db**

Returns one of the following codes based on the database type:

**2**

Oracle installed

**3**

Igres installed

**4**

RDB installed

**5**

Interbase

### **–ayt**

Executes test sequence to determine if the user can connect to the Oracle database specified by the **IMAN\_DB\_CONNECT** environment variable. Also determines if any Teamcenter Engineering data is present. Writes state of the database to the **system.log** file as follows:

**0**

Database okay and installed

**1**

Cannot connect

**2**

Database okay, needs install

**3**

Internal error

**-ask\_version**

Returns the current version of Teamcenter Engineering stored in the Oracle database.

**-set\_version**

UGS use only.

**-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 associated with the *user-id* value.

*group*

Specifies the group associated with the *user-id* value. In most cases, the group is **dba**.

See restrictions #1 and #4.

**-unlock\_db**

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 associated with the *user-id* value.

*group*

Specifies the group associated with the *user-id* value. In most cases, the group is **dba**.

See restriction #1.

**-vrf**

UGS use only.

**-stats**

UGS use only.



**-gen\_xmit\_file** *user 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.

**-priv**

Adds this custom privilege to the database. 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 associated with the *user-id* value.

*group*

Specifies the group associated with the *user-id* value. In most cases, the group is **dba**.

See restrictions #6 and #7.

**-h**

Displays help for this utility.

**ENVIRONMENT**

As specified in [Manually Setting the Teamcenter Engineering Environment](#) in chapter 1, [Introduction](#) and the following environment variables:

**IMAN\_DB\_CONNECT**  
**POM\_SCHEMA**  
**POM\_TRANSMIT\_DIR**

For more information, see the *Configuration Guide*.

**FILES**

As specified in *Log Files* in chapter 1, *Introduction* and the following:

- **\$IMAN\_DATA/iman\_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:

```
$IMAN_BIN/install -regen_schema_file infodba password dba
$IMAN_BIN/install -regen_schema_file -u=infodba -p=password -g=dba
```

2. Requires Teamcenter Engineering 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 in.

5. Only the following tokens can be changed on an existing (saved class):

**POM\_attr\_follow\_on\_export**

**POM\_attr\_export\_as\_string**

**POM\_attr\_is\_candidate\_key**

**POM\_public\_read**

**POM\_public\_write**

**POM\_public**

**POM\_null\_is\_valid**

For additional information, see *Integration Toolkit Function Reference*.

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 *Integration Toolkit Programmer's Guide*.

7. Rules-based object protection must be enabled in order to add and use new custom privileges.

#### EXAMPLES

- To return the site ID, enter the following command on a single line:

```
$TCROOT/bin/install -header > /dev/null grep "^Logical database"
system.log | sed "s/Logical database" site [^0-9]*//" |
sed "s/\([0-9.*\)\.*/\1/"
```

- To regenerate the POM transmit schema file, enter the following command on a single line:

```
$TCROOT/bin/install -gen_xmit_file infodba password dba
```

- To remove public (world) read permission from an attribute **att1** in **my\_class**, enter the following command on a single line:

```
$TCROOT/bin/install -mod_attr infodba password dba my_class
att1 POM_public_read -
```

- To define **my\_class** as being exportable, enter the following command on a single line:

```
$TCROOT/bin/install -mod_class infodba password dba my_class
POM_class_is_exportable +
```

- To add an eighty-character string attribute called **original\_name** to the **ImanFile** class, enter the following on a single line:

```
$TCROOT/bin/install -add_attr infodba password dba  
ImanFile original_name 2008  
1 64 80  
  
$TCROOT/bin/install -regen_schema_file infodba password dba
```

---

## site\_util

---

**DESCRIPTION**

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 -g=group -f=function [-site_id=site-id]
[-site_name=site-name] [-node_name=node-name] [-ods=y | n]
[-hub=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.



If Teamcenter 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 Engineering database. If you do not supply these arguments, the utility attempts 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-name* 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.

**-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.



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\_inc object because it is owned by another site. This can be run while users are logged in 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.

**-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 Help*.

**-hub**

Specifies whether the site being created or modified is a hub. For more information about hub configurations, see *Multi-Site Collaboration Help*.

**-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 Setting the Teamcenter Engineering Environment* in chapter 1, *Introduction*.

**FILES**

As specified in *Log Files* in chapter 1, *Introduction*.

**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 multi-site 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
```

---

**tem.sh/.bat**

---

**DESCRIPTION**

Starts the Teamcenter Environment Manager utility. When the **tem** command is entered on the command line with the **-s** option and a silent installation file, it bypasses the Teamcenter Environment Manager user interface and runs 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 Setting the Teamcenter Engineering Environment](#) in chapter 1, [Introduction](#).

**FILES**

As specified in [Log Files](#) in chapter 1, [Introduction](#).

**RESTRICTIONS**

None.



---

## uih\_to\_xml

---

**DESCRIPTION**

Converts existing UIH files to XML files that serve text and error messages.

**SYNTAX**

**uih\_to\_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.



If Teamcenter 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 Engineering database. If you do not supply these arguments, the utility attempts 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-name* 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.

*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 Setting the Teamcenter Engineering Environment](#) in chapter 1, [Introduction](#).

**FILES**

As specified in [Log Files](#) in chapter 1, [Introduction](#).

**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
```

## **Archive/Restore**

This section describes the utilities used to archive and restore objects.

## auto\_archive

### DESCRIPTION

Allows system level users to automatically archive objects from the Teamcenter Engineering database. The **auto\_archive** utility operates on a set of rules consisting of logic selection filters and queries, which provide the ability to specify criteria. The program executes the internal query of the database using the set of rules to obtain the list of workspace objects that can be marked for archive. Once the objects that need to be archived are located, the utility processes the archive and deletes the associated request object.

Logic filters and queries can be input through the input file (figure 10-1). This template file can also be generated through the utility.

```
//Rules Based Query Options . Enter the values inside <> with no leading/trailing spaces
//
-by_class      <ItemRevision>    // { ItemRevision | Item | Dataset | WorkspaceObject}
Defaults to ItemRevision
Type          <>                // {Type}
Owning User   <>                // {User_id}
Owning Group  <>                // {Group}
Created Before <>                // {Date created before} Ex: 23-Mar-2003 16:35
Created After <>                // {Date created after}
Modified Before <>              // {Date modified before} Ex: 23-Mar-2003 16:35
Modified After <>              // {Date modified after}
Accessed Before <>             // {Date accessed before} Ex: 23-Mar-2003 16:35
Accessed After <>             // {Date accessed after}
Released Before <>            // {Date released before} Ex: 23-Mar-2003 16:35
Released After <>            // {Date released after}
Released Status <>           // {Released status name}
//
//Rules Based Filter Options
//
-non_owner_objs <true>        // { true | false } Defaults to true
-archived_objs <true>        // { true | false } Defaults to true
-remote_objs <true>          // { true | false } Defaults to true
-checkedout_objs <true>      // { true | false } Defaults to true
-inprocess_objs <true>       // { true | false } Defaults to true
-archive_pending_objs <false> // { true | false } Defaults to false
-objs_ref_by_unarchived_parent <true> // { true | false } Defaults to true
//
//Other Options
//
-medianame <>                // {medianame} if no value is provided Default Media is used
-comment <>
//
```

**Figure 10-1. Auto Archive Template File**

### SYNTAX

#### **auto\_archive**

### 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.



If Teamcenter 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 Engineering database. If you do not supply these arguments, the utility attempts 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-name* 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.

### **-lm**

Lists the storage media defined in the database. (Optional.)

### **-interactive**

Provides the following additional capabilities during an **auto\_archive** session:

- Retry if storage media is not ready
- Switch to a different storage media
- Change the metafile name
- Show error details, if any

### **-of**

Generates a template file with allowed logic selection filters and queries. This file can be edited and used to specify the user's archive criteria using the **if=** argument. (Optional.)

### **-f**

Optional function argument that can specify one of the following values:

- =archive** Archives the objects to the storage media as specified by logic selection filters and queries defined in the file specified by the **if=** argument.
- =list** Lists the objects that can be archived as specified by logic selection filters and queries defined in the file specified by the **if=** argument.

### **-if**

Specifies the logic selection filters and queries through an input file. This argument is used in conjunction with the **-f** argument. The template input file for specifying archive criteria can be generated using the **-of** argument.

### **-presorted\_file**

Archives the list of presorted rules-based objects specified through an input file. The file format must follow the list generated using the **-f=list** option. (Optional.)

**-h**

Displays help for this utility.

**ENVIRONMENT**

If archiving for **ContentStorage** media, ensure that Neutral APIs are coded and all library paths are included.

**FILES**

The **Archive.out** file records specific information about the objects and whether the archive operation succeeded or failed.

**RESTRICTIONS**

This utility can be used to archive commonly used Teamcenter Engineering objects and their contents, such as items, item revisions, and datasets.

Successful completion of the utility with the **f=archive** and filter option **-objs\_ref\_by\_unarchived\_parent** set to true may result in child items ready for a second round of the archive process. You must run the utility again with the **f=list** option to check for child items. If found, rerun the utility using the **f=restore** option.

**EXAMPLES**

- Enter the following command to list the storage media types:

```
auto_archive -u=infodba -p=password -g=system -lm
```

- Enter the following command to generate the template file with editable filter and query options:

```
auto_archive -u=infodba -p=password -g=system -of=template.Out
```

- Enter the following command to initiate an archive process specified by the **criteria\_input** criteria file:

```
auto_archive -u=infodba -p=password -g=system -f=archive  
-if=criteria_input -interactive
```

- Enter the following command to generate a list of archivable objects specified by the **criteria\_input** criteria file:

```
auto_archive -u=infodba -p=password -g=system -f=list  
-if=criteria_input -interactive
```

- Enter the following command to store the list in a file named **sorted\_objects**:

```
auto_archive -u=infodba -p=password -f=list -if=criteria_input  
-interactive >sorted_objects
```

- Enter the following command to archive presorted objects stored in the **sorted\_objects** file:

```
auto_archive -u=infodba -p=password -g=system  
-presorted_file=sorted_objects
```

## auto\_restore

### DESCRIPTION

Allows system level users to automatically restore the objects from the Teamcenter Engineering database. The **auto\_restore** utility operates on a set of rules consisting of logic selection filters and queries, which provide the ability to specify criteria. The program executes the internal query through the database using the set of rules to obtain the list of workspace objects that can be marked for restoration. Once the objects that need to be restored are located, the utility processes the restoration and deletes the associated request (**TransferPending**) object.

Logic filters and queries can be input through the input file (figure 10-2). This template file can also be generated using the utility.

```
//Rules Based Query Options . Enter the values inside <> with no leading/trailing spaces
//
-by_class      <ItemRevision>    // { ItemRevision | Item | Dataset | WorkspaceObject }
Defaults to ItemRevision
Type          <>                // {Type}
Owning User    <>                // {User_id}
Owning Group   <>                // {Group}
Created Before <>                // {Date created before} Ex: 23-Mar-2003 16:35
Created After  <>                // {Date created after}
Modified Before <>              // {Date modified before} Ex: 23-Mar-2003 16:35
Modified After <>                // {Date modified after}
Archived Before <>              // {Date archived before} Ex: 23-Mar-2003 16:35
Archived After <>                // {Date archived after}
Released Before <>              // {Date released before} Ex: 23-Mar-2003 16:35
Released After <>                // {Date released after}
Released Status <>              // {Released status name}
//
//Rules Based Filter Options
//
-non_owner_objs <true>          // { true | false } Defaults to true
-remote_objs   <true>          // { true | false } Defaults to true
-restore_pending_objs <false>  // { true | false } Defaults to false
-objs_ref_by_archived_parent <false> // { true | false } Defaults to false
//
//
```

**Figure 10-2. Auto Restore Template File**

### 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.



If Teamcenter 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 Engineering database. If you do not supply these arguments, the utility attempts 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-name* 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.

**-lm**

Lists the storage media defined in the database. (Optional.)

**-interactive**

Provides the following additional capabilities during an **auto\_restore** session:

- Retry if storage media is not ready
- Switch to a different storage media
- Change the metafile name
- Show error details, if any

**-of**

Generates a template file with allowed logic selection filters and queries. This file can be edited and used to specify the restoration criteria using the **if=** argument. (Optional.)

**-f**

Optional function argument that can specify one of the following values:

- =restore**     Restores the objects to the storage media as specified by logic selection filters and queries defined in the file specified by the **if=** argument.
- =list**         Lists the objects that can be restored as specified by logic selection filters and queries defined in the file specified by the **if=** argument.

**-if**

Specifies the logic selection filters and queries through an input file. This argument is used in conjunction with the **-f** argument. The template input file for specifying restoration criteria can be generated using the **-of** argument.

**-presorted\_file**

Restores the list of presorted rules-based objects specified through an input file. The file format must follow the list generated using the **-f=list** option. (Optional.)

**-h**

Displays help for this utility.

**ENVIRONMENT**

If restoring for **ContentStorage** media, ensure that Neutral APIs are coded and all library paths are included.

**FILES**

The **Restore.out** file records specific information about the objects and whether the restoration operation succeeded or failed.



**RESTRICTIONS**

This utility can be used to restore commonly used Teamcenter Engineering objects and their contents, such as items, item revisions, and datasets.

Successful completion of the utility with the **f=restore** and filter option **-objs\_ref\_by\_unarchived\_parent** set to *true* may result in child items ready for a second round of the restore process. You must run the utility again with the **f=list** option to check for child items. If found, rerun the utility using the **f=restore** option.

**EXAMPLES**

- Enter the following command to list the storage media types:

```
auto_restore -u=infodba -p=password -g=system -lm
```

- Enter the following command to generate the template file with editable filter and query options:

```
auto_restore -u=infodba -p=password -g=system -of=template.Out
```

- Enter the following command to initiate a restore process specified by the **criteria\_input** criteria file:

```
auto_restore -u=infodba -p=password -g=system -f=restore  
-if=criteria_input -interactive
```

- Enter the following command to generate a list of restorable objects specified by the **criteria\_input** criteria file:

```
auto_restore -u=infodba -p=password -g=system -f=list  
-if=criteria_input -interactive
```

- Enter the following command to store the list in a file named **sorted\_objects**:

```
auto_restore -u=infodba -p=password -g=system -f=list -if=criteria_input  
-interactive >sorted_objects
```

- Enter the following command to restore presorted objects stored in the **sorted\_objects** file:

```
auto_restore -u=infodba -p=password -g=system  
-presorted_file=sorted_objects
```

---

## object\_archive

---

### DESCRIPTION

Archives all transfer pending requests in a single utility session. The **object\_archive** utility is the companion to the [object\\_restore](#) utility and must not be confused with the **object\_backup** and **object\_recover** utilities.

### TRANSFER PENDING REQUESTS

When a user archives an object from their workspace, a transfer pending request is created. The **object\_archive** utility archives all transfer pending requests in a single utility session.

### BATCH MODE AND INTERACTIVE PROCESSING

The **object\_archive** utility runs in either nonverbose batch mode or interactive mode.

In batch mode, the utility processes the transfer pending requests and notifies each user through Teamcenter Engineering mail when each object is successfully archived.

Interactive mode is set using the **-i** argument and provides the following additional capabilities during an **object\_archive** session:

- Retry if storage media was not ready.
- Switch to a different storage media.
- Change the metafile name.



To successfully archive objects using the **object\_archive** utility, the storage media must be defined and online. For more information on storage media, see *Motif System Administration or Organization Help*.

### SYNTAX

**object\_archive -u=user-id -p=password -g=system**  
**[-m=device-name] [-i] [-lm] [-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.



If Teamcenter 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 Engineering database. If you do not supply these arguments, the utility attempts 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-name* 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.

**-m**

Specifies the device used to store archived objects. See restriction #5.

**-lm**

Creates a list of previously defined Teamcenter Engineering storage media. Only media appearing in this list can be used with the **-m** argument.

**-h**

Displays help for this utility.

**ENVIRONMENT**

As specified in [Manually Setting the Teamcenter Engineering Environment](#) in chapter 1, [Introduction](#).

**FILES**

As specified in [Log Files](#) in chapter 1, [Introduction](#) and the following:

archive.out

Contents of all **object\_archive** mail messages sent to users. Typically used to review archive status messages from an **object\_archive** session.

**RESTRICTIONS**

1. The **object\_archive** utility can only be run by members of the **system** group.
2. Only a system administrator can archive released objects.
3. Users archiving objects must have the proper permissions (read, write, delete, and copy).
4. The system administrator is responsible for establishing and controlling storage media types and file naming conventions.
5. Storage media must be available (defined and on line) before the **object\_archive** operation can successfully archive an object.

For more information about storage media, see *Motif System Administration* or *Organization Help*.

**EXAMPLES**

- To archive objects in batch mode to device **dat01**, enter the following command on a single line:

```
$IMAN_ROOT/bin/object -u=infodba -p=password -g=system -m=dat01
```

- To archive objects in interactive mode to device **dat01**, enter the following command on a single line:

```
$IMAN_ROOT/bin/object_archive -u=infodba -p=password  
-g=system -m=dat01 -i
```

---

**object\_restore**

---

**DESCRIPTION**

Restores all transfer pending requests in a single utility session. The **object\_restore** utility is the companion to the *object\_archive* utility and must not be confused with the **object\_backup** and **object\_recover** utilities.

**Transfer Pending Requests**

When a user restores an object to the database, a transfer pending request is created. The **object\_restore** utility restores all transfer pending requests in single utility session.

**Batch Mode and Interactive Processing**

The **object\_restore** utility runs in either nonverbose batch mode or interactive mode.

In batch mode, the utility processes the transfer pending requests and notifies each user via Teamcenter Engineering mail when each object is successfully restored.

Interactive mode is set using the **-i** argument and provides the following additional capabilities during an **object\_restore** session:

- Retry if storage media was not ready.
- Switch to a different storage media.
- Change the metafile name.

**SYNTAX**

**object\_restore -u=user-id -p=password -g=system**  
**[-m=device-name] [-i] [-lm] [-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.



If Teamcenter 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 Engineering database. If you do not supply these arguments, the utility attempts 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-name* 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.

**-m**

Specifies the device from which the objects are restored.

**-i**

Restores objects in interactive mode.

**-lm**

Creates a list of online Teamcenter Engineering storage media. Only media appearing in this list can be used with the **-m** argument.

**-h**

Displays help for this utility.

**ENVIRONMENT**

As specified in *Manually Setting the Teamcenter Engineering Environment* in chapter 1, *Introduction*.

**FILES**

As specified in *Log Files* in chapter 1, *Introduction* and the following:

**restore.out**

Contents of all **object\_restore** mail messages sent to users.

**RESTRICTIONS**

1. The **object\_restore** utility can only be run by members of the **system** group.
2. Users restoring objects must have read permission on the object.
3. Storage media must be available (defined and online) before **object\_restore** can successfully restore an object. For more information about storage media, see *Motif System Administration* or *Organization Help*.

**EXAMPLES**

- To restore objects in batch mode from device **dat01**, enter the following command on a single line:

```
$IMAN_ROOT/bin/object_restore -u=infodba -p=password -g=system
-m=dat01
```

- To restore objects in interactive mode from device **dat01**, enter the following command on a single line:

```
$IMAN_ROOT/bin/object_restore -u=infodba -p=password
-g=system -m=dat01 -i
```

## **Audit Manager**

This section describes the utilities used to create audit definition objects, combine audit log files, and archive audit logs.

## audit\_archive

### DESCRIPTION

Searches the database for audit log records based on input criteria. Once it finds the records that need to 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 -g=group [-h] [-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] [-created_after] [-overwrite]
```

### 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.



If Teamcenter 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 Engineering database. If you do not supply these arguments, the utility attempts 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-name* 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.

#### **-delete\_record**

Specifies that audit log records are deleted after being archived.

#### **-type**=type-name

Specifies the object type to archive.

#### **-class**=class-name

Specifies the class name for the object type to be archived.

#### **-event**=event-name

Specifies the event type to archive.

#### **-id**=object-id

Specifies the ID of the object being archived.

**-name=***object-name*

Specifies the name of the object being archived.

**-owner=***object-owner-id*

Specifies the owner ID of the object being archived.

**-created\_before=***date*

Specifies that objects created before the specified date are archived.

**-created\_after=***date*

Specifies that objects created after the specified date are 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.

**ENVIRONMENT**

As specified in [Manually Setting the Teamcenter Engineering Environment](#) in chapter 1, [Introduction](#).

**FILES**

As specified in [Log Files](#) in chapter 1, [Introduction](#).

**RESTRICTIONS**

Requires Teamcenter Engineering administrator privileges.



---

## combine\_audit\_files

---

**DESCRIPTION**

Combines all the log files into an **iMANAuditLog.txt** or **iMANAuditLog.xml** file.

**SYNTAX**

**combine\_audit\_files.pl**

**ARGUMENTS****source\_dir**

Specifies the source directory containing audit log files generated during Teamcenter Engineering 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 NT platforms that install Perl, and the program is in their path. However, if they do not, you can use **%IMAN\_ROOT%\bin\perl combine\_audit\_files.pl source\_dir target\_dir** on Windows NT, or **\$IMAN\_ROOT/bin/perl combine\_audit\_files.pl source\_dir target\_dir** on UNIX.

**FILES**

The audit log files to be combined (**iman\_auditlog\_\*\*\*\*.txt**, **iman\_auditlog\_\*\*\*\*.xml**) are at **source\_dir** directory. The combined master log files (**iMANAuditLog.txt**, **iMANAuditLog.xml**) are at **target\_dir** directory. If the master files **iMANAuditLog.txt**, **iMANAuditLog.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 **iman\_auditlog\_\*\*\*\*.xml** and **iman\_auditlog\_\*\*\*\*.txt** files, moves them to the **C:\temp\audit** directory, combines them, and appends them to the **iMANAuditLog.xml** and **iMANAuditLog.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 **iman\_auditlog\_\*\*\*\*.txt** and **iman\_auditlog\_\*\*\*\*.xml** files in the source directory, move them to the target directory, combine them, and append them to the **iMANAuditLog.txt** and **iMANAuditLog.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
```

- If the **IMAN\_audit\_delimiter** preference is changed to a value other than the default value (^), you must manually edit the first line of the **iMANAuditLog.txt** master audit file to reflect the new delimiter.

For example, if the new delimiter is a dollar sign (\$), the first line of the **iMANAuditLog.txt** file must be changed to:

```
ObjectUID$objectId$objectName$revision$objectTypeName$eventTypeName  
$userId$loggedDate$properties
```

- If the **IMAN\_XML\_ENCODING** environment variable in the **iman\_profilevars.bat** file is changed to a value other than the default value **iso-8859-1**, you must edit the first line of the **iMANAuditLog.xml** master audit file and the **iMANAuditLog.xsl** and **iMANAuditLogSchema.xsd** files in the sample/audit directory.

For example, if the new encoding is **Shift\_JIS** (Japanese), the first line of the **iMANAuditLog.xml** file is changed to:

```
<?xml version="1.0" encoding=" Shift_JIS"?>
```

UGS 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 (**iMANAuditLog.xml**, **iMANAuditLog.xsl**, **iMANAuditLogSchema.xml**, and **iMANAuditLog.js**) must be in the same directory. Opening **iMANAuditLog.xml** in Microsoft Internet Explorer 5.5 or higher presents audit data in a table (figure 10-3) that can be sorted and filtered by column.

objectUID	objectId	objectName	revision	objectTypeName	eventTypeName	audit	loggedDate	properties
wMbaE3WaevgND		cdl	B	ItemRevision	_Modify	thum	2002-01-15 12:00:48	creation_date=15-Jun-2002 11:50
wMbaE3WaevgND		cdl	B	ItemRevision	_Modify	shen	2002-01-15 12:00:47	creation_date=15-Jun-2002 11:50
wMbaE3WaevgND		cdl	B	ItemRevision	_Modify	thum	2002-01-15 12:00:46	creation_date=15-Jun-2002 11:50
wMbaE3WaevgND		cdl	B	ItemRevision	_Check_In	shen	2002-01-15 12:00:39	Change ID=ba-Reason=
wMbaE3WaevgND		cdl	B	ItemRevision	_Check_Out	shen	2002-01-15 12:00:35	Change ID=ba-Reason=
wMbaE3WaevgND	temp2	cdl		Item	_Check_In	shen	2002-01-15 12:00:34	Change ID=ba-Reason=

**Figure 10-3. Audit Data Table**

The files are described as follows:

**iMANAuditLog.xml**

XML data source of audit records. The file is produced by executing the **combine\_audit\_files.pl** script.

**iMANAuditLog.xsl**

XML style sheet for displaying the **iMANAuditLog.xml** file in the Microsoft Internet Explorer Web browser.

**iMANAuditLogSchema.xsd**

XML schema for defining XML data structure and data types.

**iMANAuditLog.js**

JavaScript for adding dynamic effects to the HTML presentation.

If you use the **iMANAuditLog.xml** for purposes other than displaying the audit log in a Web browser, you can modify any of the files. For example, you can modify the **iMANAuditLog.xml** file so that the style sheet or schema is not loaded.

None of the files introduced in this section (**combine\_audit\_files.pl**, **iMANAuditLog.xml**, **iMANAuditLog.xsl**, **iMANAuditLogSchema.xsd**, and **iMANAuditLog.js**) require a Teamcenter Engineering environment. You can run them anywhere as long as you have Perl and Microsoft Internet Explorer.

---

**define\_auditdefs**

---

**DESCRIPTION**

Creates **AuditDefinition** objects in the Teamcenter Engineering database. It scans the input file given by the argument **-f=***file-name*. 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 */
```

**SYNTAX**

**define\_auditdefs -u=user-id -p=password -g=group [-h] [-v] [-f=input-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.



If Teamcenter 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 Engineering database. If you do not supply these arguments, the utility attempts 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-name* 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.

**-f=input-file-name**

Text file that contains audit definition records.

**-v**

Specifies verbose mode.

**ENVIRONMENT**

As specified in [Manually Setting the Teamcenter Engineering Environment](#) in chapter 1, [Introduction](#).

**FILES**

As specified in *Log Files* in chapter 1, *Introduction*.

**RESTRICTIONS**

None.

**EXAMPLES**

None.

## **Backup and Recovery**

This section describes the utilities used to perform Teamcenter Engineering backup and recovery operations.

---

## backup\_modes

---

**DESCRIPTION**

Allows Teamcenter Engineering to operate in 24 x 7 mode. This utility administers hot backup so that Teamcenter Engineering need not be shut down for routine backup.

The **backup\_modes** utility sets TCFS (Teamcenter Engineering volumes) in different modes and sends appropriate messages to all current Teamcenter Engineering sessions. Third-party storage management systems can use this utility to obtain the hot backup of Oracle database and the file system.

**SYNTAX**

**backup\_modes**

**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.



If Teamcenter 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 Engineering database. If you do not supply these arguments, the utility attempts 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-name* 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.

**-f**

Specifies one of the following optional function arguments:

**move**

Moves contents of a blobby volume to the original volume location in problem situations.

**remove**

Clears the contents of the blobby volume contents in problem situations.

**opencnt**

Obtains information about the Teamcenter Engineering OS files opened for the **create** or **append** operation.

**-force**

Bypasses the checking for any OS files that are open for write, if set to **true**. This argument is optional.

**-h**

Displays help for the utility.

**-h**

Displays help for this utility.

**ENVIRONMENT**

The proper values must be set for the following variables:

- **IMAN\_set\_imanfsModes**
- **blobbyVolume\_NT**
- **blobbyVolume\_UNX**
- **IMAN\_check\_imanfsMode\_interval**

Restart all active TCFS processes before running this utility.

**FILES**

As specified in *Log Files* in chapter 1, *Introduction*.

**RESTRICTIONS**

Before setting TCFS in blobby volume mode, ensure the current mode is normal mode.

**EXAMPLES**

- To place Teamcenter Engineering in read-only mode, enter the following command:  

```
backup_modes -u=infodba -p=password -g=dba -m=rdbonly
```
- To place Teamcenter Engineering in read-only mode ignoring files open for write, enter the following command:  

```
backup_modes -u=infodba -p=password -g=dba -m=rdbonly -force=true
```
- To place Teamcenter Engineering in normal mode, enter the following command:  

```
backup_modes -u=infodba -p=password -g=dba -m=normal
```
- To place Teamcenter Engineering in caution message mode, enter the following command:  

```
backup_modes -u=infodba -p=password -g=dba -m=caution
```
- To place Teamcenter Engineering in blobbyvol mode, enter the following command:  

```
backup_modes -u=infodba -p=password -g=dba -m=blobby
```
- To obtain the current TCFS mode, enter the following command:  

```
backup_modes -u=infodba -p=password -g=dba -m=current
```
- To remove the blobby volume contents, enter the following command:  

```
backup_modes -u=infodba -p=password -g=dba -f=remove
```



- To move the blobby volume contents, enter the following command:

```
backup_modes -u=infodba -p=password -g=dba -f=move
```

- To obtain the sum total of files open for write, enter the following command:

```
backup_modes -u=infodba -p=password -g=dba -f=openent
```

---

**backup\_xmlinfo**

---

**DESCRIPTION**

Provides information about Teamcenter Engineering volumes defined for a site in XML format. Third-party backup systems require this information for 24x7 hot backup of Teamcenter Engineering 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 -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.



If Teamcenter 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 Engineering database. If you do not supply these arguments, the utility attempts 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-name* 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.

**-h**

Displays help for this utility.

**ENVIRONMENT**

As specified in *Manually Setting the Teamcenter Engineering Environment* in chapter 1, *Introduction*.

**FILES**

As specified in *Log Files* in chapter 1, *Introduction*.

**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>satishl_vol</VolumeName>
    <VolumeUid>037840d6b8ac558e9f42</VolumeUid>
    <NodeName>uslvw1097a011</NodeName>
    <WntPath>c:\satishl_vol</WntPath>
  </volumeinfo>
</backupInfo>
```

---

**object\_backup**

---

**DESCRIPTION**

Allows you to selectively back up objects from the database. This utility is the companion to the **object\_recover** utility and must not be confused with the **object\_archive** or **object\_restore** utilities.

**SYNTAX**

```
object_backup -u=user-id -p=password -g=group -media=media-name
-f=file-name -object=object-name -by_user=user-id -by_group=group
-by_type={Dataset | Item | Form | Folder} [-meta=file-name]
[-mail={ALL | NONE}] [-h]
```

**ARGUMENTS****-media**

Specifies the device used to store backed up objects. See [Restrictions](#), later in this section.

**-f**

Specifies the name of the input file that contains selection criteria defined by one or more of the **-object**, **-by\_user**, **-by\_group**, or **-by\_type** arguments.

**-object**

Specifies the object name (or ID for items or item revisions) to be backed up. See [Restrictions](#), later in this section.

**-by\_user**

Backs up all objects owned by the specified user. See [Restrictions](#), later in this section.

**-by\_group**

Backs up all objects owned by the specified group.

**-by\_type**

Backs up all objects of the specified type. Valid object types are dataset, item, form, and folder.

**-meta**

Backs up objects to a specified file. This argument is optional. If this argument is not supplied, data is backed up to the **backup.meta** file. If the specified file name already exists, an error is displayed.

**-mail**

Specifies which users receive notification, via Teamcenter mail, of the backup operation. If this argument is not supplied, only the user performing the backup receives notification.

If the **ALL** value is supplied, all owning users of the backed up objects receive notification. If the **NONE** value is specified, notification is not provided.

**ENVIRONMENT**

As specified in [Manually Setting the Teamcenter Engineering Environment](#) in chapter 1, [Introduction](#).

**FILES**

As specified in [Log Files](#) in chapter 1, [Introduction](#).

## RESTRICTIONS

- Storage media must be available (defined and on line) before using this utility.
- If the **-object** parameter contains spaces, it must be enclosed in quotes.
- Selection criteria can be entered from an input file or using a command-line argument. Multiple lines of selection criteria can be specified from an input file and all objects that meet the criteria are archived.
- If the **IMAN\_STAGING\_AREA** environment variable is defined, it must point to a valid directory or an error is displayed.
- The **object\_backup** utility can be run only by members of the **System** group.
- When a folder or item is backed up, all of the referenced objects are also backed up.

## EXAMPLES

- To back up **daily\_report** datasets owned by user **Smith**, enter the following command on a single line:

```
$IMAN_ROOT/bin/object_backup -u=infodba -p=password -g=dba
-object=daily_report -by_user=smithj -by_type=dataset
```

- To back up **daily\_report** datasets owned by users **smithj**, **parkerh**, and **meyersm**, first create an input file named **selections** containing the following information:

```
-object=daily_report -by_user=smithj -by_type=dataset
-object=daily_report -by_user=parkerh -by_type=dataset
-object=daily_report -by_user=meyersm -by_type=dataset
```

Next, run the utility by entering the following command on a single line:

```
$IMAN_ROOT/bin/object_backup -u=infodba -p=password -g=dba -f=selections
```

---

**object\_recover**

---

**DESCRIPTION**

Selectively recovers Teamcenter Engineering objects from storage media. The **object\_recover** utility must not be confused with the **object\_archive** and **object\_restore** utilities.

When the **object\_recover** utility runs, objects are recovered, placed in each object owner's **Newstuff** folder and a Teamcenter Engineering mail message is sent to the object owner that recovery was successfully completed.

**SYNTAX**

```
object_recover -u=user-id -p=password -g=group -media=device-name  
-f=file-name | -object=object-name -by_user=user-id -by_group=group-name  
-by_type={Dataset | Item | Form | Folder}-meta=file-name] [-l] [-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.



If Teamcenter 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 Engineering database. If you do not supply these arguments, the utility attempts 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-name* 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.

**-media**

Specifies the name of the device from which objects are restored. See restriction #1.

**-f**

Specifies the input file containing selection criteria defined using one or more of the following arguments:

**-object**

Restores the object by name or ID for items and item revisions.

**-by\_user**

Restores all objects owned by the specified user.

**-by\_group**

Restores all objects owned by the specified group.

**-by\_type**

Restores all objects of the specified type. Choices are **Dataset**, **Item**, **Form**, or **Folder**.

**-meta**

Restores objects from a specified file. This argument is only required if backup data was saved to a file other than the default **backup.meta** file.

**-l**

Lists all backed up objects in the meta file.

**-h**

Displays help for this utility.

**ENVIRONMENT**

As specified in [Manually Setting the Teamcenter Engineering Environment](#) in chapter 1, [Introduction](#) and the following preferences:

**IMAN\_STAGING\_AREA**

**RECOVER\_TO\_VOLUME**

**RECOVER\_TO\_UNRELEASED\_STATUS**

For more information, see the *Configuration Guide*.

**FILES**

As specified in [Log Files](#) in chapter 1, [Introduction](#).

**RESTRICTIONS**

- Selection criteria can be entered from an input file or via line argument. Multiple lines of selection criteria can be specified from an input file and all objects that meet the criteria are processed.
- If the **\$IMAN\_STAGING\_AREA** environment variable is not set, the metafile containing the backed-up data must be in the current working directory.
- The **object\_recover** utility only recovers objects that have been backed up using the **object\_backup** utility.
- Users must not modify or delete objects while they are being recovered. Failure to observe this restriction could cause lost or corrupted data. It is the responsibility of the system administrator to inform users that objects are being recovered and that they must not modify or delete any existing object until the recovery process is complete.

**EXAMPLES**

None.

---

**sfr\_instances**

---

**DESCRIPTION**

Creates and deletes single file recovery instances.

**SYNTAX**

**sfr\_instances** [-h] -u=*user-name* -p=*password* -g=*group-name*  
[-d=*datasettype*] [-ou=*owning-user*] [-og=*owning-group*] [-v=*volume-name*]  
[-ib=*any-previous-backupLabel*] -b=*new-backup-label* -f=[*function*]  
{**create** | **delete** | **list**}

**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.



If Teamcenter 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 Engineering database. If you do not supply these arguments, the utility attempts 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-name* 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.

**-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 Setting the Teamcenter Engineering Environment](#) in chapter 1, [Introduction](#). In case this is not already initialized, set the proper values for the following variables to enable further recovery:

**sfrVolume\_NT**

**sfrVolume\_UNX**

**IMAN\_sfr\_recovery\_interval**

**IMAN\_sfr\_process\_life\_time**

#### FILES

As specified in [Log Files](#) in chapter 1, [Introduction](#).

#### RESTRICTIONS

None, however it is a good practice to run this utility after putting Teamcenter Engineering 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
```

## **Engineering Translation Services**

This section describes the script files used to start and stop the ETS registry and ETS services, create the POM schema for translation request objects, and create translation requests from the command line. To fully process an ETS translation request there must be a service registry, one or more instances of the extract and load services, and a single instance of the scheduler service, all running as daemons. These commands can be combined into scripts as required.

---

## ETSDbConfig (.bat/sh)

---

### DESCRIPTION

Adds the ETSTranslationRequest schema to the installation database. This script uses the ETSUpdateSchema executable to create the POM schema for the ETS translation request object and subsequently runs the **install -regen\_schema**, **install -gen\_xmit\_file**, and **install -unlock\_db** commands to update client files.



This script is normally executed by the Teamcenter Environment Manager installation program.

### SYNTAX

**ETSDbConfig.bat infodba infodba**

### ENVIRONMENT

This script runs **iman\_profilevars** to set up a complete environment.

### FILES

None.

### RESTRICTIONS

The **ETSDbConfig** script must be run when there are no users logged in to the database.

### EXAMPLES

To update the installation schema for the **ETSTranslationRequest**, run the script as follows:

Windows

```
ETSDbConfig.bat -u=user-name -p=password
```

UNIX

```
ETSDbConfig.sh -u=user-name -p=password
```



The script must be run by a user with **DBA** privileges.

---

**etsServiceRegistry**

---

**DESCRIPTION**

Activates the ETS Service Registry to perform registry-related actions.

**SYNTAX**

**\$ETS\_HOME/bin/etsServiceRegistry -action=start | stop | createpwf | ping**

**ARGUMENTS****-u**

Specifies the user ID.



This argument is used only with the **createpwf** action.

This is generally **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 password.



This argument is used only with the **createpwf** action.

If used without a value, the system assumes a null value.

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

**-action**

Specifies one of the following actions:

**start**       Starts the ETS registry.

**stop**       Stops the service registry.



Ensure that all ETS services are stopped before attempting to stop the ETS registry.

**ping**       Displays a status message from the ETS registry. Use this option to determine if the service is running.

**createpwf** Creates an ETS proxy password file to avoid interactive login when starting ETS services.

**-m**

Defines the Teamcenter Engineering database marker specified in the **ETS Service.properties** file, which is defined in the **PORTAL\_ROOT client\_specific.properties** file.



This argument is used only with the **createpwf** action.

**ENVIRONMENT**

For Windows, the **ETS\_HOME** and **IPR** variables must be set. If they are not set, the script sets them as specified during installation.

For Windows and UNIX, the script is edited during installation to set the appropriate environment variables.

**FILES**

None.

**RESTRICTIONS**

This script must be run by the user with **DBA** privileges who owns the ETS installation.

**EXAMPLES**

- To start the ETS registry, enter the following command on a single line:

```
etsServiceRegistry -action=start
```

- To stop the ETS registry, enter the following command on a single line:

```
etsServiceRegistry -action=stop
```

- To verify whether the ETS registry is running, enter the following command on a single line:

```
etsServiceRegistry -action=ping
```

- To create the ETS proxy user password file, enter the following command on a single line:

```
etsServiceRegistry -action=createpwf -m=installation-database-marker  
-u=ETS-proxy-user-id -p=password
```

---

**etsServiceManager (.bat/.sh)**

---

**DESCRIPTION**

Activates the ETS Service Manager to perform service related actions.

**SYNTAX**

**\$ETS\_HOME/bin/etsServiceManager -action=[start | stop | poke | ping]**  
**-service=Extractor | Scheduler | Loader [-instance=1 | 2 | ...]**

**ARGUMENTS****-action**

Specifies one of the following actions:

**start**        Instantiates the specified service.



Before starting an ETS service daemon, ensure that the ETS service registry daemon is running. Only one instance of a scheduler daemon should be run.

**stop**        Stops the specified service.

**poke**        Causes the current request polling interval to expire for the specified service daemon, resulting in an immediate query for translation requests to process.

**ping**        Contacts the service and displays a status message from the specified service instance. Use this option to determine if the service is running.

**-service**

Specifies one of the following services to be run:

**Extractor**

**Scheduler**

**Loader**

**-instance**

Identifies a particular service instance. For example, **Extractor.inst1**

**ENVIRONMENT**

For Windows, the **ETS\_HOME** and **IPR** variables must be set. If they are not set, the script sets them as specified during installation.

For Windows and UNIX, the script is edited during installation to set the appropriate environment variables.

**FILES**

None.

**RESTRICTIONS**

This script must be run by the user with **DBA** privileges who owns the ETS installation.

**EXAMPLES**

- To start the **Scheduler** service, enter the following command on a single line:

```
etsServiceManager -action=start -service=Scheduler
```

- To stop the **Scheduler** service, enter the following command on a single line:

```
etsServiceManager -action=stop -service=Scheduler
```

- To verify whether the **Scheduler** service is running, enter the following command on a single line:

```
etsServiceManager -action=ping -service=Scheduler
```

- To force the **Scheduler** service polling interval to expire, enter the following command on a single line:

```
etsServiceManager -action=poke -service=Scheduler
```

**Startup and Shutdown Sequence**

A typical startup and shutdown sequence is illustrated below:

*Startup sequence*

```
echo starting ETS Registry
start "ETS Registry" /DIMAN_ROOT\bin cmd /C
"etsServiceRegistry -action=start && if errorlevel 1 pause"

sleep 10

echo starting ETS Extractor
start "ETS Extractor" /DIMAN_ROOT\bin cmd /C
"etsServiceManager -action=start -service=Extractor && if
errorlevel 1 pause"

echo starting ETS Scheduler

start "ETS Scheduler" /DIMAN_ROOT\bin cmd /C
"etsServiceManager -action=start -service=Scheduler && if
errorlevel 1 pause"

echo starting ETS Loader

start "ETS Loader" /DIMAN_ROOT\bin cmd /C
"etsServiceManager -action=start -service=Loader &&
errorlevel 1 pause"
```

*Shutdown sequence*

```
echo stopping ETS Extractor

call IMAN_ROOT\bin\etsServiceManager -action=stop -service=Extractor

echo stopping ETS Scheduler

call IMAN_ROOT\bin\etsServiceManager -action=stop -service=Scheduler

echo stopping ETS Loader

call IMAN_ROOT\bin\etsServiceManager -action=stop -service=Loader

echo Waiting for services to stop

sleep 10

echo stopping ETS Registry

call IMAN_ROOT\bin\etsServiceRegistry -action=stop
```



---

**ets\_create\_rqst**


---

**DESCRIPTION**

Provides the ability to create a translation request using command line arguments.

**SYNTAX**

*ETS\_HOME***support/sample/utilities/ets\_create\_rqst**

**ARGUMENTS**

The **-u**, **-p**, and **-g** arguments are used to automatically log in to a Teamcenter server session to create the ETSTranslationRequest object.

- u** Specifies the user ID of the user who will own the translation request.
- p** Specifies the password corresponding with the user ID.
- g** Specifies the group to which the user specified by the **-u** argument belongs.
- 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.
- dn** Specifies the dataset name. This argument is optional.
- dv** Specifies the dataset version. If no version number is specified, the latest version is used. This argument is optional.
- 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 UGS.
- tn** Specifies the name of the translator, for example **ideastojt**.
- tr** Specifies the trigger name, for example **COMMANDLINE**.
- verbose** Provides additional information. This argument is optional.
- f** Specifies an input file used to create one or more translation 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]
```



Commas are required.

**ENVIRONMENT**

As specified in *Manually Setting the Teamcenter Engineering Environment* in chapter 1, *Introduction*.

**FILES**

As specified in *Log Files* in chapter 1, *Introduction*.

**RESTRICTIONS**

None.

**EXAMPLES**

- To create a translation 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:

```
ets_create_rqst -i=Block -r=A -dt=IdeasPart -pr=2 -pn=UGS -tn=ideastojt  
-tr=COMMAND_LINE -u=infodba -p=infodba -g=dba
```

- To create a translation request for version 2 of the **Block/A** item revision, enter the following command on a single line:

```
ets_create_rqst -i=Block -r=A -dv=2 -dt=IdeasPart -pr=2 -pn=UGS  
-tn=ideastojt -tr=COMMAND_LINE -u=infodba -p=infodba -g=dba
```

- To create a translation request for two parts using an input file, enter the following command on a single line:

```
ets_create_request ...
```

The lines in the input file are:

```
Block,A,,,  
Cone,A,,,2
```



All parts in the file are subject to the same translation, because the translator is specified on the command line.

## **Migration**

This section describes utilities used to migrate Teamcenter Engineering data.

---

**convert\_distribution\_lists**

---

**DESCRIPTION**

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 [-h] [-all] [-delete]
[-dist_list_name=distribution-list-name] [-import_file=file-name]
[-import=file-name] [-new_list_name=alias-list-name] [-overwrite]
```

**ARGUMENTS****-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.

**ENVIRONMENT**

As specified in *Manually Setting the Teamcenter Engineering Environment* in chapter 1, *Introduction*.

**FILES**

As specified in *Log Files* in chapter 1, *Introduction*.

**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
```

---

**migrate\_type\_display\_prefs**

---

**DESCRIPTION**

In the Teamcenter Engineering 8.0 thin client interface, only those form type values listed in the **FormTypePref** preference are displayed. This utility considers the preference values stored at the site level and creates type display rules so that only the types listed under the site preference are displayed in the object creation dialog windows. Then the utility traverses the group hierarchy and considers the preference values in the group preference files and creates type display rules.

In the NX Manager implementation, the display of part types is controlled by the **IMAN\_part\_types\_display\_filter** preference. The type values assigned to this preference are not shown in the Part Creation dialog window. The utility considers the preference values stored at the site level and creates type display rules so that the types listed under the site preference are not displayed in the object creation dialog windows. Then the utility traverses the group hierarchy and considers the preference values under the group preference files and creates type display rules.

**SYNTAX**

```
migrate_type_display_prefs [-h] -u=user-id -p=password  
-g=dba [-option=FormTypesPref | DatasetTypesPref |  
IMAN_part_types_display_pref]
```

**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.



If Teamcenter 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 Engineering database. If you do not supply these arguments, the utility attempts 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-name* 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.

**-option**

This argument accepts the following values:

**FormTypesPref**

Creates type display rules for each value in the **FormTypesPref** preference.

**DatasetTypesPref**

Creates type display rules for each value in the **DatasetTypesPref** preference.

**IMAN\_part\_types\_display\_pref**

Creates type display rules for each entry in the **IMANpart\_types\_display\_pref** preference.

If the **-option** argument is not specified, the utility creates type display rules for all of the entries found for all three preferences.

**RESTRICTIONS**

This utility must be run by an administrator.

**EXAMPLES**

- Enter the following command to migrate the data in the **IMAN\_part\_types\_display\_filter** preference to the new type display rules functionality:  

```
migrate_type_display_prefs -u=user -p=password -g=dba
-option=IMAN_part_types_display_pref
```
- Enter the following command to migrate the data in the **FormTypesPref** preference to the type display rules functionality:  

```
migrate_type_display_prefs -u=user-id -p=password -g=dba
-option=FormTypesPref
```
- Enter the following command to migrate the data in the **FormTypesPref**, **DatasetTypesPref**, and **IMAN\_types\_display\_filter** to the type display rules functionality:  

```
migrate_type_display_prefs -u=user-name -p=password -g=dba
```

---

**move\_mso\_forms**

---

**DESCRIPTION**

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 in to a 10.0 database. These forms are used for property synchronization between Microsoft Office and the Teamcenter database.



This utility is called by the upgrade script when upgrading from Teamcenter 2005 to Teamcenter 2005 SR1.

**SYNTAX**

**move\_mso\_forms** [-u=*user-name* -p=*password* -g=*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.



If Teamcenter 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 Engineering database. If you do not supply these arguments, the utility attempts 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-name* 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.

**-h**

Displays help for this utility.

**ENVIRONMENT**

As specified in [Manually Setting the Teamcenter Engineering Environment](#) in chapter 1, [Introduction](#) .

**FILES**

As specified in [Log Files](#) in chapter 1, [Introduction](#).

**RESTRICTIONS**

None.



## **Project**

The utilities used to create projects and update project data are described in this section.

---

**create\_project**

---

**DESCRIPTION**

Creates projects in the database based on command line input or input from a text file.

**SYNTAX**

```
create_project -u=user-id -p=password -g=group-name
{-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... ]}
{-input=full-path-to-input-file [-delimiter=delimiter-character]}
```

**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.



If Teamcenter 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 Engineering database. If you do not supply these arguments, the utility attempts 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-name* 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.

**-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:

```
group1~role1~user1~group2~role2~user2...
```

In addition, you can specify all members in a group as follows:

```
group1~**~*
```

**-privileged**

Defines privileged group members using the following format:

```
group1~role1~user1...
```

**-input**

Specifies the path to a text file containing multiple entries of project id, project name, project description, teams, and privileged members. 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
```

**-delimiter**

Specifies the delimiting character used in the input file to parse id, name, description, status, and teams.

**ENVIRONMENT**

As specified in [Manually Setting the Teamcenter Engineering Environment](#) in chapter 1, [Introduction](#).

**FILES**

Input file must conform to the syntax described in the **-input** argument description in the Arguments section, earlier in this chapter.

**RESTRICTIONS**

None.

**RETURN  
VALUES**

**Return value** 0  
**upon success**

**Return value** >1  
**upon failure**

**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\_bom

### DESCRIPTION

Allows you to update all items in a BOM structure within specified projects.

### SYNTAX

```
update_project_bom -u=user-id -p=password -g=group [-f={add | remove}]  
[-type={item | rev}] -item=item-id [-rev_id=revision-id] [-rev_rule=revision-rule]  
[-unit_no=unit-number] [-date=date] [-end_item=end-item-id]  
[-var_rule=variant-rule] -projects=project-lists [-level=level-number] [-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.



If Teamcenter 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 Engineering database. If you do not supply these arguments, the utility attempts 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-name* 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.

#### **-f**

Specifies one of the following types of operation for this utility:

**=add** Adds item objects to projects.

**=remove** Removes item objects from projects.



If this argument is omitted, the default action is to add items to 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.



If this argument is omitted, the default type is **item**.

**-item**

Specifies the ID of the root item of the BOM to update.

**-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.

**-var\_rule**

Specifies the variant rule to be applied to the BOM structure. If omitted, no variant rule is applied.

**-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. More than one project can be specified. If there is more than one project in the list, each project name is separated by a comma (,).

**-level**

Specifies to what depth the BOM tree is traversed. The default value is **all**, which indicates the entire BOM structure is traversed.

**-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 -level=3
```

---

**update\_project\_data**

---

**DESCRIPTION**

Updates project data in the Teamcenter Engineering database.



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 [-h] -u=user-name -p=password -g=group-name  
[-f=function] [-force] [-t=relation-type-name1[,relation-type-name2...]]  
[-pid=project-ID1[, project-ID2...]]
```

**ARGUMENTS**

**-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 Setting the Teamcenter Engineering Environment* in chapter 1, *Introduction*.

**FILES**

As specified in *Log Files* in chapter 1, *Introduction*.

**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
```



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
```



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...]
```



Processing time can be considerable depending on the number of objects in each project.

## **Subscription Manager**

This section describes the utility used to delete invalid and expired subscriptions.

---

**purge\_invalid\_subscriptions**

---

**DESCRIPTION**

Provides the capability to delete those 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****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.



If Teamcenter 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 Engineering database. If you do not supply these arguments, the utility attempts 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-name* 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.

**-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 Setting the Teamcenter Engineering Environment* in chapter 1, *Introduction*.

**FILES**

As specified in *Log Files* in chapter 1, *Introduction*.

**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**

This section describes the utilities used to maintain your Teamcenter Engineering installation.

---

## clearlocks

---

### DESCRIPTION

Clears dead process locks from the database. Dead process locks typically occur when a Teamcenter Engineering session terminates abnormally. Process locks are set on an object when it is being modified or deleted. If a Teamcenter Engineering 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 Engineering 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.



To use the **-assert\_dead** or **-assert\_all\_dead** options, you must specify the administrator's user name, password, and group.

### SYNTAX

```
clearlocks [-verbose] [-one_pass] [-retry time] [-node_names] [-assert_dead
-u=user-name -p=password -g=group-name node-name | -assert_all_dead
-u=user-name -p=password -g=group-name] [-h]
```



The **clearlocks** utility can be run with active Teamcenter Engineering 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

#### **-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.

#### **-one\_pass**

Executes the utility once and stops. This is the default if no other arguments are supplied. Opposite of the **-retry** argument.

#### **-retry**

Continuously executes the utility according to the time, specified in seconds, before the next execution. Opposite of **-one\_pass** argument.

**-assert\_dead**

Asserts that all processes on a particular node are dead and clears all process locks held by sessions running on that node. If any of those sessions are alive and in use, the locks held by those sessions will be compromised.



To use this argument, you must enter the node name and the administrator's user name, password, and group.

**-assert\_all\_dead**

Asserts that all processes in the database are dead and clears all process 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 and reports on the sessions that were asserted to be dead.



To use this argument, you must enter the administrator's user name, password, and group.

**-h**

Displays help for this utility.

**ENVIRONMENT**

As specified in [Manually Setting the Teamcenter Engineering Environment](#) in chapter 1, [Introduction](#).

**FILES**

As specified in [Log Files](#) in chapter 1, [Introduction](#).

**RESTRICTIONS**

- Do not run the **clearlocks** utility with the **-assert\_dead** or **-assert\_all\_dead** arguments if there are any active Teamcenter Engineering 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.



## EXAMPLES

- To clear process locks for dead sessions, enter the following command from that node:

```
$IMAN_ROOT/bin/clearlocks
```

- To obtain a list of all network nodes which have process locks set on the database, enter the following command:

```
$IMAN_ROOT/bin/clearlocks -node_names
```

- To run the **clearlocks** utility every four hours, enter the following command:

```
$IMAN_ROOT/bin/clearlocks -retry 14400
```

- To clear all process locks (active and dead) on a single network node, in this example **ntssun9**, enter the following command:

```
$IMAN_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:

```
$IMAN_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 Engineering and NX Manager 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:

```
$IMAN_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:

```
$IMAN_ROOT/bin/clearlocks
```

4. Create a report of all remaining process locks by entering the following command:

```
$IMAN_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:

```
$IMAN_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:

```
$IMAN_ROOT/bin/clearlocks -node_names
```

This report should be clean (empty). If there are any nodes listed in this report, contact the UGS Global Technical Access Center (GTAC) for assistance.

---

**imanhelp**

---

**DESCRIPTION**

Accesses verbose information about Teamcenter Engineering error messages.

**SYNTAX**

**imanhelp** *error-message-number*

**ARGUMENTS**

This utility accepts only one value: *error-message-number*. This is the integer error code displayed in the Error Message dialog window.

**ENVIRONMENT**

As specified in [Manually Setting the Teamcenter Engineering Environment](#) in chapter 1, [Introduction](#).

**FILES**

As specified in [Log Files](#) in chapter 1, [Introduction](#).

**RESTRICTIONS**

None.

**EXAMPLES**

None.

---

**tc\_mail\_smtp**

---

**DESCRIPTION**

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 -to=address {-to=address}* [{-cc=address}*]  
[-subject=subject-enclosed-in-double-quotes] [-server=mail-server-name]  
[-body=body-file-name] [-attachments=file-name-consisting-of-attachments-list]  
[-user=sender's-name]
```

**ARGUMENTS****-to**

Specifies the E-mail address of the recipient. If there are multiple recipients, use a **-to** argument for each recipient. This 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. This argument is optional.

**-subject**

Specifies the subject line of the E-mail. The subject line is enclosed in double quotes. This argument is optional.

**-server**

Specifies the name of the mail server. If not defined, the SMTP mail port (25) of the local machine is used. This argument is optional.

**-body**

Specifies the file name of the E-mail body. This argument is optional.

**-attachments**

Specifies the format of the attachment list. Each line of the list requires the attachment file name and the format (**B**=binary text, **T**=text). This argument is optional.

**-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. This argument is optional.

**ENVIRONMENT**

As specified in *Manually Setting the Teamcenter Engineering Environment* in chapter 1, *Introduction*.

**FILES**

As specified in *Log Files* in chapter 1, *Introduction*.

**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  
-cc=person4@company4.com -server=mail_server_1  
-body=C:\temp\test.txt -attachments=C:\temp\attachment_list.txt  
-user=person5
```

---

**install\_event\_types**

---

**DESCRIPTION**

Defines which event and object types can be subscribed to and/or audited. This utility also creates new event types and adds them as valid event types to an object type.

**SYNTAX**

```
install_event_types [-h] -u=user-name -p=password -g=group -f=function
{install | create | add | remove | modify | listValidEvents |
listEventtypes | text-file-name} [ -overwrite ] (For -f=install and
-f=input-file only) [-eventtype=eventTypeId] [ -imantype=imanTypeName]
[ -imanclass=imanClassName] [-remove] [ -audit ] [-noaudit ]
[ -noaudit ] [-nosubscribe]
```

**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.



If Teamcenter 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 Engineering database. If you do not supply these arguments, the utility attempts 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-name* 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.

**-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.

- **create**

Creates the specified event type.

- **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 types, 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.

**-subscribe**

Specifies that the event type can be subscribed to. Valid only with the **-f=modify** argument.

**-noaudit**

Specifies that the event type cannot be audited. Valid only with the **-f=modify** argument.

**-nosubscribe**

Specifies that the event type cannot be subscribed to. Valid only with the **-f=modify** argument.

**ENVIRONMENT**

As specified in [Manually Setting the Teamcenter Engineering Environment](#) in chapter 1, [Introduction](#).

**FILES**

As specified in [Log Files](#) in chapter 1, [Introduction](#).

**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 create the **MyEventType** event type, enter the following command on a single line:

```
install_event_types -f=create -eventtype=MyEventType
```

- To add the **MyEventType** event type as a valid event type for the Teamcenter Engineering type **Item**, enter the following command on a single line:

```
install_event_types -f=add -imantype=Item -imanclass=Item  
-eventtype=MyEventType
```

- To remove the **MyEventType** event type from the Teamcenter Engineering type **Item**, enter the following command on a single line:

```
install_event_types -f=remove -imantype=Item -imanclass=Item  
-eventtype=MyEventType
```



Users cannot remove Teamcenter Engineering internal event types.



- To set the **MyEventType** event type as subscribable but not auditable for the Teamcenter Engineering type **Item**, enter the following command on a single line:

```
install_event_types -f=modify -imantype=Item
                    -imanclass=Item -eventtype=MyEventType -audit -nosubscribe
```

- To list all valid event types for the Teamcenter Engineering type **Item**, enter the following command on a single line:

```
install_event_types -f=listValidEvents -imantype=Item -imanclass=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
```

---

**list\_users**

---

**DESCRIPTION**

Creates a list of users currently logged in to Teamcenter Engineering and the node they are using. This information is useful if database maintenance is necessary and all users currently logged in must be notified.

**SYNTAX**

**list\_users** **-u**=*user-id* **-p**=*password* **-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.



If Teamcenter 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 Engineering database. If you do not supply these arguments, the utility attempts 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-name* 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.

**ENVIRONMENT**

As specified in [Manually Setting the Teamcenter Engineering Environment](#) in chapter 1, [Introduction](#).

**FILES**

As specified in [Log Files](#) in chapter 1, [Introduction](#).

**RESTRICTIONS**

None.

**EXAMPLES**

None.

---

## purge\_file\_cache

---

**DESCRIPTION**

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 Engineering volume.

**SYNTAX**

**purge\_file\_cache**

**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.



If Teamcenter 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 Engineering database. If you do not supply these arguments, the utility attempts 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-name* 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.

**-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 Engineering volume are purged.

**-h**

Displays help for this utility.

**ENVIRONMENT**

As specified in [Manually Setting the Teamcenter Engineering Environment](#) in chapter 1, [Introduction](#).

**FILES**

As specified in [Log Files](#) in chapter 1, [Introduction](#).

**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 Engineering volume are purged from the cache.

```
purge_file_cache -u=smith -p=password -g=design
```

---

## reset\_user\_home\_folder

---

**DESCRIPTION**

Repairs corruption that may occur when deleting a user from the database that redirects the user's home folder, mailbox folder, and new stuff folder to the deleting user's.

If a home folder owned by the user is found, the utility points the user's home folder back to its own home folder. If the user's home folder is not found, the utility creates a new home folder, and points the user's home folder back to it.

**SYNTAX**

```
%IMAN_BIN%\reset_user_home_folder -u=user-id -p=password -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.



If Teamcenter 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 Engineering database. If you do not supply these arguments, the utility attempts 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-name* 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.

**-id**

Specifies the user id of the user who owns the home folder to be reset.

**ENVIRONMENT**

As specified in [Manually Setting the Teamcenter Engineering Environment](#) in chapter 1, [Introduction](#).

**FILES**

As specified in [Log Files](#) in chapter 1, [Introduction](#).

**RESTRICTIONS**

This utility must be run by a system administrator.

**EXAMPLES**

None.



---

## Chapter

# *11 NX Manager Utilities*

export_attr_mappings .....	11-2
import_attr_mappings .....	11-3
refile_info .....	11-4
reset_protection .....	11-5
ugmgr_upgrade_bvrsyncform .....	11-8
ugmgr_upgrade_transforms .....	11-10





---

## Chapter

# *11 NX Manager Utilities*

---

This chapter describes the utilities used to import and export attribute mappings and assemblies in to and out of the Teamcenter Engineering database, update files to the latest version, transfer files between Teamcenter Engineering databases, upgrade transforms, upgrade assemblies to use the **BVRSyncInfo** form, and synchronize attribute protection attributes for **UGPART** and **UGMASTER** datasets.

---

---

**export\_attr\_mappings**

---

**DESCRIPTION**

Exports attribute mappings from the Teamcenter Engineering database to a text file.

**SYNTAX**

```
export_attr_mappings \ -f=text-file \ [-test] \ [-u=user-name]
\ [-p=password] \ [-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.



If Teamcenter 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 Engineering database. If you do not supply these arguments, the utility attempts 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-name* 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.

**-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 Engineering shell environment.

**FILES**

As specified in [Log Files](#) in chapter 1, [Introduction](#).

**RESTRICTIONS**

None.

**EXAMPLES**

None.

---

## import\_attr\_mappings

---

**DESCRIPTION**

Imports attribute mappings into the Teamcenter Engineering database. UGS 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-name] \ [-p=password] \ [-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.



If Teamcenter 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 Engineering database. If you do not supply these arguments, the utility attempts 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-name* 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.

**-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 must be run in the iMAN shell environment.

**FILES**

As specified in [Log Files](#) in chapter 1, [Introduction](#).

**RESTRICTIONS**

None.

---

**refile\_info**

---

**DESCRIPTION**

Generates a list of identifiers of every item revision in the database. 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 -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.



If Teamcenter 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 Engineering database. If you do not supply these arguments, the utility attempts 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-name* 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.

**-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 Setting the Teamcenter Engineering Environment](#) in chapter 1, [Introduction](#).

**FILES**

As specified in [Log Files](#) in chapter 1, [Introduction](#).

**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:

```
$IMAN_ROOT/bin/refile_info -u=infodba -p=password -g=dba
-o=item_revision_list
```

## reset\_protection

### DESCRIPTION

Synchronizes protection attributes for **UGPART** and **UGMASTER** datasets. The utility searches for all **UGPART** and **UGMASTER** datasets in the database and synchronizes the revision 0 dataset with **UGATTR**. It provides several modes: **query**, **check**, **list**, **confirm**, and **make**. The default mode is **check**, and the default search criteria is the **whole database**. However, users can expand or limit the search by entering other search criteria.

This utility supports Teamcenter Engineering 3.3 (and earlier) databases. It should not be necessary to run this against newer databases. However, you can use it as a diagnostic tool on all databases.

### SYNTAX

```
reset_protection -u=user-id -p=password -g=group -mode={query | check  
| list | confirm | make} [-file=file-name | -uid=string [-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.



If Teamcenter 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 Engineering database. If you do not supply these arguments, the utility attempts 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-name* 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.

#### **-mode**

Specifies the mode for the utility session, as follows:

##### **query**

Queries the database for all revision 0 datasets and generates the **dataset.query** dataset tag summary file.

##### **check**

Counts the number of dataset mismatches. If a dataset tag summary file is not supplied using the **-file** argument, the entire database is checked.

**list**

Lists all datasets in the file that don't match and generates the **dataset.list** mismatched datasets tag file.

**confirm**

Sequentially lists dataset mismatches in the file and asks for confirmation before changing.

**make**

Makes all datasets protections in sync.

**-file**

Specifies the dataset tag summary file (typically, **dataset.query**) used as input in **check**, **list**, **confirm** or **make** modes.

**-uid**

Specifies the universal identifier (UID) of a single dataset. Used instead of the **-file** argument in **check**, **list**, **confirm** or **make** modes.

**-h**

Displays help for this utility.

**ENVIRONMENT**

As specified in [Manually Setting the Teamcenter Engineering Environment](#) in chapter 1, [Introduction](#).

**FILES**

As specified in [Log Files](#) in chapter 1, [Introduction](#).

**RESTRICTIONS**

None.

## EXAMPLES

- To search the entire database for all mismatched dataset names, enter the following command on a single line:

```
$IMAN_ROOT/bin/reset_protection -u=infodba -p=password -g=dba -mode=list
```

The **dataset.query** dataset summary file is created.

- To fix a large number of datasets, perform the following steps:
  1. Run the utility in **query** mode to extract tags for all the revision 0 datasets in the database by entering the following command on a single line:

```
$IMAN_ROOT/bin/reset_protection -u=infodba -p=password  
-g=dba -mode=query
```

The **dataset.query** dataset summary file is created. This file contains tags for all revision 0 datasets in the database.

2. Split the **dataset.query** file into reasonable subsets (for example, 1000 tags per file) to prevent the utility from running out of memory when running in **check**, **list** or **make** modes. To split the file into chunks of 1000 tags at a time, enter the following command:

```
split -l 1000 dataset.query dataset.query_
```

The **dataset.query** file is split into smaller files, each containing 1000 tags in each file. Typical file names are **dataset.query\_aa**, **dataset.query\_bb** and so forth.

3. Run the utility in **check**, **list**, **confirm** or **make** mode using these files as input with the **-file** argument.

---

**ugmgr\_upgrade\_bvrsyncform**

---

**DESCRIPTION**

Upgrades assemblies created in a Teamcenter Engineering version earlier than 8.0 to use the **BVRSyncInfo Form**.

**SYNTAX**

```
ugmgr_upgrade_bvrsyncform [-h] [-u=user-id] [-p=password] [-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]
```

**ARGUMENTS****-h**

Displays help for the utility.

**-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.

**-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.

**-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 (used in conjunction with **-item**).

**-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**.

**ENVIRONMENT**

Requires the standard Teamcenter Engineering environment for running Integration Toolkit programs.

**FILES**

Standard Teamcenter Engineering system log files as specified in *System Log Files* in chapter 1, *Introduction*, in the Utilities Reference manual.

**RESTRICTIONS**

This utility is supported for Hewlett-Packard HP-UX, Sun Solaris, and Microsoft Windows 2000 systems only.

**EXAMPLES**

- The following example displays the usage message:

```
ugmgr_upgrade_bvrsyncform -h
```

- The following example upgrades the **UGMASTER** dataset attached under the **parent A** item with the new **BVRSYNCINFO** named reference:

```
ugmgr_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:

```
ugmgr_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:

```
ugmgr_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).

```
ugmgr_upgrade_bvrsyncform -i=list.txt
```

---

**ugmgr\_upgrade\_transforms**

---

**DESCRIPTION**

Upgrades transforms.

**SYNTAX**

```
ugmgr_upgrade_transforms [-h] [-u=user-id] [-p=password] [-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] [-force_units=measurement-unit]
```

**ARGUMENTS****-h**

Displays help for the utility.

**-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.

**-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.

**-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 (used in conjunction with **-item**).

**-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**.

**-force\_units**

Specifies measurement unit of revisions being upgraded. Valid values are **inches** and **millimeters**.



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.

**ENVIRONMENT**

Requires the standard Teamcenter Engineering environment for running Integration Toolkit programs.

**FILES**

Standard Teamcenter Engineering system log files as specified in *System Log Files* in chapter 1, *Introduction*, in the Utilities Reference manual.

**RESTRICTIONS**

This utility is supported for Hewlett-Packard HP-UX, Sun Solaris, and Microsoft Windows 2000 systems only.

**EXAMPLES**

- The following example displays the usage message:

```
ugmgr_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.

```
ugmgr_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:

```
ugmgr_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:

```
ugmgr_upgrade_transforms -item=top -bypass=yes -force_units=inches
```



---

Chapter

# *12 Integration Utilities*

Teamcenter Engineering/Teamcenter Community Integration . . . . .	12-1
rdv_context_download . . . . .	12-2
tcc_context_upload . . . . .	12-7
Teamcenter Engineering/Teamcenter Requirements Integration . . . . .	12-12
proxy_sync . . . . .	12-13



---

## Chapter

# *12 Integration Utilities*

---

This chapter describes utilities related to integrations between Teamcenter Engineering, other Teamcenter products, and third-party software applications.

---

### **Teamcenter Engineering/Teamcenter Community Integration**

This section describes utilities used to facilitate the integration between Teamcenter Engineering and Teamcenter Community.

---

**rdv\_context\_download**

---

**DESCRIPTION**

Runs against a workspace object in an RDV setup, executes a search and downloads the resulting data in to a flat file (AJT/PLM XML) in a specified directory.

**SYNTAX**

```
rdv_context_download [-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]
[-operator=BOX(Within | Outside | Interferes) | Plane(Above
| Below | Intersects)] [-output_format=BOM_writer
FormatAJT | PLMXML] [-file_name=file-name-without-extension]
[-absolute_path=path-of-file-to-be-stored-without-extension]
```

**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.



If Teamcenter 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 Engineering database. If you do not supply these arguments, the utility attempts 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-name* 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.

**-sco\_name**

Specifies the name of the structure context object.

**-item\_id**

Specifies the item ID.

**-rev\_id**

Specifies the revision ID.



**-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 Engineering 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



Search results are affected by the following Teamcenter Engineering preferences:

- **IMAN\_config\_rule\_name**
- **WebDesignContextDefaultSearchDistance**
- **PortalDesignContextMaxMatchingObjects**
- **PortalDesignContextMaxMatchingBOMLines**

For more information, see the *Configuration Guide*.

**-variant\_rule\_name**

Specifies the name of the variant rule.

**-revision\_rule\_name**

Specifies the revision rule, which defaults according to the **IMAN\_config\_rule\_name** preference.

**-h**

Displays help for this utility.

**-operator**

Specifies the zone type operator. Valid values for the **BOX** zone type are **Within**, **Outside**, or **Interferes**, and the default value is **Within**. Valid values for the **PLANE** zone type are **Above**, **Below**, and **Intersects**, and the default value is **Intersects**. The **-zone\_name** and **-zone\_type** arguments must be specified in conjunction with the **-operator** argument.



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.

**-file\_name**

Specifies the name of the AJT or PLM XML file to which the data is output. The default file name is the name of the input workspace object with **BOM\_writer\_format** extension (provide file name without 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.



The **absolute\_path** must be specified without an extension. For example, **/users/x\_user/tempfile** for UNIX or **c:\temp\tempfile** for Windows.

**-output\_format**

Specifies the BOM writer format, either PLM XML or AJT. The default format is PLM XML.

**ENVIRONMENT**

As specified in *Manually Setting the Teamcenter Engineering Environment*, in chapter 1, *Introduction*, and the **RDV\_debug** environment variable. If this variable is set, the Teamcenter Engineering **syslog** file contains additional debugging information.

**FILES**

As specified in *Log Files* in chapter 1, *Introduction*.

**EXAMPLES**

- The following example creates a PLM XML file called **TL109375** and uses the **IMAN\_config\_rule\_name** preference to determine the revision rule:

```
$IMAN_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:

```
$IMAN_ROOT/bin/rdv_context_download -item_id=TL109375 -rev_id=004  
-r="Beta or less w/pdi"
```

- The following example logs in to Teamcenter Engineering as the **infodba** user and creates a PLM XML file named **11\_21\_sco1\_a**:

```
$IMAN_ROOT/bin/rdv_context_download -sco_name=11_21_sco1_a -u=infodba  
-p=infodba -g=dba
```



- The following example prompts the user for auto login and creates an ASCII JT assembly file named **assy\_ajt\_file**:

```
$IMAN_ROOT/bin/rdv_context_download -folder=assy_folder
-filename=assy_ajt_file -bom_writer_format=AJT
```

Figure 12-2 shows the sample AJT file created in this example.

```

#####
# DirectModel ASCII file - version 1.0
# Written by BOMWriterFormatAJT Teamcenter Engineering P9.0.0.11
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_o40Cyn6c4PB"
  ATTR Type="STRING" Key="DB_OCC_UID" Value="AAAAAAAAAAAAAA"
  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="QX9o40Cyn6c4PB"
    ATTR Type="STRING" Key="DB_OCC_UID" Value="0z$o40Cyn6c4PB"
    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
#####

```

### Figure 12-2. Sample AJT File

## tcc\_context\_upload

### DESCRIPTION

Uploads fully or partially configured assemblies to Teamcenter Community after downloading them from Teamcenter Engineering. 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 [-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.



If Teamcenter 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 Engineering database. If you do not supply these arguments, the utility attempts 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-name* 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.

**-sco\_name**

Specifies the name of the structure context object.

**-item\_id**

Specifies the item ID.

**-rev\_id**

Specifies the revision ID.

**-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 Engineering 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



Search results are affected by the following Teamcenter Engineering preferences:

- **IMAN\_config\_rule\_name**
- **WebDesignContextDefaultSearchDistance**
- **PortalDesignContextMaxMatchingObjects**
- **PortalDesignContextMaxMatchingBOMLines**

For more information, see the *Configuration Guide*.

**-variant\_rule\_name**

Specifies the name of the variant rule.

**-revision\_rule\_name**

Specifies the revision rule, which defaults according to the **IMAN\_config\_rule\_name** preference.

**-h**

Displays help for this utility.

**-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 **\$IMAN\_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.

**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 Engineering **syslog** file contains information about the **rdv\_context\_download** utility.
- If the environment variable **RDV\_debug=Init+QPL+SearchCriteria+Variants** is set, the Teamcenter Engineering **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:

- One product item revision
- One or more component item IDs
- Optional revision rule, overwrites the **-r** argument
- Optional variant rule, overwrites the **-v** argument



Search results are affected by the following Teamcenter Engineering preferences:

- **IMAN\_config\_rule\_name**
- **WebDesignContextDefaultSearchDistance**
- **PortalDesignContextMaxMatchingObjects**
- **PortalDesignContextMaxMatchingBOMLines**

For more information, see the *Configuration Guide*.

#### 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 **IMAN\_config\_rule\_name** user preference to determine the revision rule.

```
$IMAN_ROOT/bin/tcc_context_upload -item_id TL109375 -rev_id 004
```



```
-title 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.

```
$IMAN_ROOT/bin/tcc_context_upload -item_id TL109375 -rev_id 004
-title 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:
  - Requires an NX ENGINE box zone in the top-level NX part file
  - Requires an NX-based QPL search index
  - Overlays all applicable variants
  - May require a higher value for the **PortalDesignContextMaxMatchingBOMLines** preference if the ENGINE zone has many components.

```
$IMAN_ROOT/bin/tcc_context_upload -item_id RDV00190 -rev_id 008
-title 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:
  - 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**)
  - The subset contains all components inside the ENGINE zone.
  - It requires a QPL search index.
  - It overlays all applicable variants.

```
$IMAN_ROOT/bin/tcc_context_upload -engg_change_id 000042RDV
-title RadiatorSupport -stage /tmp -tccuser subrata
-tccdestination http://usamseveh001/mydocs
```

## **Teamcenter Engineering/Teamcenter Requirements Integration**

This section describes the utility used to synchronize objects linked between Teamcenter Engineering and Teamcenter Requirements.

---

## proxy\_sync

---

**DESCRIPTION**

Synchronizes Teamcenter Engineering objects that are linked to remote Teamcenter systems engineering 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-name -p=password**

**ARGUMENTS**

**-app\_guid=app\_guid**

Selects records that were exported to the foreign application identified by the specified application GUID.

**-obj\_uid=obj\_uid**

Selects records of the Teamcenter Engineering objects specified by the object UID.



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.

**-force**

Selects and synchronizes objects regardless of the last modification date.

**-sync**



Either the **-sync**, **-delete**, or **-report** operation must be specified when running this utility.

Performs the synchronization.

**-delete**

Deletes the queried objects.

**-report**

Displays 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:

- **tcprj**
- **tcreq**
- **tceng**

This argument is mandatory when the **-diagnostic** argument is used.

**-h**

Displays help for this utility.

**ENVIRONMENT**

As specified in *Manually Setting the Teamcenter Engineering Environment* in chapter 1, *Introduction*.

**FILES**

As specified in *Log Files* in chapter 1, *Introduction*.

**EXAMPLES**

None.

---

## Chapter

# *13 Teamcenter Automotive Edition Utilities*

<a href="#">gmo_assoc_items_to_project</a>	<a href="#">13-2</a>
<a href="#">gmpdm_get_location_info</a>	<a href="#">13-4</a>
<a href="#">gmo_ipvbom_import</a>	<a href="#">13-6</a>
<a href="#">gmo_ipvbom_export</a>	<a href="#">13-8</a>
<a href="#">gmo_ipvbom_pulldate</a>	<a href="#">13-10</a>
<a href="#">gmo_create_material_form_templates</a>	<a href="#">13-12</a>
<a href="#">migrate_gmo_to_gcn_events</a>	<a href="#">13-15</a>
<a href="#">upgrade_dlist_objects</a>	<a href="#">13-17</a>



---

## Chapter

# *13 Teamcenter Automotive Edition Utilities*

---

This chapter describes the utilities that are used to administer the Teamcenter Automotive Edition and/or Teamcenter Automotive Edition–GM Overlay.

---

---

**gmo\_assoc\_items\_to\_project**

---

**DESCRIPTION**

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 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-name*] [-p=*password*]  
[-g=*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.



If Teamcenter 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 Engineering database. If you do not supply these arguments, the utility attempts 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-name* 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.

**-h**

Displays help for this utility.



**ENVIRONMENT**

As specified in *Manually Setting the Teamcenter Engineering Environment* in chapter 1, *Introduction*.

**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
```

---

**gmpdm\_get\_location\_info**

---

**DESCRIPTION**

Populates URLs for existing parts in GPDS and generates two output files:

- *site-name\_timestamp\_parts\_fnd.txt*  
Contains all parts found in the database.
- *site-name\_timestamp\_parts\_notfnd.txt*  
Contains all parts not found in the database.

This utility must be run on all Teamcenter Engineering databases. Final output can be returned to GPDS only after the utility has been run against all databases.

**SYNTAX**

**gmpdm\_get\_location\_info** [-u=*user-name*] [-p=*password*] [-g=*group-name*]  
**-input\_file**=*full-path-to-GPDS-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.



If Teamcenter 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 Engineering database. If you do not supply these arguments, the utility attempts 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-name* 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.

**-input\_file**

Specifies the absolute path to the file provided by GPDS.

**-h**

Displays help for this utility.

**ENVIRONMENT**

It is assumed that the user has write access to create the required file in the **\$IMAN\_LOG** directory.

**FILES**

If the **\$IMAN\_LOG\_DIR** is specified, all output files, error files, and audit log files are placed in that directory. If not specified, all files are placed in the directory from which the utility is run.

**RESTRICTIONS**

This utility must be run by a user with **dba** privileges.

**EXAMPLES**

- To obtain help for this utility, enter the following command on a single line:

```
gmpdm_get_location_info -h
```

- To find the parts listed in the file **/tmp/GPDS.txt**, enter the following command on a single line:

```
gmpdm_get_location_info -u=infodba -p=infodba -g=dba  
-input_file=/tmp/GPDS.txt
```

---

## gmo\_ipvbom\_import

---

### DESCRIPTION

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-name*] [-p=*password*] [-g=*group-name*]  
-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.



If Teamcenter 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 Engineering database. If you do not supply these arguments, the utility attempts 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-name* 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.

**-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 Setting the Teamcenter Engineering Environment](#) in chapter 1, [Introduction](#).

### FILES

As specified in [Log Files](#) in chapter 1, [Introduction](#).

### 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

---

### DESCRIPTION

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-name] [-p=password] [-g=group-name]  
[-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.



If Teamcenter 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 Engineering database. If you do not supply these arguments, the utility attempts 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-name* 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.

#### **-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.

#### **-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 Setting the Teamcenter Engineering Environment* in chapter 1, *Introduction*. 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* in chapter 1, *Introduction*. 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 *Configuration Guide*.

**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

### DESCRIPTION

Updates the pull date information for each build intent that is defined in a PLM XML file.

### SYNTAX

**gmo\_ipvbom\_pulldate** [-u=*user-name*] [-p=*password*] [-g=*group-name*] [-inputfile*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.



If Teamcenter 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 Engineering database. If you do not supply these arguments, the utility attempts 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-name* 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.

#### -inputfile

Specifies the full path of the PLM XML file.

#### -h

Displays help for this utility.

### ENVIRONMENT

As specified in [Manually Setting the Teamcenter Engineering Environment](#) in chapter 1, [Introduction](#).

### FILES

As specified in [Log Files](#) in chapter 1, [Introduction](#).

### 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 -inputfile=c:/tmp/PULLDATE.xml
```

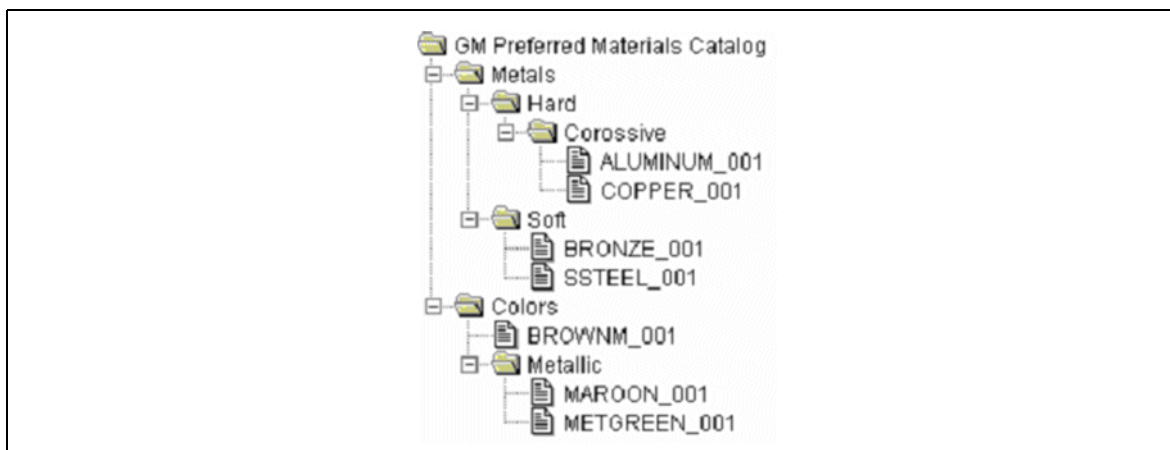
## gmo\_create\_material\_form\_templates

### DESCRIPTION

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 figure 13-1.



**Figure 13-1. Sample Directory Structure**

To achieve the structure shown in figure 13-1, 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.



The delimiting symbols described in the previous paragraph assume that these symbols are not used as values in any of the materials form fields.

**Example 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 grease#none#user#user#user

```

**SYNTAX**

**gmo\_create\_material\_form\_templates** [-u=*user-name*] [-p=*password*]  
 [-g=*group-name*] [-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.



If Teamcenter 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 Engineering database. If you do not supply these arguments, the utility attempts 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-name* 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.

**-infile**

Specifies the full path to the input file.

**-h**

Displays help for this utility.

**ENVIRONMENT**

As specified in *Manually Setting the Teamcenter Engineering Environment* in chapter 1, *Introduction*.

**FILES**

As specified in *Log Files* in chapter 1, *Introduction*. In addition, the **gmo\_create\_material\_form\_templates.log** file is created in the directory from which the utility is run.

**RESTRICTIONS**

This utility can only be run by users with **dba** privileges.

**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
```

## migrate\_gmo\_to\_gcn\_events


### DESCRIPTION

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 -g=dba [-list
<-split=number-of-subscriptions-in-a-file>] [-input_file=input-file]
[-report=report-file] [-h]
```

### ARGUMENTS

- |                    |  |
|--------------------|--|
| <b>-u</b>          | <p>Specifies the user ID.</p> <p>This is generally <b>infodba</b> or another user with administration privileges. If this argument is used without a value, the operating system user name is used.</p> <div style="display: flex; align-items: flex-start;">  <p>If Teamcenter Security Services single sign-on (SSO) is enabled for your server, the <b>-u</b> and <b>-p</b> arguments are authenticated externally through SSO rather than being authenticated against the Teamcenter Engineering database. If you do not supply these arguments, the utility attempts to join an existing SSO session. If no session is found, you are prompted to enter a user ID and password.</p> </div> |
| <b>-p</b>          | <p>Specifies the password.</p> <p>If used without a value, the system assumes a null value.</p> <p>If this argument is not used, the system assumes the <i>user-name</i> value to be the password.</p>   |
| <b>-g</b>          | <p>Specifies the group associated with the user.</p> <p>If used without a value, the user's default group is assumed.</p>  |
| <b>-list</b>       | <p>Outputs all UIDs of instances of the <b>ImanSubscription</b> class in the database.</p>   |
| <b>-split</b>      | <p>Limits the number of subscriptions output to each file.</p>   |
| <b>-input_file</b> | <p>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.</p>  |
| <b>-report</b>     | <p>Writes all messages and errors to the file specified by this argument.</p>  |
| <b>-h</b>          | <p>Displays help for this utility.</p>   |

**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
```

## upgrade\_dlist\_objects

### DESCRIPTION

Transforms distribution list objects created using Teamcenter Automotive Edition 8.1 and 2005 into **EPMAssignmentList** objects used in Teamcenter 2005 SR1.

### SYNTAX

**upgrade\_dlist\_objects** [-u=*user-name*] [-p=*password*]  
[-g=*group-name*] [-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.



If Teamcenter 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 Engineering database. If you do not supply these arguments, the utility attempts 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-name* 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.

#### -delete\_old

Deletes all existing distribution list objects in the database. This argument is optional.

#### -h

Displays help for this utility.

### ENVIRONMENT

None.

### FILES

The **upgrade\_dlist\_objects\_timestamp\_dbname** log file lists the objects that were migrated and the names of the new objects.

### RESTRICTIONS

None.

**EXAMPLES**

- To obtain help for this utility, enter the following command on a single line:  

```
upgrade_dlist_objects -h
```
- To delete old objects when migrating to the new object type, enter the following command on a single line:  

```
upgrade_dlist_objects -u=infodba -p=infodba -g=dba -delete_old
```
- To migrate to the new distribution list object type without deleting the old objects, enter the following command on a single line:

```
upgrade_dlist_objects -u=infodba -p=infodba -g=dba
```



---

**Chapter**

# *14 Computer-Aided Engineering (CAE) Utilities*

<code>cae_save_result_data</code> .....	14-2
<code>epm_import_batch_meshing_results</code> .....	14-5
<code>epm_notify_batch_meshing_results</code> .....	14-7



---

## Chapter

# *14 Computer-Aided Engineering (CAE) Utilities*

---

This chapter describes the utilities used to notify users of batch meshing results and import those results in to the Teamcenter database. In addition, this chapter describes the utility that saves result data from an analysis application.

---

---

**cae\_save\_result\_data**

---

**DESCRIPTION**

Saves the output of an analysis run (results) to Teamcenter Engineering when called by an analysis application.

**SYNTAX**

```
cae_save_result_data -u=user-name -p=password -g=group-name  
-name=result-name [-type=type-name] [-desc=result-description] [-item=item-id]  
-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.



If Teamcenter 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 Engineering database. If you do not supply these arguments, the utility attempts 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-name* 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.

**-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 ID of the item which contains the item revision containing the results dataset in which the result is created. This argument is required.

**-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 Setting the Teamcenter Engineering Environment](#) in chapter 1, [Introduction](#).

**FILES**

As specified in [Log Files](#) in chapter 1, [Introduction](#).

**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

### DESCRIPTION

Imports batch meshing results into the Teamcenter Engineering database.

### SYNTAX

```
epm_import_batch_meshing_results -u=user-id -p=password
-g=group-name -workdir=working-directory -itemid=item-id
-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.



If Teamcenter 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 Engineering database. If you do not supply these arguments, the utility attempts 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-name* 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.

#### -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 Engineering 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 and value are mandatory.

#### -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 Engineering database. This argument and value are mandatory.

**-h**

Displays help for this utility.

**ENVIRONMENT**

As specified in [Manually Setting the Teamcenter Engineering Environment](#) in chapter 1, [Introduction](#).

**FILES**

As specified in [Log Files](#) in chapter 1, [Introduction](#).

**RESTRICTIONS**

This utility is intended to be called only from the batch meshing interface from within Teamcenter Engineering .

**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

---

**DESCRIPTION**

Notifies the user of the results of a batch meshing job.

**SYNTAX**

**epm\_notify\_batch\_meshing\_results** **-u**=*user-id* **-p**=*password* **-g**=*group-name*  
**-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.



If Teamcenter 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 Engineering database. If you do not supply these arguments, the utility attempts 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-name* 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.

**-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 Setting the Teamcenter Engineering Environment](#) in chapter 1, [Introduction](#).

**FILES**

As specified in *Log Files* in chapter 1, *Introduction*.

**RESTRICTIONS**

This utility is intended to be called only from the batch meshing interface from within Teamcenter Engineering .

**EXAMPLES**

To send the results of a batch meshing job to the current user's Teamcenter Engineering mailbox (using autologin), 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
```

---

Chapter

*15 Mechatronics Utilities*

[install\\_esm\\_software\\_types](#) ..... 15-2

[install\\_kbl](#) ..... 15-4

[update\\_gde\\_types](#) ..... 15-5



---

**Chapter**

# *15 Mechatronics Utilities*

---

This chapter describes the utilities used to administer the Mechatronics solution.

---

---

**install\_esm\_software\_types**

---

**DESCRIPTION**

Installs the item types, relation types, lists of values, compatibility report datasets, and software types required by the Embedded Software Manager functionality. The utility also updates the Command Suppression preferences so that Embedded Software Manager menus are suppressed (by default). Administrative users can expose these menus to the required groups using the Command Suppression application. In addition, this utility creates several configuration preferences to capture the types used to represent software types and also creates the compatibility report configuration preferences.

**SYNTAX**

```
install_esm_software_types -u=infodba  
-p=infodba -g=dba -default_software_types="y | n" -processor_type="y | n"
```

**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.



If Teamcenter 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 Engineering database. If you do not supply these arguments, the utility attempts 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-name* 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.

**-default\_software\_types**

Specifies whether to use the default software types provided by the Embedded Software Manager or use existing item types to represent software types. Valid values are **"y"** (use default software types) and **"n"** (use existing types).

If **"n"** is specified, an administrative user must manually set the software configuration preferences after running this utility.

**-processor\_type**

Specifies whether to use the default processor types provided by the Embedded Software Manager or use an existing item type to represent the processor type. Valid values are “y” (use default processor types) and “n” (use existing types).

If “n” is specified, an administrative user must manually set the software configuration preferences after running this utility.

**-h**

Displays help for this utility.

**ENVIRONMENT**

As specified in *Manually Setting the Teamcenter Engineering Environment* in chapter 1, *Introduction*.

**FILES**

As specified in *Log Files* in chapter 1, *Introduction*.

**RESTRICTIONS**

If you choose not to install the default software or processor types, the configuration capture preference values are not set automatically; therefore, you must update the values of these preferences after installation is complete. For more information, see the *Configuration Guide*.

**EXAMPLES**

- To use the default software and processor types provided by the Embedded Software Manager, enter the following command on a single line:

```
install_esm_software_types -u=infodba -p=infodba -g=dba
-default_software_types="y" -processor_type="y"
```

- To use existing types to represent the software and processor, enter the following command on a single line:

```
install_esm_software_types -u=infodba -p=infodba -g=dba
-default_software_types="n" -processor_type="n"
```



When using existing types, you must update the values of the configuration capture preferences as described in the *Configuration Guide*.

---

**install\_kbl**

---

**DESCRIPTION**

Extends the schema to provide Teamcenter Engineering support of wire harnesses meeting the KBL standard.



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 -g=dba**

**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.



If Teamcenter 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 Engineering database. If you do not supply these arguments, the utility attempts 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-name* 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.

**ENVIRONMENT**

As specified in [Manually Setting the Teamcenter Engineering Environment](#) in chapter 1, [Introduction](#).

**FILES**

As specified in [Log Files](#) in chapter 1, [Introduction](#).

**RESTRICTIONS**

None.



## update\_gde\_types

### DESCRIPTION

Allows site administrators to updated the parent types of existing GDE types based on information provided in an input file.

### SYNTAX

```
update_gde_types -u=infodba -p=infodba -g=dba [-s=parent-type
-t=type1,type2 | -f=input-file
```

### 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.



If Teamcenter 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 Engineering database. If you do not supply these arguments, the utility attempts 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-name* 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.

#### **-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 Setting the Teamcenter Engineering Environment* in chapter 1, *Introduction*.

**FILES**

As specified in *Log Files* in chapter 1, *Introduction*.

**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,
```

---

## Chapter

# *16 Volume and Database Management Utilities*

iman_dbconfig	16-2
install_lovs	16-3
install_types	16-5
make_datasettype	16-10
make_user	16-14
Schema Browser (Character Mode)	16-20
upgrade_to_group_hierarchy	16-26
xml_validator	16-29
collect_garbage	16-30
dataset_cleanup	16-38
datasettype_cleanup	16-43
index_verifier	16-45
move_iman_files	16-47
purge_datasets	16-50
purge_mirror_volumes	16-52
purge_volumes	16-54
report_volume	16-56
review_volumes	16-58
File Management System (FMS) Utilities	16-60
fscadmin.sh/.bat	16-61
fccstat	16-66
install_encryptionkeys	16-70



---

## Chapter

# *16 Volume and Database Management Utilities*

---

This chapter describes the utilities used to manage Teamcenter volumes and databases.

---

This section describes the utilities used to manage Teamcenter Engineering volumes and databases.

---

**iman\_dbconfig**

---

**DESCRIPTION**

Populates a new Teamcenter Engineering database or upgrades an existing Teamcenter Engineering 3.x database.

**SYNTAX**

**install/iman\_dbconfig [-p] [-u] [-h]**

**ARGUMENTS****-p**

Populates the database specified by **\$ORACLE\_SID** located on **\$ORACLE\_SERVER**.

**-u**

Upgrades an existing Teamcenter Engineering 3.x database specified by **\$ORACLE\_SID** located on **\$ORACLE\_SERVER**.

**-h**

Displays help for this utility.

**ENVIRONMENT**

As specified in *Manually Setting the Teamcenter Engineering Environment* in chapter 1, *Introduction*.

To access the newly populated database, the following environment variables must be set prior to running the iMAN startup script:

**IMAN\_DB\_CONNECT**  
**ORACLE\_SERVER**  
**ORACLE\_SID**  
**POM\_SCHEMA**

**FILES**

As specified in *Log Files* in chapter 1, *Introduction*.

**RESTRICTIONS**

- Requires operating system **root** privileges on UNIX or **administrator** privileges on Windows NT.
- The **iman\_dbconfig** utility must be run from the server where an existing **\$IMAN\_DATA** directory has been installed. Your system environment is evaluated to determine if a connection to that database can be established.

**EXAMPLES**

To populate a new database, enter the following on a single line:

```
$IMAN_ROOT/install/iman_dbconfig -p
```

---

## install\_lovs

---

**DESCRIPTION**

Creates the default lists of values (LOVs). Prior to Teamcenter Engineering 8.0, default LOVs were created using the **install\_types** utility. The Teamcenter Engineering 8.0 LOV system is enhanced to allow attachments to multiple types and properties. To upgrade a pre-8.0 database, the **install\_lovs** utility must be run with the **-f=upgrade** option.

**SYNTAX**

**install\_lovs -u=user-id -p=password -g=group [-f={upgrade}] [-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.



If Teamcenter 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 Engineering database. If you do not supply these arguments, the utility attempts 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-name* 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.

**-f**

Sets additional function to be performed. Must be either null or the following:

**=upgrade** Run **ListOfValues** upgrade

**-v**

Runs utility in verbose mode, which displays the maximum amount of information. Typically, nonverbose utility sessions only display error messages.

**-h**

Displays help for this utility.

**ENVIRONMENT**

As specified in [Manually Setting the Teamcenter Engineering Environment](#) in chapter 1, [Introduction](#).

**RETURN  
VALUES**

**Return value**    0  
**upon success**

**Return value**    >1  
**upon failure**

**FILES**

As specified in [Log Files](#) in chapter 1, [Introduction](#).

**RESTRICTIONS**

Always run this utility after running the **install\_types** utility to ensure that all types have been installed when populating a new database or that all new types have been installed when upgrading an existing database.

**RETURN  
VALUES**

**Return value**    0  
**upon success**

**Return value**    >1, -1  
**upon failure**

**EXAMPLES**

- Enter the following command on a single line to create a set of default list of values in the verbose mode for a brand new database:

```
$IMAN_ROOT/bin/install_lovs -u=infodba -p=password -g=dba  
-v
```

- Enter the following command on a single line to upgrade list of values in an existing database to Teamcenter Engineering 8.0:

```
$IMAN_ROOT/bin/install_lovs -u=infodba -p=password -g=dba -f=upgrade
```



## install\_types

### DESCRIPTION

Adds new types to existing classes and may also be used to delete types, provided that they are not referenced by other objects. This utility is initially run as part of the installation program to install predefined object types.

### SYNTAX

```
install_types -u=user-id -p=password -g=group [-h] [-f={install
| list | add {-df=file-name -cf=form-definition-class-name-for-form
| -di=file-name -ci=form-definition-class-name-for-item-master |
-dir=file-name -cir=form-definition-class-name-for-ItemRevision-master
] [-o] [-cgde=storage-class-name-for-GDEType
[-view=PS-view-type-with-which-GDEType-will-be-associated ]
[-max_occs=maximum-number-of-GDE-occurrences]] | delete}] [-t=type-name]
[-pt=parent-type-name] [-c=class-name] [-ntdesc] [-lov] [-default_value] [-o]
```

### 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.



If Teamcenter 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 Engineering database. If you do not supply these arguments, the utility attempts 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-name* 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.

#### **-h**

Displays help for this utility.

#### **-f**

Specifies the function to be performed. Must be one of the following:

#### **=install**

Creates all predefined types (default function).

#### **=list**

Lists all previously defined types.

**=add**

Adds a new form, item, or item revision type. Subsequent arguments are used to define a display file and class name as follows:

**-df -cf**

Defines the display *file-name* and *class-name* for a new form type. If these arguments are not supplied, the default class is **ImanFile**.

**-di -ci**

Defines the display *file-name* and *class-name* for a new item type. If these arguments are not supplied, the **itemmaster.uid** display file and the **ItemMaster** default class are used.

**-dir -cir**

Defines the item revision display *file-name* and *class-name*. If these arguments are not supplied, the **itemversionmaster.uid** display file and the **ItemVersionMaster** class are used.

**-cdge=**

Specifies the storage class name for the GDE type. Mandatory when the specified type is a GDE type.



The storage class must be defined before adding the GDE type.

**-view=**

Specifies the PS view type with which the GDE type will be associated. Mandatory when the specified type is a GDE type.

**-max\_occs=**

Specifies the maximum number of GDE occurrences allowed for the GDE type. Mandatory when the specified type is a GDE type.

**=delete**

Delete the type set by the **-t=type-name** argument.

**-t**

Specifies the *type-name* to be added or deleted.

**-pt**

Specifies the name of the parent type of the type specified by the **-t** argument.

**-c**

Specifies the *class-name* to be added or deleted.



- This argument is not required if the **-t** and **-pt** arguments are used. The new type is created for the same class as the parent type.
- If the **-c** argument is used in conjunction with GDE types, the class name must be **GDETypeDefinition**.

**-ntdesc**

Describes the note type. This argument is required when adding note types.

**-lov**

Specifies the list of values (LOV) name. The specified LOV must already exist. This argument is only applicable to note types.

**-default\_value**

Specifies the note type default value for LOV. The **-lov** argument must be specified if the **-default\_value** argument is used. This argument is only applicable to note types.

**-o**

Overrides validation of class name.

**ENVIRONMENT**

As specified in [Manually Setting the Teamcenter Engineering Environment](#) in chapter 1, [Introduction](#).

**FILES**

As specified in [Log Files](#) in chapter 1, [Introduction](#).

**RESTRICTIONS**

- When adding new types, observe the following naming conventions:
  - A new type name may contain up to 16 characters.
  - Type names are provided to the iMAN methods module and must be unique in the entire system.
  - All predefined type names are prefixed with **IMAN\_** (for example, **IMAN\_specification** and **IMAN\_manifestation**).
  - User-defined type names should be prefixed by a name meaningful to your site, for example, **MYCOMPANY\_**.
  - Use uppercase for prefixes and lowercase for the basic name.
  - Words in names should be separated by the underscore character.
  - For names, avoid using prefixes or suffixes that imply relationship direction. For example, use **specification** rather than **specified\_by**.
  - Avoid abbreviations.
- Types cannot be deleted if they are referenced by another object.
- Form types are automatically created for items and item revisions. The form type name is appended with **gk\_master\_prefix** and **gk\_master\_suffix** to the item form type name and item revision form type name. These prefixes and suffixes are defined in the site **iman\_text.uil** file.

- When using the **-f=add -df=file-name -cf=class-name** argument to add a new form type, the **-df** and **-cf** arguments are optional. However, to use a form, a display file must be defined for each form type. Form display files can be defined using the Form Type Definition dialog window.
- When using **-f=add -di=file-name -ci=class-name** or **-f=add -dir=file-name -cir=class-name** to add a new item and item revision type, the **-di**, **-ci**, **-dir**, and **-cir** arguments are optional. If these arguments are not supplied, defaults are used.
- An **ImanType** must be created for all classes before defining the type using the **WSOM\_set\_object\_type** preference. Additionally, all form types must be represented by a corresponding **ImanType**.
- The **-ntdesc**, **-lov** and **-default\_value** values are only applicable when adding note types using the **-c=NoteType** argument.

#### EXAMPLES

- To list all types, enter the following command on a single line:  

```
$IMAN_ROOT/bin/install_types -u=infodba -p=password -g=dba -f=list
```
- To add a new relation type, **my\_relation\_type**, based on the class **ImanRelation**, enter the following command on a single line:  

```
$IMAN_ROOT/bin/install_types -u=infodba -p=password -g=dba -f=add  
-t=my_relation_type -c=ImanRelation
```
- To add a new item type (**my\_item\_type**), based on the class item, enter the following command on a single line:  

```
$IMAN_ROOT/bin/install_types -u=infodba -p=password -g=dba -f=add  
-t=my_item_type -c=Item
```
- To add a new item type, **Engineer Part**, that uses the **part.uid** file for the item master display file and the **partrevision.uid** file for the item revision master display file, enter the following command on a single line:  

```
$IMAN_ROOT/bin/install_types -u=infodba -p=password -g=dba -f=add  
-t="Engineer Part" -c=Item -di=part.uid -dir=partrevision.uid
```
- To add a new item type, **Gear**, as a subtype of **Engineer Part**, enter the following command on a single line:  

```
$IMAN_ROOT/bin/install_types -u=infodba -p=password -g=dba  
-f=add -t=Gear -pt="Engineer Part"
```

- To add a new form type **Contract Form** that uses the **contract.uid** file for the display file name and **Contract** for the form definition class, enter the following command on a single line:

```
$IMAN_ROOT/bin/install_types -u=infodba -p=password -g=dba -f=add
-t="Contract tForm" -c=Form -df=contract.uid -cf=Contract
```

- To add a new form type, **SimpleContractForm**, that is a subtype of **Contract Form** using the definition class of the **Contract Form** as the form definition class, enter the following command on a single line:

```
$IMAN_ROOT/bin/install_types -u=infodba -p=password -g=dba -f=add
-t=SimpleContractForm -pt="Contract Form"
```

- To add a new form type, **SimpleContractForm**, that is a subtype of **Contract Form** using **SimpleContract** as the form definition class, enter the following command on a single line:

```
$IMAN_ROOT/bin/install_types -u=infodba -p=password -g=dba -f=add
-t=SimpleContractForm -pt="Contract Form" -cf=SimpleContract
```

- To override the validation of the class name, use the **-o** option.
- To add a new GDE type named **MyGDE** using **MyGDEStorage** as the storage class and **view** as the view type, enter the following command on a single line:

```
$IMAN_ROOT/bin/install_types -u=infodba -p=password -g=dba -f=add
-t=MyGDE -c=GDETypeDefinition -cgde=MyGDEStorage -view=view -max_occs=-1
```

- To add a new GDE link type, **MyGDELink**, using **MyGDELinkStorage** as the storage class, enter the following command on a single line:

```
$IMAN_ROOT/bin/install_types -u=infodba -p=password -g=dba -f=add
-t=MyGDELink -c=GDELink -cgde=MyGDELinkStorage
```



The **MyGDEStorage** class must be predefined.

- To add a new view type called **Design**, enter the following command on a single line:

```
install_types-u=infodba -p=password -g=dba -f=add
-t=Design -c= PSViewType
```

- To add a new status type called **Approved**, enter the following command on a single line:

```
install_types-u=infodba -p=password -g=dba -f=add -t=Approved -c=TaskType
```

- To add a new note type called **DesignNote**, enter the following command on a single line:

```
install_types-u=infodba -p=password -g=dba -f=add -t=DesignNote
-c=NoteType -ntdesc="Notes for Design"
```

---

**make\_datasettype**

---

**DESCRIPTION**

Defines new dataset types and tools for Teamcenter Engineering. The **make\_datasettype** utility can define a single dataset type or tool directly from the command line or create multiple dataset types and tools in batch mode using an input definition file.

The **make\_datasettype** utility adds only new object definitions and does not change existing definitions of tools or named references in dataset types.

**Sample Definition Files**

Three sample definition files are shipped with Teamcenter Engineering: **editor\_shell.dat**, **ug\_shell.dat**, and **ugmanager\_exports.dat**.

The **editor\_shell.dat** file is a sample definition file for creating other dataset type definition files. The file is divided into three sections, as follows:

<b>Tool definition</b>	Specifies software applications.
<b>Dataset type definition</b>	Specifies dataset types and associates each with tools and file references.
<b>Export tool</b>	Specifies files to be exported and parameters to be passed for each tool/dataset type combination.

Figure 16-1 shows a sample **editor\_shell.dat** file containing one NX tool and dataset type. For NX Manager tool definitions, use the exact symbol and formats for the tools as shown here and specify no parameters in the export section.

```
! editor_shell.dat
! =====
!
! File description:
!
! Supplied datasettype, tool and export definitions for editor
! For use with the make_datasettype utility. This file should be used
! during the installation for IMAN-UG/Manager, and modified if you wish.
! to run use the command
!
! make_datasettype -u=infodba -p=password -g=dba
! -definition=editor_shell.dat
!
! with optional extra -v argument to see messages as it processes the file.
! As supplied this contains tool definitions and export definitions for
! the Text dataset types.
!
! -----
! Datasettype definitions. Each has syntax
!
! DATASETTYPE name
! [TOOL name]*
! [REFERENCE name format template]*
! END_DATASETTYPE
!
```

**Figure 16-1. Sample editor\_shell.dat file** (Continued)

```

! Text dataset type, with its tools and references
DATASETTYPE Text
! Valid tools for Text dataset type:
TOOL "TextEditor"
TOOL "NoteEditor"
!References for Text dataset type:
REFERENCE Text Text *
END_DATASETTYPE
! Text dataset type, with its tools and references
!-----
! Now the exports - each has syntax
!
! EXPORT TOOL name DATASETTYPE name
! [EXPORT reference name]*
! END_EXPORT
!
! Exports for the Base tool(note that all other tools include these
! as a subset):
EXPORT TOOL "TextEditor" DATASETTYPE Text
EXPORT Text
PARAMETER $Text
END_EXPORT
EXPORT TOOL "NoteEditor" DATASETTYPE Text
EXPORT Text
PARAMETER $Text
END_Export

```

Figure 16-1. Sample editor\_shell.dat file

## SYNTAX

```

make_datasettype -u=user-id -p=password -g=group {-type=name [[-rm_ref]
-ref= name [-format={Text | Binary | Object}] [-template=file-name]]
[[-rm_tool] -tool=name [-program=name] [-tool_format=format]
[[-rm_action] -action={open | openusing | print | printusing}
[-export=name] [-param=name]]] | -tool=name [-program=name ]
[-tool_format=format] | [-definition=file-name ]} [-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.



If Teamcenter 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 Engineering database. If you do not supply these arguments, the utility attempts 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-name* 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.

**-type**

Creates a new dataset type with the name specified, if there is no existing dataset type by that name. The **-tool** argument must be supplied when creating a new dataset type. If an existing dataset type identified by the **-type** argument is found, modification of the existing dataset type is assumed.

**-tool**

Creates a new tool with the name specified, if there is no existing tool by the that name. If an existing tool of the specified name is found, modification of the existing tool is assumed. If the **-type** argument is also specified, the tool is added to the dataset type identified by the **-type** argument.

**-rm\_tool**

Indicates that the tool identified by the **-tool** argument is removed from the dataset type identified by the **-type** argument.

**-program**

Defines the program to use with the tool.

**-tool\_format**

Specifies the input and output format of tool files.

**-ref**

Specifies the named reference associated with the new dataset type. The default is **ImanFile**. See restriction #1. If the **-rm\_ref** argument is also specified, the named reference is removed from the specified dataset type.

**-format**

Specifies the file format for the dataset type named reference, either **text**, **binary**, or **object**. The default is **text**.

**-template**

Specifies the template for the file reference. The default is **\***.

**-rm\_ref**

Indicates that the named reference specified by the **-ref** argument will be removed from the dataset type identified by the **-type** argument. Optionally, the **-format** and **-template** arguments can be combined to explicitly identify the named reference entry to be removed. This argument must be used with the **-ref** and **-type** arguments.

**-action**

Specifies one of the following actions to be performed by the tool:

**open**  
**opening**  
**print**  
**printing**

If the **-rm\_action** argument is also specified, the modification of the specified action is assumed.



**-rm\_action**

Indicates that the named reference specified by the **-export** argument and parameters specified by the **-param** argument are removed from the tool identified by the **-tool** argument in the dataset type identified by **-type** argument. This argument must be used with the **-action**, **-tool**, and **-type** arguments.

**-export**

Specifies the reference to export for the tool, if the **-rm\_action** is not specified.

Specifies the reference to be removed from the action if the **-rm\_action** argument is specified.

**-param**

Specifies the parameter to pass to the tool when performing the action defined by the **-action** argument, if the **-rm\_action** argument is not specified. Specifies the parameters to be removed from the action if the **-rm\_action** argument is specified.

**-definition**

Specifies the dataset type definition file used for batch mode processing. If an existing tool is encountered, the tool is modified. If an existing dataset type is encountered, the dataset type is modified. This option cannot be used to achieve the functionality provided by the **-rm\_tool**, **-rm\_ref**, or **-rm\_action** arguments.

**-v**

Runs utility in verbose mode to display the maximum amount of information. Typically, nonverbose utility sessions only display error messages.

**ENVIRONMENT**

As specified in *Manually Setting the Teamcenter Engineering Environment* in chapter 1, *Introduction*.

**FILES**

As specified in *Log Files* in chapter 1, *Introduction*.

**RESTRICTIONS**

- Each named reference can be one of two different types: **ImanFile** or another existing Teamcenter Engineering object type (for example, form or folder).
- Definition file keyword entries must be all upper case.
- Definition file string entries must be enclosed in double quotes if they contain blanks.

**RETURN  
VALUES**

**Return value**    0  
**upon success**

**Return value**    >1  
**upon failure**

**EXAMPLES**

None.

---

**make\_user**

---

**DESCRIPTION**

Creates new users, groups, persons, roles, and volumes outside of a Teamcenter Engineering 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 source code for this utility is located in the samples directory; the executable is located in the **\$IMAN\_ROOT/bin** directory.

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.



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, or user using command line options. Use the **-update** option to modify the properties of user, group, or role objects. Use the **-rename** option to rename an existing user, group, or role. For a user, the **-rename** option specifies the user ID; for group and role objects, the **-rename** option specifies the group name and role name, respectively.

Use the **-description** option to set or modify the description of the group and/or role. When using the **-description** option with the **-role** and **-group** options, the same description is set for both the group and role.

**SYNTAX**

```
make_user -u=user-id -p=password -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] [-role=name]  
[-rename=user-id | group-name | role-name] [-os] [-volume=name]  
[-node=name] [-path=name] [-file=file] [-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.



If Teamcenter 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 Engineering database. If you do not supply these arguments, the utility attempts 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-name* 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.

**-update**

Use when modifying any of the existing user, group, or role objects. See restriction #5.

**-user**

Creates a new Teamcenter Engineering user.

**-password**

Creates a password for the new user.



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. See restriction #7.

**-role**

Specifies the role to which the new user is assigned.

**-rename**

Specifies a new name for an existing user, group or role.

**-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 restriction #4.

**-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
```

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.



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** option.

**-v**

Runs the utility in verbose mode to display the maximum amount of information. Typically, nonverbose utility sessions only display error messages.

**-h**

Displays help for this utility.

#### ENVIRONMENT

As specified in *Manually Setting the Teamcenter Engineering Environment* in chapter 1, *Introduction*.

#### FILES

As specified in *Log Files* in chapter 1, *Introduction*.

#### RESTRICTIONS

1. If the argument parameter contains spaces, it must be enclosed in quotes, for example, **product validation**.
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** options are specified when using the **-update** option, the user object is assumed as the target object of the update. If the **-group** or **-role** options are specified without the **-user** option, the group object is updated. If the **-role** option is specified without the **-user** or **-group** options, 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 option values for the **-status** option are **0** (active) and **1** (inactive). If this option is not specified, the default status is active.

#### RETURN VALUES

**Return value**    0  
**upon success**

**Return value**    1  
**upon failure**

## 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:

```
$IMAN_ROOT/bin/make_user -u=infodba -p=password -g=dba -user=tm
-group=london.dev -person=tom
$IMAN_ROOT/bin/make_user -u=infodba -p=password -g=dba -user=dm
-group=london.dev -person=dan
$IMAN_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:

```
$IMAN_ROOT/bin/make_user -u=infodba -p=password -g=dba -user=tm
-group=london.dev -role=planner
$IMAN_ROOT/bin/make_user -u=infodba -p=password -g=dba -user=dm
-group=london.dev -role=qc
$IMAN_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:

```
$IMAN_ROOT/bin/make_user -u=infodba -p=password -g=dba -user=tm -group=qa
$IMAN_ROOT/bin/make_user -u=infodba -p=password -g=dba
-user=dm -group=qa
$IMAN_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:

```
$IMAN_ROOT/bin/make_user -u=infodba -p=password -g=dba -volume=test
-node=svr1 -path=/user/volumes/test
```

- 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:

```
$IMAN_ROOT/bin/make_user -u=infodba -p=password -g=dba
-volume=test -node=svr1
-path=/user/volumes/test -group=hongkong.dev2
```

- 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 (**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:

```
$IMAN_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
```

---

## Schema Browser (Character Mode)

---

### DESCRIPTION

The **sb** utility is a character-mode (VT100) utility used to view (browse) the Persistent Object Model (POM) class hierarchy (schema) as well as to create and manage object classes and attributes. The utility operates in two modes: browser mode and command mode.

#### Browser mode

If either the **-l** or **-a** argument is supplied, the **sb** utility runs in browser mode. Browser mode is used to view the object class hierarchy and class attributes. When the utility is in browser mode, it terminates after executing the requested command.

#### Command mode

If no arguments are supplied, the **sb** utility runs in command mode. Command mode displays a command line prompt (**sb>**) used to access the schema browser commands.



The **sb** utility is extremely powerful. New Teamcenter Engineering classes can be easily created, and once a class is defined it cannot be deleted. The number of classes in Teamcenter Engineering greatly affects overall performance. Therefore, ensure that all classes created using the **sb** utility are necessary before they are instantiated.

### SYNTAX

**sb -u=user-id -p=password -g=group [-l] | [-a=class-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.



If Teamcenter 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 Engineering database. If you do not supply these arguments, the utility attempts 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-name* 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.



**-l**

Lists object class hierarchy (schema) in browser mode. The **-l** and **-a** arguments are mutually exclusive.

**-a**

List attributes of the specified class in browser mode. The **-l** and **-a** arguments are mutually exclusive.

**-set**

Sets the initial value and upper and lower bounds of an attribute. The syntax for the **-set** command is as follows:

```
sb> set class-name attr-name ['='initial-value]
['>'lowerbound] ['<'upperbound]
```



There must not be white space between the (=), (<), and (>) symbols and the values. If the symbols do not contain a value, (...) should be used.

This command alters the contents of the database.

**ENVIRONMENT**

As specified in [Manually Setting the Teamcenter Engineering Environment](#) in chapter 1.

**FILES**

As specified in [Log Files](#) in chapter 1.

**RESTRICTIONS**

1. A class cannot be deleted if it is defined or referenced by another object.
2. An attribute cannot be removed if it is referenced by another object.
3. When extending the Teamcenter Engineering schema by adding new classes and attributes, there is a danger that you may add attributes and classes that conflict with future releases of Teamcenter Engineering. To avoid this possibility, UGS strongly recommends that you prefix your class and attribute names with something unique, such as an underscore (\_) or your site name (for example, **mysite\_**). UGS cannot guarantee that sites that do not follow this convention will be able to successfully upgrade to future releases of Teamcenter Engineering.

**RETURN  
VALUES**

**Return value** 0  
**upon success**

**Return value** 0  
**upon failure**

**EXAMPLES**

- To list the object class hierarchy (schema), enter the following command on a single line:
- To list attributes of the **item** class, enter the following command on a single line:
- To enter command mode, enter the following command on a single line:

```
$IMAN_ROOT/bin/sb -u=infodba -p=password -g=dba -l
```

```
$IMAN_ROOT/bin/sb -u=infodba -p=password -g=dba -a=Item
```

```
$IMAN_ROOT/bin/sb -u=infodba -p=password -g=dba
```

**COMMANDS**

When the **sb** utility is in command mode, the following commands are available at the (**sb>**) prompt:

**SYNTAX**

```
list  
attributes classname [attribute]  
create parentclassname newclassname  
delete classname  
define classname attribute {POM_int | POM_short | POM_string | POM_float |  
POM_double | POM_logical | POM_date | POM_char | POM_note} [length]  
[{-A | r | w | u | n | k | f | s | i | S}]  
reference classname refname [refclass]  
[{-A | r | w | u | n | k | f | s | i | S}]  
remove classname attribute  
save [classname]  
quit  
exit  
help
```

## ARGUMENTS

**list**

Lists object class hierarchy (schema). The same as the **sb -l** command.

**attributes**

Lists attributes of a specified class. Similar to the **sb -a** command. The class name must be supplied. If *attribute* is not supplied, general information about all attributes of the class are returned. If the attribute is supplied, detailed information about that attribute is returned.

**create**

Creates a new class. Command should contain any super-class (parent class) as well as the new class name. See restriction #3 above.



To be able to view the new class in the Schema Editor, you must use the **save** option in conjunction with the **create** option.

**delete**

Deletes a specified class from the database. See restriction #1.

**define**

Using the **define** argument to add attributes to classes can produce undesirable results. UGS recommends using the **install** utility to add attributes to new classes. For more information, see *install* in chapter 10, *Maintenance Utilities*.

Defines (adds) new attributes to the class. This command adds all attributes except references. See restriction #3. One or more of the following arguments must be supplied:

**A**

Specifies the attribute as a variable length array (VLA).

**r**

Grants public (world) read access.

**w**

Grants public (world) write access.

**u**

Specifies a unique attribute.

**n**

Allows nulls.

**k**

Specifies that the attribute is candidate key.

**f**

Followed on export.

**s**

Exports as string.

**i**

Ignores errors on import.

**s**

Attribute may not be stubbed.

**POM\_note**

**POM\_note** attributes can only be created before the class is saved; these attributes cannot be added to an existing class. For example:

```
sb> create POM_object ACME_Class1
sb> define ACME_Class1 ACME_Note1 POM_note 1024 -rwn define... (other
sb> attributes in the new class) save ACME_Class1
```

**reference**

Defines (adds) new reference attribute to the class. One or more of the following arguments must be supplied:

**A**

Specifies the attribute as a variable length array (VLA).

**r**

Grants public (world) read access.

**w**

Grants public (world) write access.

**u**

Specifies a unique attribute.

**n**

Allows nulls.

**k**

Specifies that the attribute is candidate key.

**f**

Followed on export.

**s**

Export as string.

**i**

Ignores errors on import.

**s**

Attribute may not be stubbed.

**save**

Saves class to the database. If the class is not supplied, all class definitions modified during the session are saved.

**quit**

Quits the **sb** session without saving changes. If you have modified a class definition, **sb** prompts for confirmation that changes will not be saved before exiting.

**exit**

Exits the **sb** session and saves changes. All class definitions modified during the session are saved. The same as entering **save** then **quit**.

**help**

Displays help for command mode.

**COMMAND  
MODE  
EXAMPLES**

- To generate a summary report of all Folder attributes, enter the following command at the **sb>** prompt:

```
attributes Folder
```

- To generate a detailed listing of the Folder attribute contents, enter the following command at the **sb>** prompt:

```
attributes Folder contents
```

- To create a new class called **my\_class** of super-class Folder that can be viewed in the Schema Editor, enter the following command at the **sb>** prompt:

```
create Folder my_class save my_class
```



You can save all created classes by using the **save** command without specifying a class name.

- To delete the **my\_class** class from the database, enter the following command at the **sb>** prompt:

```
delete my_class
```

- To define an attribute called **my\_attr** as a string 32 characters in length with public read and write access and add it to the **my\_class** class, enter the following command at the **sb>** prompt:

```
define my_class my_attr POM_string 32 -rw
```

- To define an attribute called **my\_second\_attr** as a variable length array of integers and add it to the **my\_class** class, enter the following command at the **sb>** prompt:

```
define my_class my_second_attr POM_int -A
```

- To define a reference called **my\_parent** that points to an instance of class Folder and add it to the **my\_class** class, enter the following command at the **sb>** prompt:

```
define my_class my_parent Folder
```

---

**upgrade\_to\_group\_hierarchy**

---

**DESCRIPTION**

Converts your Teamcenter Engineering installation to support hierarchical groups. This involves doing two things:

- Converting the existing groups to support group hierarchies, and optionally to create one or more hierarchies based on the names of the existing groups.
- Converting the existing volumes to allow for group hierarchies.



This program is automatically run as part of the upgrade to Teamcenter Engineering 6.0.

During the creation of the new hierarchies, new groups may be created. However, the existing groups are merely (potentially) renamed. Therefore, they retain their associated roles and users, so there is no need to repopulate the groups, all membership information remains unchanged.

**SYNTAX**

**upgrade\_to\_group\_hierarchy** [-u=*user-id*] [-p=*password*] [-g=*group*] [-h]  
[-separators=*separators*] [-ignore\_from=*terminators*] [-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.



If Teamcenter 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 Engineering database. If you do not supply these arguments, the utility attempts 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-name* 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.

**-h**

Displays help for this utility.

**-separators**

When creating a hierarchy based on the names of existing groups, *separators* is a list of characters that are to be considered as separating a group's name from its parent's name.

Default value: .

**-ignore\_from**

When creating a hierarchy based on the names of existing groups, *terminators* is a list of characters at which processing of a group's name should stop. In other words, any of these characters and anything after them is ignored.

Default value: none.

**-v**

Displays information about what processing is being done.

**ENVIRONMENT**

As specified in [Manually Setting the Teamcenter Engineering Environment](#) in chapter 1, [Introduction](#).

**FILES**

As specified in [Log Files](#) in chapter 1, [Introduction](#).

**RESTRICTIONS**

Requires Teamcenter Engineering administrator privileges.

**EXAMPLES**

- Suppose that you have an existing (nonhierarchical) group called **development.paris.europe**. If you run the following command:

```
$IMAN_ROOT/bin/upgrade_to_group_hierarchy -u=infodba -p=password -g=dba
```

The following things occur:

- A new root group called **europe** (root groups have no parent group) is created.
- A new group called **paris** is created as a child of **europe**.
- The existing group is renamed **development** and becomes a child of **paris**.

- Suppose instead that you have two existing (nonhierarchical) groups called **development.paris.europe** and **marketing.paris.europe**. (Of course, being nonhierarchical, these groups are entirely independent at this point, despite the connection implied by their names.)

If you run the following command:

```
$IMAN_ROOT/bin/upgrade_to_group_hierarchy -u=infodba -p=password -g=dba
```

The following things occur:

- A new root group called **europe** (root groups have no parent group) is created.
- A new group called **paris** is created as a child group of **europe**.
- The existing **development.paris.europe** is renamed to **development** and makes it a child of **paris**.
- The existing **marketing.paris.europe** is renamed to **marketing** and also becomes a child of **paris**.

Thus the two original groups are now siblings, as part of an overall actual group hierarchy that reflects the original implied hierarchy.

- Suppose that you have an existing (nonhierarchical) group called **development#paris#europe@installation1**. Running **upgrade\_to\_group\_hierarchy -u=infodba -p=password -g=dba -separators="#" -ignore\_from=@"** has the same overall effect as in the first example, because you have specified the pound sign (#) to be the separator instead of the default dot (.), and that processing should stop when it reaches a the (@) character.



---

## xml\_validator

---

### DESCRIPTION

Checks the XML file against the document type definition (DTD) to which it should conform.

### SYNTAX

**xml\_validator** [-v=**always** | **never** | **auto\***] *file.xml*

### ARGUMENTS

**-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 Setting the Teamcenter Engineering Environment* in chapter 1, *Introduction*.

### FILES

As specified in *Log Files* in chapter 1, *Introduction*.

### 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
```

---

**collect\_garbage**

---

**DESCRIPTION**

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\_gabage** 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 basked. 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=infodba -p=password -g=dba -rf=report-file-name  
-if=input-file-name -sub_folder=folder-name [-dataset] [-item] [-occurrence]  
[-form] [-folder] [-envelope] [-all] [-orphan] [-delete] [-report]  
[-query] [-h][-start=number] [-end=number]
```



You must run Teamcenter Engineering 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.



If Teamcenter 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 Engineering database. If you do not supply these arguments, the utility attempts 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-name* 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.

**-dataset**

Specifies that datasets be collected or deleted.

**-item**

Specifies that items be collected or deleted.



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.

**-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.

UGS 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. Orphan operations should not be combined with other operations. See restriction #2.



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.

**-query**

Queries the database for the instances of the specified object type when used in combination with a defined object type.



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.



The **-report** argument generates output in a different format than the **-orphan** operations because orphan objects are not valid workspace objects.

**-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.



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, UGS 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.



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.

**-sub\_folder=folder-name**

Creates a subfolder name from unreferenced instances of the given object to be deleted. When an object type and subfolder name are defined, this argument deletes all defined instances in the specified subfolder within the **WASTE BASKET** folder of the specified object type.



This argument works only in combination with an object type argument (**-item**, **-dataset**, **-form**, **-envelope**, **-folder**, **-all**) and the **-delete** argument.

**-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.

**-h**

Displays help for this utility.

**ENVIRONMENT**

As specified in *Manually Setting the Teamcenter Engineering Environment* in chapter 1, *Introduction*.

**FILES**

As specified in *Log Files* in chapter 1, *Introduction*.

**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
```

When working with a large database, or with a large number of instances in a report file, UGS 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 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.

```
collect_garbage -dataset -sub_folder=WBDSET_dataset_report1 -delete
```

- 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
```

**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** file when running in orphan collection mode. Errors reported as errors in attempting to load an indirected object are benign.

---

**dataset\_cleanup**

---

**DESCRIPTION**

Repairs corrupted datasets and removes orphaned revision anchors.



UGS 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 **-g**=group **-rf**=file-name | **-if**=file-name [**-of**=log-file-name] [**-b**=beginning-anchor] **-e**=ending-anchor

**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.



If Teamcenter 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 Engineering database. If you do not supply these arguments, the utility attempts 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-name* 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.

#### **-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**.



A revision anchor is an object that keeps track of a set of revisions of some object. One such class of objects is datasets.

#### ENVIRONMENT

As specified in [Manually Setting the Teamcenter Engineering Environment](#) in chapter 1, [Introduction](#).

#### FILES

As specified in [Log Files](#) in chapter 1, [Introduction](#).

#### RESTRICTIONS

None.

#### EXAMPLES

- To generate a report file called **myreportfile** listing corrupted dataset objects, enter the following command on a single line:

```
$IMAN_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:

```
$IMAN_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:

```
$IMAN_BIN/dataset_cleanup -u=infodba -p=infodba -g=dba -b=1 -e=500
-rf=dataset_cleanup_500.report
$IMAN_BIN/dataset_cleanup -u=infodba -p=infodba -g=dba -b=501 -e=1000
-rf=dataset_cleanup_1000.report
```

#### 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:

```
$IMAN_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:

```
$IMAN_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.

---

## datasettype\_cleanup

---

**DESCRIPTION**

Locates corrupt dataset types and removes them from the database.

The Teamcenter Engineering database uses three objects to represent a dataset type:

**DatasetType**

**ImanType**

**ImanNextID** (use of this object depends on the database configuration)

The utility has two modes of operation:

- *General mode* examines all **ImanType** objects and removes **ImanType** objects associated with datasets that do not have a matching **DatasetType**. It also removes **ImanNextId** objects with the same name as the **ImanType** object. This mode is the default mode of the utility.
- *Targeted mode* takes the specified dataset type name and examines the database for the matching **DatasetType**, **ImanType**, and **ImanNextID**. If the utility finds a matching **DatasetType** and **ImanType**, the dataset type is considered uncorrupted, and the utility takes no action. If one or both are missing, the dataset type is considered corrupt and the utility removes the objects matching the specified name.

The key differences between the modes are:

- General mode operates on all dataset types; targeted mode operates only on the specified dataset type.
- General mode examines all **ImanType** objects and handles the **ImanNextId** only if there is an **ImanType** object with no matching **DatasetType** object. Targeted mode removes the **ImanNextId** even if there is no matching **DatasetType** object or **ImanType** object.

**SYNTAX**

```
datasettype_cleanup -u=user-name -p=password [-debug]  
[-target=dataset-type]
```

**ARGUMENTS**

**-u**

Specifies the user name.

**-p**

Specifies the password.

**-debug**

Prints information about what the utility is processing and a message about what the utility intends to do. When you specify this argument, the utility does not clean up dataset types in either the general mode or the targeted mode of operation.

**-target**

Specifies the target mode of operation in which the utility examines the database for **DatasetType**, **ImanType**, and **ImanNextId** objects that match the specified dataset type. If you do not include the **-target** argument, the utility operates in general mode.



The targeted mode is dangerous and should be used with care; **ImanNextId** objects are used by other things than **DatasetTypes** and misuse could cause problems.

**ENVIRONMENT**

As specified in [Manually Setting the Teamcenter Engineering Environment](#) in chapter 1, [Introduction](#).

**FILES**

As specified in [Log Files](#) in chapter 1, [Introduction](#).

**RESTRICTIONS**

None.



---

## index\_verifier

---

### DESCRIPTION

Detects missing indexes in a Teamcenter Engineering database. There are five types of indexes that can be detected using this utility:

- Indexes on the primary key of each Teamcenter Engineering class.  
The PUID (internal attribute of each table mapped to a Teamcenter Engineering class) must have a unique index.
- Indexes on variable length arrays (VLA).  
Each VLA must have two indexes, as follows:
  - An index on the PUID+PSEQ. This index must be unique.
  - 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 Engineering.  
These indexes are created using Teamcenter Engineering 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 Engineering in use. These functional indexes may include, but are not limited to the following:
  - A functional index on **WorkspaceObject.object\_type**
  - A functional index on **WorkspaceObject.object\_desc**
  - A functional index on **WorkspaceObject.object\_name**
  - A functional index on **Item.Item\_id**
  - 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-name* **-p**=*password* **-g**=*group-name*

**ARGUMENTS****-u**

This is generally a **infodba** or another user with administration privileges. If this argument is used without a value, the **userid** is blank and you must supply a value. If this argument is not used, a **USAGE** message is displayed.

**-p**

Specifies the password. If used without a value, the system assumes a null value. If this argument is not used, a **USAGE** message is displayed.

**-g**

Specifies the group associated with the user. If used without a value, the group is blank. If this argument is not used, a **USAGE** message is displayed.

**-h**

Displays help for this utility.

**ENVIRONMENT**

As specified in [Manually Setting the Teamcenter Engineering Environment](#) in chapter 1, [Introduction](#).

**FILES**

As specified in [Log Files](#) in chapter 1, [Introduction](#).

**RESTRICTIONS**

Users must have administrative privileges to run this utility. This utility can be run while users are logged in to Teamcenter Engineering.

**EXAMPLES**

None.

---

## move\_iman\_files

---

**DESCRIPTION**

Generates the list of Teamcenter Engineering data that matches the specified criterion for given source volumes and moves them to the specified destination volumes.

For information on the environment variables associated with file relocation, see the *Configuration Guide*.

**SYNTAX**

**move\_iman\_files**

**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.



If Teamcenter 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 Engineering database. If you do not supply these arguments, the utility attempts 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-name* 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.

**-source\_vol**

Specifies the names of Teamcenter Engineering source volumes.

To specify more than one volume name, supply the volume names delimited by the tilde character (~).

**-dest\_vol**

Specifies the names of Teamcenter Engineering destination volumes.

To specify more than one volume name, supply the volume names delimited by the tilde character (~).

**-list\_file**

Specifies the complete path and name of the file that contains the predetermined list of OS files that need to be moved. If the **-list\_file** argument is supplied, the utility adds these files to the generated output list.

**-mode**

Specifies whether the utility lists, moves, or lists and moves data.

**-list**

Lists the files to be moved in the specified source volumes. If a predetermined list is available, it is added to the output list generated based on the specified criterion.

This selection requires the **-source\_vol** argument.

**-move**

Takes the input list file and moves the Teamcenter Engineering data to the specified destination volumes.

This selection requires both the **-list\_file** and **-dest\_vol** arguments.

**-list\_and\_move**

Lists and moves the files from the source volumes to the destination volumes. If a predetermined list is available, it is added to the output list generated based on the specified criterion before moving.

This selection requires both the **-source\_vol** and **-dest\_vol** arguments.

**-atime**

Specifies access time in number of days.

The syntax of this argument is similar to the **-atime** syntax of the UNIX **find** command. The plus (+) argument lists the files that were not last accessed in the specified number of days, and the minus (-) argument lists the files that were last accessed in the specified number of days.

If the **-atime** argument is not supplied, the utility lists all the files in the supplied volumes.

**-h**

Displays help for this utility.

**RESTRICTIONS**

None.

**ENVIRONMENT**

If the **-txn\_count** argument is not supplied on the command line, the **MOVE\_TRANSACTIONS\_COUNT** environment variable must be defined and set to a numeric value.

It is assumed that the user running this utility has write access to create the required files in the **\$IMAN\_LOG** directory. The utility uses this directory to store the configuration file, which allows the process to shut down gracefully.

**FILES**

- The **HSM\_os\_files\_\*.txt** file is placed in the directory from which the utility is run. The **move\_iman\_files\*.log** and **move\_iman\_files\*.jnl** files are placed in the directory specified by the **%IMAN\_TMP\_DIR%** variable. If this variable is not set, these files are placed in the directory specified by the **%TEMP%** variable.
- The format of the file name in which the generated list of OS files is placed is as follows:

```
move_iman_files_list_time-stamp_process-id.txt
```

- The **HSM\_move\_process\_shutdown\_time-stamp\_process-id.cfg** configuration file is located in the directory specified by the **\$IMAN\_LOG** variable.

To terminate the move process, edit this file such that the first line reads:

```
HSM_CONFIG_INFO=ON
```

#### EXAMPLES

- Enter the following command on a single line to generate the list of all files in Teamcenter Engineering volumes **vol001** and **vol002**:

```
move_iman_files -u=infodba -p=infodba -g=dba -mode=list
-source_vol=vol001~vol002
```

- Enter the following command on a single line to generate the list of all files in Teamcenter Engineering volumes **vol001** and **vol002** that were last accessed in the past 30 days:

```
move_iman_files -u=infodba -p=infodba -g=dba -mode=list
-source_vol=vol001~vol002 -atime=-30
```

- Enter the following command on a single line to generate the list of all files in Teamcenter Engineering volumes **vol001** and **vol002** that have not been accessed in the past 30 days:

```
move_iman_files -u=infodba -p=infodba -g=dba -mode=list
-source_vol=vol001~vol002 -atime=+30
```

- Enter the following command on a single line to generate the list of all files in Teamcenter Engineering volume **vol001** that have not been accessed in the past 30 days, along with the predetermined list of files contained in the file named **pre\_list.txt**, located in the current directory:

```
move_iman_files -u=infodba -p=infodba -g=dba -mode=list
-source_vol=vol001 -atime=-30 -list_file=pre_list.txt
```

- Enter the following command on a single line to relocate the Teamcenter Engineering data that corresponds to the **os\_file\_list.txt** list file to the destination volumes **newvol001** and **newvol002**:

```
move_iman_files -u=infodba -p=infodba -g=dba -mode=move
-dest_vol=newvol001~newvol002 -list_file=os_file_list.txt
```

- Enter the following command on a single line to relocate the Teamcenter Engineering data that was last accessed in the past 30 days from Teamcenter Engineering volumes **vol001** and **vol002** to the destination volumes **newvol001** and **newvol002**.

```
move_iman_files -u=infodba -p=infodba -g=dba
-mode=list_and_move -source_vol=vol001~vol002
-dest_vol=newvol001~newvol002 -atime=-30
```

- Enter the following command on a single line to relocate the Teamcenter Engineering data that was last accessed in past 30 days from Teamcenter Engineering volumes **vol001** and **vol002**, along with the predetermined list of files contained in the **pre\_list.txt** file, located in the current directory, to the destination volumes **newvol001** and **newvol002**:

```
move_iman_files -u=infodba -p=infodba -g=dba
-mode=list_and_move -source_vol=vol001~vol002
-dest_vol=newvol001~newvol002 -atime=-30 -list_file=pre_list.txt
```

---

**purge\_datasets**

---

**DESCRIPTION**

Removes (purges) old versions of datasets from the database. Normally, Teamcenter Engineering 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 *Configuration Guide*. However, certain conditions, for example, when a user does not have permission to purge a dataset owned by another user and a group is given read/write permission but not delete permission, can prevent automatic purging of old datasets.

A listing is produced that shows each dataset purged, along with the owning user and group.

**SYNTAX**

```
purge_datasets -u=infodba -p=password -g=dba -b=beginning_anchor  
-e=ending_anchor -k=keep-limit -set [-report] [-replica_only  
-site=site-name] [-h]
```

**ARGUMENTS****-u**

Specifies the user ID. This utility must be run by the **infodba** user. See restriction #1.

**-p**

Specifies the password associated with the **infodba** user account.

**-g**

Specifies the group associated with the **infodba** user. Must be the **dba** group.

**-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.

**-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.

**ENVIRONMENT**

As specified in *Manually Setting the Teamcenter Engineering Environment* in chapter 1, *Introduction*.

**FILES**

As specified in *Log Files* in chapter 1, *Introduction*.

**RESTRICTIONS**

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:

```
$IMAN_BIN/purge_datasets -u=infodba -p=password -g=dba -b=beginning-anchor
-e=ending-anchor -k=keep-limit -set
```

---

**purge\_mirror\_volumes**

---

**DESCRIPTION**

Purges mirrored volumes after the Teamcenter Engineering database is backed up or recovered.

**SYNTAX**

**purge\_mirror\_volumes -u=user-id -p=password -g=group -f=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.



If Teamcenter 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 Engineering database. If you do not supply these arguments, the utility attempts 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-name* 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.

**-f**

Specifies the name of the log file that contains the results of the purge mirror session.

**-v**

Runs the utility in verbose mode to display the maximum amount of information. Typically, nonverbose utility sessions only display error messages.

**-h**

Displays help for this utility.

**ENVIRONMENT**

As specified in [Manually Setting the Teamcenter Engineering Environment](#) in chapter 1, [Introduction](#).

Mirror volumes and the **purge\_mirror\_volumes** utility are only used with TCFS.

**FILES**

As specified in [Log Files](#) in chapter 1, [Introduction](#).



**RESTRICTIONS**

None.

**EXAMPLES**

None.

---

**purge\_volumes**

---

**DESCRIPTION**

Removes (purges) operating system files that represent deleted Teamcenter Engineering objects. The **purge\_volumes** utility can be run interactively, in forced execution mode, or periodically.

During a Teamcenter Engineering session, users delete objects. However, if the user deleting the object does not have the necessary operating system privileges to delete the associated operating system file along with the Teamcenter Engineering object, the file remains in the Teamcenter Engineering volume. The **purge\_volumes** utility unlocks these files so that they can be deleted at the operating system level.

**SYNTAX**

**purge\_volumes -u=user-id -p=password -g=group [-f] [-stime]**

**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.



If Teamcenter 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 Engineering database. If you do not supply these arguments, the utility attempts 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-name* 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.

**-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 does not prompt 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.

**ENVIRONMENT**

As specified in *Manually Setting the Teamcenter Engineering Environment* in chapter 1.

**FILES**

As specified in *Log Files* in chapter 1, *Introduction*.

**RESTRICTIONS**

1. The **purge\_volumes** utility must be run by a privileged user (for example, the **root** user on UNIX).
2. 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.
3. If the **purge\_volumes** is run on a node other than the volume server or if volumes reside on more than one server, ensure that the volumes are exported with **root** privilege to the node that is used to run the **purge\_volumes** utility. This ensures that **root** retains privilege on the NFS-mounted file system. Otherwise, the **purge\_volumes** utility fails to delete designated files from the operating system.

**EXAMPLES**

To run the **purge\_volumes** utility interactively, enter the following command on a single line:

```
$IMAN_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
```

**Running the purge\_volumes Utility at Boot Time**

The **purge\_volumes** utility can be run each time the system is booted in order to recover disk space. You may find this to be a valuable practice. In order to run the **purge\_volumes** utility at boot time on UNIX platforms, add the following line to the **/etc/rc.ed**s file:

```
purge_volumes -u=infodba -p=password -g=dba -f
```

This starts **purge\_volumes** at boot time and disables confirmation of each delete action.

---

**report\_volume**

---

**DESCRIPTION**

Lists the operating system path of all existing Teamcenter Engineering volumes. The path does not include the network node name.

**SYNTAX**

**report\_volume** **-u**=*user-id* **-p**=*password* **-g**=*group* **-f**=*file-name*  
**[-date=***yymmddhhmmss***]**

**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.



If Teamcenter 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 Engineering database. If you do not supply these arguments, the utility attempts 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-name* 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.

**-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.

**ENVIRONMENT**

As specified in [Manually Setting the Teamcenter Engineering Environment](#) in chapter 1, [Introduction](#).

**FILES**

As specified in [Log Files](#) in chapter 1, [Introduction](#).

**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:

```
$IMAN_ROOT/bin/report_volume -u=infodba -p=infodba -g=dba -f=out
```

---

**review\_volumes**

---

**DESCRIPTION**

Allows you to view detailed information about Teamcenter Engineering volumes 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 Engineering 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. Simply remove any file names that you do not want to delete before using the report file as input.

You can also save any deleted files to a ZIP format compressed file.

**SYNTAX**

**review\_volumes** **-u**=*user-id* **-p**=*password* **-g**=*group* **-v**=*volume* **-rf**= *file-name*  
| **-if**=*file-name* [**-of**=*file-name*] [**-zf**=*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.



If Teamcenter 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 Engineering database. If you do not supply these arguments, the utility attempts 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-name* 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.

**-v**

Runs the utility against a specified volume. If not supplied, the utility is run against all volumes at the site.

**-rf**

Creates a report file listing unreferenced files in volumes.

**-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 but is optional with the **-rf** argument.

**-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.

**-h**

Displays help for this utility.

**ENVIRONMENT**

As specified in [Manually Setting the Teamcenter Engineering Environment](#) in chapter 1, [Introduction](#).

**FILES**

As specified in [Log Files](#) in chapter 1, [Introduction](#).

**RESTRICTIONS**

1. Requires Teamcenter Engineering administrator privileges.
2. May also require operating system **root** (UNIX) or **administrator** (Windows NT) permission to delete some files.

**EXAMPLES**

- To generate a report on a single volume, enter the following command on a single line:

```
$IMAN_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:

```
$IMAN_ROOT/bin/review_volumes -u=user-id -p=password -g=group
-rf=file-name -of=file-name
```

- To delete files on all volumes from a previously executed report, enter the following on a single line:

```
$IMAN_ROOT/bin/review_volumes -u=user-id -p=password -g=group
-if=file-name -of=file-name
```

## **File Management System (FMS) Utilities**

This section describes utilities used to maintain the File Management System.



---

**fscadmin.sh/.bat**


---

**DESCRIPTION**

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 [-kkeyfile] [-sserveraddr ] [command]**

**ARGUMENTS**

**-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:port ./status**

Default: **http://127.0.0.1:4444**

**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:port ./status**

Example: **fscadmin -s http://myserver:port ./log**

Default: **./status.**

**ENVIRONMENT**

- The **FMS\_HOME** variable must be set to a valid FMS server directory.
- The **JAVA\_HOME** variable must be set to a valid JDK 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

- FSCs do not forward requests to destination FSCs; therefore, you must specify the server 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
<b>cachesummary</b>	Summary of the read and write caches (number of files, bytes, hits, misses).
<b>cachesummary/read</b>	Summary of the read cache.
<b>cachesummary/write</b>	Summary of the write cache.
<b>cachedetail</b>	Summary and detail of the read and write caches (guids, filesizes).
<b>cachedetail/read</b>	Summary and detail of the read cache.
<b>cachedetail/write</b>	Summary and detail of the write cache.
<b>config</b>	Dumps the configuration for this FSC.
<b>filestoresummary</b>	Summary of all filestores (volumeid, root path, files, dirs, bytes).
<b>filestoresummary/myvol</b>	Summary of a specific filestore.
<b>filestoredetail</b>	Summary and detail of all filestores (filename, len, last modified, last accessed).
<b>filestoredetail/myvol</b>	Summary and detail of a specific filestore.
<b>clearcache</b>	Clears (empties) both read and write caches.
<b>clearcache/read</b>	Clears the read cache.
<b>clearcache/write</b>	Clears the write cache.

FUNCTION[/SUBFUNCTION]	Description
<b>purgecache</b>	Purges the caches, reclaiming disk space.
<b>purgecache/read</b>	Purges the read cache.
<b>purgecache/write</b>	Purges the write cache.
<b>purgeguid/xxx</b>	Removes a specific guid (file) from the cache.
<b>perfcounters</b>	Performance counters
<b>perfcounters/reset</b>	Resets the performance counters.
<b>loglevel/[fatal   error   warn   info   debug]</b>	Sets the log level.
<b>loglevel/logger/[fatal   error   warn   info   debug]</b>	Sets error log level on the logger and all children. Logger is in the form of the name of the <b>class.package</b> of which you wish to change the loglevel.  <b>com.teamcenter.fms.servercache</b> is all of the server <b>com.teamcenter.fms.servercache.FileHandleCacheManager</b> sets only the loglevel on that class.
<b>log</b>	Dumps the current logfile contents.
<b>status</b>	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.
<b>shutdown</b>	Stops the FSC when it becomes idle (when all current file transfers are complete). The FSC will reject all new incoming file requests.

<b>FUNCTION[/SUBFUNCTION]</b>	<b>Description</b>
<b>shutdown/maxwait/XXX</b>	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, XXX is the number of seconds to wait before forcing the shutdown.
<b>shutdown/now</b>	Stops the FSC.
<b>stop</b>	Stops the FSC.
<b>useragents</b>	Prints a tally of all the useragents (clients) that have connected.
<b>version</b>	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:port ./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:port ./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:port ./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:port ./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:port ./shutdown/maxwait/3600
```

### Example output:

```
FSC will shutdown when idle, or in 3600 seconds...
```

---

**fccstat**

---

**DESCRIPTION**

Monitors the local FMS cache (FCC).

**SYNTAX**

**\$FMS\_HOME/fccstat**

**ARGUMENTS****none**

Prints whether FCC is online, displays FCC versions and run time if running.

**-help (or) -h (or) -?**

Displays help for this utility. Help may be localized if the FCC is running.

**-x**

Prints FCC cache statistics summary, including whole file read, whole file write, and segment cache statistics.

**-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.

**-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.

**-remove {UID}**

Purges only the requested UID from the FCC cache.

**-restart**

Stops (if running) and restarts the FCC, effectively reloading the configuration. Environment variables still override any configuration file settings or changes.

**-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 Setting the Teamcenter Engineering Environment* in chapter 1, *Introduction* 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 JDK directory.

**FILES**

As specified in *Log Files* in chapter 1, *Introduction*.

**RESTRICTIONS**

None.

**EXAMPLES**

- To determine whether the FCC is running, enter the following command on a single line:

```
D:\apps\UGS\Teamcenter\Engineering\2005\fms\bin>fccstat
```

Example output:

```
fccstat: FCC offline.
```

- To display help for this utility and a statistic summary, enter the following command on a single line:

```
D:\apps\UGS\Teamcenter\Engineering\2005\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, enter the following command on a single line:

```
D:\apps\UGS\Teamcenter\Engineering\2005\fms\bin>fccstat -config -clear
```

**Example output:**

```
fccstat -config:
Z:\pkmvobl\fms\build\install\fms/fcc.xml
fccstat -clear:
Cache cleared.
```

- To purge a UID from the cache, enter the following command on a single line:

```
D:\apps\UGS\Teamcenter\Engineering\2005\fms\bin>fccstat
-remove abcd1234fedc9876ef005678ba005432
```

**Example output:**

```
fccstat -remove:
UID abcd1234fedc9876ef005678ba005432 has been removed from the FCC Cache.
```

- To stop the FCC, enter the following command on a single line:

```
D:\apps\UGS\Teamcenter\Engineering\2005\fms\bin>fccstat -stop
```

**Example output:**

```
fccstat -stop:
FCC Stopped.
```

- To restart the FCC, enter the following command on a single line:

```
D:\apps\UGS\Teamcenter\Engineering\2005\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, enter the following command on a single line:

```
D:\apps\UGS\Teamcenter\Engineering\2005\fms\bin>fccstat -stop
```

**Example output:**

```
fccstat -stop:
FCC Stopped.
```

- To kill the FCC, enter the following command on a single line:

```
D:\apps\UGS\Teamcenter\Engineering\2005\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, enter the following command on a single line:

```
D:\apps\UGS\Teamcenter\Engineering\2005\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/'
```

---

**install\_encryptionkeys**

---

**DESCRIPTION**

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-name* **-p**=*password* **-g**=*group-name*  
**-f**=**install** | **list** | **modify** | **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.



If Teamcenter 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 Engineering database. If you do not supply these arguments, the utility attempts 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-name* 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.

**-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.

**ENVIRONMENT**

As specified in *Manually Setting the Teamcenter Engineering Environment* in chapter 1, *Introduction*.

**FILES**

As specified in *Log Files* in chapter 1, *Introduction*.

**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
```



---

# Index

## A

Access Manager utilities . . . . . 2-6  
Action rules . . . . . 2-27  
Adobe Acrobat Reader . . . . . 12  
am\_install\_tree utility . . . . . 2-7  
Appearance Configuration utilities  
    appr\_update\_console . . . . . 3-4  
    appr\_update\_manager . . . . . 3-2  
    appr\_update\_supervisor . . . . . 3-3  
    create\_appearances . . . . . 3-10  
    purge\_baselined\_item\_revisions . . . . . 3-12  
appearance\_updater utility . . . . . 3-7  
Appearances  
    Finding . . . . . 9-9  
apply\_naming\_rule utility . . . . . 2-10  
appr\_update\_console utility . . . . . 3-4  
appr\_update\_manager utility . . . . . 3-2  
appr\_update\_supervisor utility . . . . . 3-3  
Archive/Restore utilities  
    archive\_object . . . . . 10-26  
    auto\_archive . . . . . 10-20  
    auto\_restore . . . . . 10-23  
    object\_restore . . . . . 10-28  
Attribute mapping  
    tc\_config\_attr\_mapping utility . . . . . 2-42  
    Teamcenter to external attributes . . . . . 2-42  
Audit Manager utilities  
    audit\_archive . . . . . 10-31  
    combine\_audit\_files . . . . . 10-33  
    define\_auditdefs . . . . . 10-36  
audit\_archive utility . . . . . 10-31  
auto\_archive utility . . . . . 10-20  
auto\_restore utility . . . . . 10-23

## B

Backup and Recovery utilities  
    backup\_modes . . . . . 10-39  
    backup\_xmlinfo . . . . . 10-42  
    object\_backup . . . . . 10-44  
    object\_recover . . . . . 10-46  
    sfr\_instances . . . . . 10-48  
backup\_modes utility . . . . . 10-39  
backup\_xmlinfo utility . . . . . 10-42

Baseline revision, purging . . . . . 3-12  
Batch meshing utilities  
    epm\_import\_batch\_meshing\_results . . . 14-5  
    epm\_notify\_batch\_meshing\_results . . . 14-7  
Batch mode reporting . . . . . 2-53  
batch\_export\_translate\_import utility . . . 5-2  
batchmode\_clearance\_analysis.pl . . . . . 3-39,  
    7-2  
bomwriter utility . . . . . 7-3  
build\_fts\_index utility . . . . . 9-2  
Business Modeler utilities  
    apply\_naming\_rule . . . . . 2-10  
    business\_rules\_dtdxml2plmxml . . . . . 2-16  
Business Modeler utilities  
    deepcopyrules\_migration . . . . . 2-18  
    grm . . . . . 2-19  
    import\_export\_business\_rules . . . . . 2-21  
    install\_bmf\_rules . . . . . 2-24  
    install\_type\_display\_rules . . . . . 2-26  
    migrate\_action\_rules\_to\_bmf . . . . . 2-27  
    migrate\_complex\_property\_rules . . . . . 2-35  
business\_rules\_dtdxml2plmxml  
    utility . . . . . 2-16

## C

CAE utilities  
    cae\_save\_result\_data . . . . . 14-2  
    epm\_import\_batch\_meshing\_results . . . 14-5  
    epm\_notify\_batch\_meshing\_results . . . 14-7  
cae\_save\_result\_data utility . . . . . 14-2  
cc\_writer utility . . . . . 7-7  
Change Management utilities . . . . . 2-36  
Classification utilities  
    icsutility . . . . . 8-10  
    icsxml . . . . . 8-6  
    smlutility . . . . . 8-2  
clear\_process\_stage\_list utility . . . . . 4-2  
Clearance analysis . . . . . 3-39, 7-2  
clearlocks utility . . . . . 10-79  
Code conventions . . . . . 10  
collect\_garbage utility . . . . . 16-30  
combine\_audit\_files utility . . . . . 10-33  
Command line entry conventions . . . . . 10  
compare Ug\_and\_iman\_ps utility . . . . . 3-41

---

## Index

Configuration utilities . . . . . 2-1

Conventions

- Caution icons . . . . . 9
- Code . . . . . 10
- Command line entries . . . . . 10
- File contents . . . . . 10
- Names . . . . . 9
- Note icons . . . . . 9
- Revisions . . . . . 9
- Syntax definitions . . . . . 11
- Values . . . . . 9
- Warning icons . . . . . 9

convert\_distribution\_lists Utility . . . . 10-60

convert\_forms utility . . . . . 6-2

Converting action rules to extension rules . . . . . 2-27

Converting distribution lists . . . . . 10-60

create\_appearances utility . . . . . 3-10

create\_change\_types utility . . . . . 2-37

create\_project utility . . . . . 10-66

create\_validationdata utility . . . . . 2-48

Creating external attribute mapping . . . 2-42

Creating FMS encryption keys . . . . . 16-70

Creating projects . . . . . 10-66

Customization utilities

- convert\_forms . . . . . 6-2
- taxonomy . . . . . 6-10
- tc\_erp\_schema . . . . . 6-8

**D**

Data sharing utilities

- batch\_export\_translate\_import . . . . . 5-2
- data\_share . . . . . 5-15
- data\_sync . . . . . 5-25
- database\_verify . . . . . 5-4
- dsa\_util . . . . . 5-6
- EMBulkloader . . . . . 5-9
- export\_recovery . . . . . 5-11
- idsminetd . . . . . 5-24
- import\_file . . . . . 5-35
- item\_export . . . . . 5-38
- item\_import . . . . . 5-44
- item\_relink . . . . . 5-47
- item\_rename . . . . . 5-51
- plmxml\_export . . . . . 5-54
- plmxml\_import . . . . . 5-57
- step\_export . . . . . 5-60
- step\_import . . . . . 5-62

data\_share utility . . . . . 5-15

data\_sync utility . . . . . 5-25

database\_verify utility . . . . . 5-4

dataset\_cleanup utility . . . . . 16-38

Datasets

- Find modified . . . . . 9-13

datasettype\_cleanup utility . . . . . 16-43

deepcopyrules\_migration utility . . . . . 2-18

default\_queries utility . . . . . 9-8

define\_auditdefs utility . . . . . 10-36

delete\_namedrevisions utility . . . . . 4-3

Deleting external attribute mapping . . . 2-42

Deleting tasks with null tags . . . . . 4-3

Distribution list conversion . . . . . 10-60

Documentation . . . . . 12

- Online help . . . . . 12
- Printable files . . . . . 12

dsa\_util utility . . . . . 5-6

## E

Effectivity mode utilities

- effupgrade . . . . . 3-36

effupgrade utility . . . . . 3-36

Embedded Software Manager . . . . . 15-2

EMBulkloader utility . . . . . 5-9

Engineering Translation Services

- Activating ETS registry . . . . . 10-52
- Creating request objects . . . . . 10-57
- Managing services . . . . . 10-54
- Updating schema . . . . . 10-51

Engineering Translation Services utilities . . . . . 10-50

- ets\_create\_rqst . . . . . 10-57
- ETSDBConfig . . . . . 10-51
- etsServiceManager . . . . . 10-54
- etsServiceRegistry . . . . . 10-52

Environment variables

- IMAN\_DATA . . . . . 1-1

epm\_import\_batch\_meshing\_results utility . . . . . 14-5

epm\_notify\_batch\_meshing\_results utility . . . . . 14-7

Error 100228 . . . . . 5-34

ets\_create\_rqst utility . . . . . 10-57

ETSDBConfig utility . . . . . 10-51

etsServiceManager utility . . . . . 10-54

etsServiceRegistry utility . . . . . 10-52

export\_attr\_mappings utility . . . . . 11-2

export\_recovery utility . . . . . 5-11

Extension rules . . . . . 2-27

Extensions . . . . . 2-24

## F

fccstat utility . . . . . 16-66

fcsadmin utility . . . . . 16-61

File contents conventions . . . . . 10  
 File Management System (FMS)  
   Utilities . . . . . 16-61, 16-66  
     install\_encryptionkeys . . . . . 16-70  
 find\_appearances utility . . . . . 9-9  
 find\_modified\_datasets utility . . . . . 9-13  
 find\_processes utility . . . . . 4-4  
 find\_recently\_saved\_item\_rev utility . . . 9-15  
 find\_released\_item\_rev utility . . . . . 9-17  
 FMS encryption keys . . . . . 16-70

## G

generate\_iman\_ps\_path utility . . . . . 3-15  
 get\_qpl\_harvester\_assemblies utility . . . 7-21  
 global\_transfer utility . . . . . 4-6  
 GM Overlay subscription events . . . . . 13-15  
 gmo\_create\_material\_form\_templates . . . 13-12  
 gmo\_ipvbom\_export utility . . . . . 13-8  
 gmo\_ipvbom\_import utility . . . . . 13-6  
 grm utility . . . . . 2-19

## H

harvester\_jt.pl utility . . . . . 7-12  
 harvester.pl utility . . . . . 7-9

## I

Icon conventions . . . . . 9  
 icsutility utility . . . . . 8-10  
 icxml utility . . . . . 8-6  
 idsminetd utility . . . . . 5-24  
 IMAN\_DATA environment variable . . . . . 1-1  
 iman\_dbconfig utility . . . . . 16-2  
 imanhelp utility . . . . . 10-83  
 import\_attr\_mappings utility . . . . . 11-3  
 import\_export\_business\_rules utility . . . 2-21  
 import\_file utility . . . . . 5-35  
 Importing NX CAM templates . . . . . 5-64  
 Importing workflow templates . . . . . 5-57  
 index\_verifier utility . . . . . 16-45  
 input\_notetype\_default utility . . . . . 3-17  
 install utility . . . . . 10-2  
 install\_bmf\_rules utility . . . . . 2-24  
 install\_default\_report\_designs utility . . . 2-51  
 install\_encryptionkeys utility . . . . . 16-70  
 install\_esm\_software\_types utility . . . . . 15-2  
 install\_event\_types utility . . . . . 10-86  
 install\_handlers utility . . . . . 4-14  
 install\_kbl utility . . . . . 15-4  
 install\_lovs utility . . . . . 16-3  
 install\_type\_display\_rules utility . . . . . 2-26  
 install\_types utility . . . . . 16-5  
 Installation utilities

  tem . . . . . 10-16  
 Installing embedded software types . . . . 15-2  
 Installing extensions . . . . . 2-24  
 Installing report designs . . . . . 2-51  
 Integration utilities  
   rdv\_context\_download . . . . . 12-2  
   tcc\_context\_upload . . . . . 12-7  
 item\_export utility . . . . . 5-38  
 item\_import utility . . . . . 5-44  
 item\_relink utility . . . . . 5-47  
 item\_rename utility . . . . . 5-51

## L

LDAP synchronization . . . . . 2-45  
 list\_users utility . . . . . 10-90  
 Location of program files . . . . . 1-1  
 Log file best practices . . . . . 1-3

## M

Maintenance utilities . . . . . 10-1  
 make\_datasettype utility . . . . . 16-10  
 make\_user utility . . . . . 16-14  
 Manual set . . . . . 12  
 Mapping to external attributes . . . . . 2-42  
 Mechatronics utilities . . . . . 15-8  
   install\_esm\_software\_types utility . . . 15-2  
   install\_kbl utility . . . . . 15-4  
   update\_gde\_types . . . . . 15-5  
 migrate\_action\_rules\_to\_bmf utility . . . . 2-27  
 migrate\_alias utility . . . . . 2-29  
 migrate\_complex\_property\_rules  
   utility . . . . . 2-35  
 migrate\_type\_display\_prefs utility . . . . . 10-62  
 Migrating  
   Workflow distribution lists . . . . . 13-17  
 Migrating GM Overlay subscription  
   events . . . . . 13-15  
 Migrating Microsoft Office forms . . . . . 10-64  
 Migration utilities  
   convert\_distribution\_lists . . . . . 10-60  
   migrate\_action\_rules\_to\_bmf . . . . . 2-27  
   migrate\_alias . . . . . 2-29  
   migrate\_type\_display\_prefs . . . . . 10-62  
   move\_mso\_forms . . . . . 10-64  
 Monitoring FMS FSC Servers . . . . . 16-61  
 Monitoring local FMS cache . . . . . 16-66  
 move\_iman\_files utility . . . . . 16-47  
 move\_mso\_forms utility . . . . . 10-64

## N

Name conventions . . . . . 9  
 NX CAM templates, importing . . . . . 5-64

---

## Index

NX Manager utilities  
  export\_attr\_mappings . . . . . 11-2  
  import\_attr\_mappings . . . . . 11-3  
  refile\_info . . . . . 11-4  
  reset\_protection . . . . . 11-5

## O

Object validation utilities . . . . . 2-47  
Object validation utilities,  
  create\_validationdata . . . . . 2-48  
object\_archive utility . . . . . 10-26  
object\_backup utility . . . . . 10-44  
object\_recover utility . . . . . 10-46  
object\_restore utility . . . . . 10-28  
Online help . . . . . 12  
Organization utilities, ldapsync . . . . . 2-45

## P

plmxml\_export utility . . . . . 5-54  
plmxml\_import utility . . . . . 5-57  
Preferences Manager utilities . . . . . 2-1  
preferences\_manager utility . . . . . 2-2  
Product structure comparison utilities  
  compare Ug\_and\_Iman\_ps . . . . . 3-41  
  ugmanager\_compare\_ps . . . . . 3-43  
Product structure maintenance utilities  
  generate\_Iman\_ps\_path . . . . . 3-15  
  input\_notetype\_default . . . . . 3-17  
  ps\_rename\_bvrs . . . . . 3-19  
  ps\_traverse . . . . . 3-21  
  update\_bomchanges . . . . . 3-29  
  update\_project\_bom . . . . . 3-31, 10-69  
  upgrade\_rev\_rules . . . . . 3-34  
Program files . . . . . 1-1  
Project utilities . . . . . 10-65  
Project utilities, create\_project . . . . . 10-66  
proxy\_sync utility . . . . . 12-13  
ps\_rename\_bvrs utility . . . . . 3-19  
ps\_traverse utility . . . . . 3-21  
ps\_upload utility . . . . . 3-25  
purge\_baselined\_item\_revisions  
  utility . . . . . 3-12  
purge\_datasets utility . . . . . 16-50  
purge\_file\_cache utility . . . . . 10-91  
purge\_invalid\_subscriptions utility . . . . . 10-76  
purge\_mirror\_volumes utility . . . . . 16-52  
purge\_processes utility . . . . . 4-12  
purge\_volumes utility . . . . . 16-54  
Purging  
  Baselined item revisions . . . . . 3-12  
  Datasets . . . . . 16-50  
  File cache . . . . . 10-91  
  Invalid subscriptions . . . . . 10-76

Mirrored volumes . . . . . 16-52  
Volumes . . . . . 16-54  
Workflow processes . . . . . 4-12

## Q

Query utilities  
  build\_fts\_index . . . . . 9-2  
  default\_queries . . . . . 9-8  
  find\_appearances . . . . . 9-9  
  find\_modified\_datasets . . . . . 9-13  
  find\_recently\_saved\_item\_rev . . . . . 9-15  
  find\_released\_item\_rev . . . . . 9-17  
  query\_xml . . . . . 9-19  
query\_xml utility . . . . . 9-19

## R

RDV cache maintenance . . . . . 7-20  
RDV utilities  
  bomwriter . . . . . 7-3  
  Cache maintenance . . . . . 7-20  
  cc\_writer . . . . . 7-7  
  get\_qpl\_harvester\_assemblies . . . . . 7-21  
  harvester\_jt.pl . . . . . 7-12  
  harvester.pl . . . . . 7-9  
  released\_parts\_collector utility . . . . . 7-26  
rdv\_context\_download utility . . . . . 12-2  
refile\_info utility . . . . . 11-4  
release\_man utility . . . . . 4-16  
released\_parts\_collector utility . . . . . 7-26  
rep\_batch\_report utility . . . . . 2-53  
Report Designer utilities . . . . . 2-50  
Report designs, installing . . . . . 2-51  
report\_volume utility . . . . . 16-56  
Reporting utilities  
  rep\_batch\_report . . . . . 2-53  
reset\_protection utility . . . . . 11-5  
reset\_user\_home\_folder utility . . . . . 10-93  
review\_volumes utility . . . . . 16-58  
Revision conventions . . . . . 9  
runJtCacheClean utility . . . . . 2-57  
runJtCacheInit utility . . . . . 2-58  
runJtCachePopulator utility . . . . . 2-59

## S

Schema Browser (Character Mode)  
  utility . . . . . 16-20  
sfr\_instances utility . . . . . 10-48  
site\_util utility . . . . . 10-13  
smlutility utility . . . . . 8-2  
step\_export utility . . . . . 5-60  
step\_import utility . . . . . 5-62  
Subscription Manager utilities . . . . . 10-75



Synchronizing user data . . . . . 2-45  
 Syntax definition conventions . . . . . 11  
 System maintenance utilities  
   clearlocks . . . . . 10-79  
   imanhelp . . . . . 10-83  
   install . . . . . 10-2  
   install\_event\_types . . . . . 10-86  
   list\_users . . . . . 10-90  
   purge\_file\_cache . . . . . 10-91  
   reset\_user\_home\_folder . . . . . 10-93  
   site\_util . . . . . 10-13  
   tc\_mail\_smtp . . . . . 10-84

## T

taxonomy utility . . . . . 6-10  
 tc\_config\_attr\_mapping utility . . . . . 2-42  
 tc\_erp\_schema utility . . . . . 6-8  
 tc\_mail\_smtp utility . . . . . 10-84  
 tc\_workflow\_postprocess utility . . . . . 4-8  
 tcc\_context\_upload utility . . . . . 12-7  
 TcEnggWolfAppRegistryUtil utility . . . 2-61  
 Teamcenter Automotive Edition–GM Overlay  
   utilities  
     gmo\_create\_material\_form\_templates . . 13-12  
     gmo\_ipvbom\_export . . . . . 13-8  
     gmo\_ipvbom\_import . . . . . 13-6  
 Teamcenter Engineering Data Integration  
   Services Adapter utilities  
     runJtCacheClean . . . . . 2-57  
     runJtCacheInit . . . . . 2-58  
     runJtCachePopulator . . . . . 2-59  
 Teamcenter Engineering environment  
   settings . . . . . 1-1  
 Teamcenter Linking utilities . . . . . 2-60  
 tem utility . . . . . 10-16

## U

ugmanager\_compare\_ps utility . . . . . 3-43  
 ugmgr\_upgrade\_bvrsyncform utility . . . 11-8  
 ugmgr\_upgrade\_transforms utility . . . 11-10  
 uih\_to\_xml utility . . . . . 10-17  
 update\_bomchanges utility . . . . . 3-29  
 update\_gde\_types utility . . . . . 15-5  
 update\_project\_bom utility . . . . . 3-31, 10-69  
 update\_project\_data utility . . . . . 10-72  
 Updating BOM items in projects . . . . . 3-31,  
   10-69  
 upgrade\_motif\_workflow\_templates  
   utility . . . . . 4-18  
 upgrade\_process\_template utility . . . . 4-18

upgrade\_rev\_rules utility . . . . . 3-34  
 upgrade\_to\_group\_hierarchy utility . . . 16-26  
 upgrade\_workflow\_objects Utility . . . . 4-20  
 Utilities  
   ugmgr\_upgrade\_bvrsyncform . . . . . 11-8  
   ugmgr\_upgrade\_transforms . . . . . 11-10

## V

Value conventions . . . . . 9  
 Verification verdicts . . . . . 5-26  
 verify\_tasks utility . . . . . 4-22  
 Volume and database management utilities  
   iman\_dbconfig . . . . . 16-2  
   install\_lovs . . . . . 16-3  
   install\_types . . . . . 16-5  
   make\_datasettype . . . . . 16-10  
   Schema Browser (Character  
     Mode) . . . . . 16-20  
   upgrade\_to\_group\_hierarchy . . . . . 16-26  
   xml\_validator . . . . . 16-29  
 Volume and database utilities  
   collect\_garbage . . . . . 16-30  
   dataset\_cleanup . . . . . 16-38  
   datasettype\_cleanup . . . . . 16-43  
   index\_verifier . . . . . 16-45  
   move\_iman\_files . . . . . 16-47  
   purge\_datasets . . . . . 16-50  
   purge\_mirror\_volumes . . . . . 16-52  
   purge\_volumes . . . . . 16-54  
   report\_volume . . . . . 16-56  
   review\_volumes . . . . . 16-58

## W

Workflow distribution lists  
   Migrating . . . . . 13-17  
 Workflow templates, importing . . . . . 5-57  
 Workflow utilities . . . . . 4-3  
   clear\_process\_stage\_list . . . . . 4-2  
   find\_processes . . . . . 4-4  
   global\_transfer . . . . . 4-6  
   purge\_processes . . . . . 4-12  
   release\_man . . . . . 4-16  
   tc\_workflow\_postprocess . . . . . 4-8  
   upgrade\_motif\_workflow\_templates . . . 4-18  
   upgrade\_process\_template . . . . . 4-18  
   verify\_tasks . . . . . 4-22

## X

xml\_validator utility . . . . . 16-29