serena.com

SERENA®
DIMENSIONS® CM10.1.1

Command-Line Reference

Serena Proprietary and Confidential Information

SERENA™

Part number: MA-CMCLR-002

Publication date: April 2007

Table of Contents

Welcome to Serena [Product Name Short No Marks]

Thank you for choosing Serena® Dimensions® CM, the configuration management component of Dimensions. [Product Name Short No Marks] is a powerful process management and change control system that will revolutionize the way you develop software. [Product Name Short No Marks] helps you organize, manage, and protect your software development projects on every level—from storing and tracking changes to individual files, to managing and monitoring an entire development cycle.

About Serena Dimensions

[Product Name Short No Marks] is a principal component of the integrated components that constitute the Serena Dimensions product. Other components include:

- Serena Dimensions RM, which offers full requirements management and traceability throughout the development lifecycle by centralizing and organizing requirements using role base views and a user configurable requirements process.
- Serena Command Center, which provides development project monitoring.

In this book, the term *Dimensions* is used in the context of Dimensions CM rather than Dimensions RM or Command Center.

For more information

Refer to the *Introduction to Dimensions CM* manual for a description of the [Product Name Short No Marks] documentation set, a summary of the ways to work with [Product Name Short No Marks], and instructions for accessing the online help.

Edition status

The information in this guide applies to *Release 10.1.0 of Serena*® Dimensions® CM. This edition supersedes earlier editions of this manual.

Acknowledgment

Some of the information contained in this document regarding the Oracle Server and Architecture has been reproduced from the Oracle Product Set documentation with permission from Oracle Corporation.

Product Name Changes and New Terminology

Beginning with Serena® Dimensions® CM 10.1, product name changes took place and terminology changes were also introduced.

The following table details the product rebranding:

Current product name	Legacy product name
Serena Command Center[a]	n/a
Serena Dimensions[b]	
Serena Dimensions Build[c]	n/a
Serena [Product Name Short No Marks]	Serena ChangeMan Dimensions
Serena Dimensions RM	Serena RTM
Serena Mover	Serena ChangeMan Mover
Serena PVCS Version Manager	Serena ChangeMan Version Manager

a.New product.
b.Encompasses all Dimensions components.
c.New product.

The above product name changes have also led to changes in the [Product Name Short No Marks] documentation set. Refer to the *Introduction to Dimensions CM* manual for a comprehensive list of the [Product Name Short No Marks] documentation set.

The following table details the terminology changes:

Current terminology	Legacy terminology
project	workset
request	change document
inbox	pending list
custom list / request list	user list[a]
custom list / request list	custom list[b]
work area[c]	n/a
library cache area[c]	n/a
privilege[c]	n/a
deployment area	build area
deployment stage	build stage
user interface profile[c]	n/a

a.Desktop client.
b.Web client and Visual Studio .NET Add In.
c.New term. See the *User's Guide* and *Process Modeling Guide* for details of these new terms.

The latest Dimensions 10.1 documentation reflects the new terms, so if you are using a new Dimensions 10.1 component with a component from an earlier release of Dimensions (for example, the Migration Console in conjunction with Dimensions 9.1), you will need to bear in mind the terminology changes when working with the older components.

NOTE

Certain specialized features will continue to use legacy terminology, but the new terminology will be used when describing these features in the documentation. These features comprise:

- The [Product Name Short No Marks] Command-Line Interface (dmcli).
- The public [Product Name Short No Marks] C /C++ Developer's Toolkit and Java API (dmpmcli).
- The [Product Name Short No Marks] Data Interchange File Format (PDIFF).

Typographical Conventions

The following typographical conventions are used in the online manuals and online help. These typographical conventions are used to assist you when using the documentation; they are not meant to contradict or change any standard use of typographical conventions in the various product components or the host operating system.

italics	Introduces new terms that you may not be familiar with and occasionally indicates emphasis.
bold	Emphasizes important information and field names.
UPPERCASE	Indicates keys or key combinations that you can use. For example, press the ENTER key.
`monospace`	Indicates syntax examples, values that you specify, or results that you receive.
`monospace italics`	Indicates names that are placeholders for values you specify; for example, *`fileName`*.
`monospace bold`	Indicates the results of an executed command.
vertical bar |	Separates menus and their associated commands. For example, select File | Copy means to select Copy from the File menu. Also, indicates mutually exclusive choices in a command syntax line.
angle brackets <>	Indicates names that are placeholders for values that you specify; for example, `<file-name>`.
square brackets []	Indicates optional items. For example, in the following statement: `SELECT [DISTINCT]`, DISTINCT is an optional keyword.
. . .	Indicates command arguments that can have more than one value.

Printing Manuals

As part of your Dimensions license agreement, you may print and distribute as many copies of the Dimensions manuals as needed *for your internal use, so long as you maintain all copies in strict confidence and take all reasonable steps necessary to ensure that the manuals are not made available or disclosed to anyone who is not authorized to access Dimensions under your Dimensions license agreement*.

Contacting Technical Support

Serena provides technical support for all registered users of this product, including limited installation support for the first 30 days. If you need support after that time, contact Serena Support at the following URL and follow the instructions:

http://support.serena.com/

Language-specific technical support is available during local business hours. For all other hours, technical support is provided in English.

The Serena Support web page can also be used to:

- Report problems and ask questions.
- Obtain up-to-date technical support information, including that shared by our customers via the Web, automatic e-mail notification, newsgroups, and regional user groups.
- Access a knowledge base, which contains how-to information and allows you to search on keywords for technical bulletins.
- Download fix releases for your Serena products.

Chapter 1

Use of Command Mode

Introduction

NOTE The term z/OS in this manual covers both the z/OS 1.1 (or later) and OS/390 V2R10 (or later) operating systems.

Command-mode (available for the majority, but not all, Dimensions functions) is an efficient alternative to Dimensions GUI-based clients, **provided that you are familiar with both Dimensions and the product to be processed**. Command mode is particularly suited for situations where you wish to perform unattended bulk batch operations. (It should be noted that in batch mode, Dimensions sends you e-mail confirming the submission and completion of each operation. Mail subscription features, including the batching of message bodies into a single e-mail message, are available through the SUB, UMNR, USUB, DMNR, and LMNR commands.)

IMPORTANT! Command mode can be used in any of several ways, on either a Dimensions CM server or a Dimensions client. In all cases you must first log in to Dimensions—see "Running Commands from a Dimensions Client" on page 22 for details of Dimensions Client command-line operation/connection.

Several of the mechanisms detailed below are not supported by Dimensions for z/OS. Refer to the *Serena Dimensions for z/OS Users and Administrator's Guide* for further details.

- A single command may be preceded by DMCLI and entered at the operating system prompt.

NOTE Supported from USS for z/OS with some restrictions for credentials. Supported from MVS batch via DIM390B.

- A single command may be entered at the `Dimensions>` prompt that results from typing DMCLI at the operating system prompt.

NOTE Supported from USS for z/OS with some restrictions for credentials.

- A command may be placed on a line or several consecutive lines of a text file (see below), which is specified as the parameter in an CMD command. The file may contain any number of commands to be processed sequentially in a single batch job. However, the interactive functions (BI, and some uses of BC and UC) cannot be included.
- A single command may be entered at the Execute Command window in the Dimensions desktop client's Run interface.

IMPORTANT!
See also "Using Quoted Strings in Dimensions Commands" on page 18.

The following topics are covered in this chapter:

- Running Commands from a Dimensions Client

- Command mode syntax.
- Using quoted strings in Dimensions commands.
- Compound fields in Dimensions commands.
- Note on the selection of item revisions by default.
- Note on project assignment.
- Request attributes.
- Operating System Differences.
- Command-Line Logging and Usage Analysis.

This is then followed by:

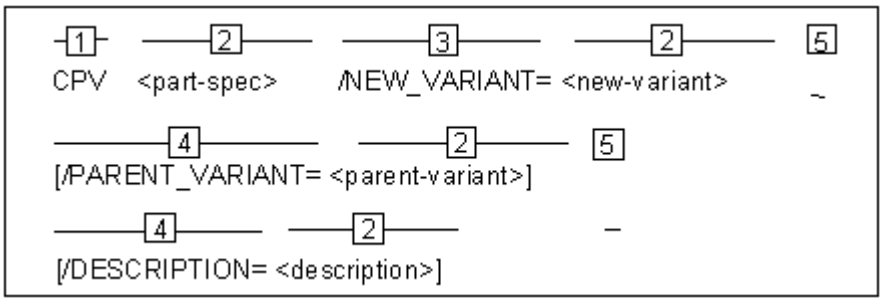
- Chapter 2, "Command Reference" on page 37, which covers most of the Dimensions commands, specifying and describing the parameters and qualifiers used by each one.
- Chapter 3, "Standalone Dimensions Utilities" on page 411, which identifies various miscellaneous Dimensions standalone utilities.

Command Mode Syntax

A command consists of a Dimensions function mnemonic, followed by parameters and qualifiers. A typical Dimensions command looks like this:

```
CPV SOMEPROD:"RELEASE MANAGEMENT".AAAA /NEW_VAR=IBM -
/DESC="Release Support - IBM Version"
```

The basis for coding each command is the **syntax diagram**, and there is a different one for each mnemonic. This is the syntax diagram for **CPV** (Create Design Part Variant):



The meaning of each part of the diagram is as follows:

1 The *function mnemonic* identifies which Dimensions function is to be performed.

2 A parameter is indicated by lower-case letters enclosed in angle brackets. This shows where a variable value is to be substituted. Parameters which end in `–spec` denote compound fields, and the syntax for coding all the components of these is specified below in "Compound Fields in Dimensions Commands" on page 21.

An *ellipsis (…)* indicates that a list of any number of parameters may be specified, separated by commas and enclosed in parentheses, for example:

```
/CHANGE_DOC_IDS=(<request1>,<request2>,…)
```

If there is only one parameter in the list, the parentheses are not essential, for example:

```
/CHANGE=PROD_DR_25
```

After each comma separating the parameters in the list, one or more spaces are optional before the next parameter. Along with the spaces, if required, a continuation character can be included and a new line begun.

3 A *required qualifier* is coded as shown. It always begins with a forward slash (**/**) and usually ends with an equal sign (**=**), the latter indicating that a substituted parameter variable **must follow**. (A qualifier, which does not end with **=**, is complete in itself.)

4 An *optional qualifier* is indicated by being enclosed in square brackets ([ ]), and **may be omitted in certain circumstances** (as detailed in this reference). The square brackets themselves are never included in the Dimensions command.

All qualifiers (required and optional) may be abbreviated provided that no ambiguity is caused (e.g. /NEW_VAR= in the example above). Options are shown on consecutive lines **that have the same indentation**, with an underscored **or** at the start of the lower line. **Only one** of the lines so designated may be chosen.

5 A *continuation indicator* is shown as hyphen (–). This is the character normally used at the end of a line to indicate that a command is being continued on another line.

There must be at least one space between the last command character and the hyphen (or backslash), but there must not be any spaces between that and the end of the line.

Exceptions:

- *UNIX systems* (**only** if the command is being entered at the UNIX system prompt): a backslash (\ ) must be used instead of a hyphen to indicate continuation.
- *Windows system* (**only** if the command is being entered at the operating system prompt): there is no continuation available, the command must be entered on a single line and is limited to 256 characters maximum.

Using Quoted Strings in Dimensions Commands

If you want to include spaces, or any characters outside the standard set, in the substitution for a parameter variable, then the variable value, or the part of it that contains the non-standard characters, must be enclosed in double-quotation characters (" "), for example:

```
"RELEASE MANAGEMENT"
```

There are additional conventions that you must follow if you want to enter a Dimensions command, which includes a quoted string, at the operating system prompt:

Windows System Prompt

The syntax is identical to that used at the Dimensions prompt (except that no continuation line is available).

IMPORTANT! When you execute a command using the `-cmd` parameter of `dmcli`, each double quotation mark used for wrapping strings that contain spaces must be escaped with a backslash (\).

UNIX System Prompt

The double-quoted string must itself be enclosed in single-quotation characters (' '), for example

```
'PROD:"QUERY RELEASE".AAAA-SRC;2'
```

NOTE This alternative syntax is not shown in the remainder of this reference. **You must understand implicitly that it is required whenever the command is used in this way**.

The Escape Character

NOTE The escape character discussed here does not refer to the Esc key on your keyboard.

An escape character must precede any character in a parameter that might otherwise be interpreted as part of the syntax of the command. Such characters include:

'at'	@	Double quotation	"
Comma	,	Single quotation	'
Left parentheses	(	Forward slash	/
Right parentheses	)	Backslash	\
Newline	@n		

The default escape character is **@**, but the setting of the Dimensions symbol DM_ESCAPE_CHAR may be used to specify any alternative as the escape character. The Windows command line interface escape character is a backslash (****) and this **cannot** be changed.

For example, in UNIX, to set the attribute TITLE to:

```
The "at" symbol (@)
```

the command would be:

```
UC PROD_DC_17 -
/ATTR=(TITLE="The @"at@" symbol @(@@@)")
```

The same command submitted using *dmcli* from the Windows operating system prompt would be:

```
dmcli -cmd "UC PROD_DC_17 -
/ATTR=(TITLE=\"The @\"at@\" symbol @(@@@)\")"
```

A Perl script on Windows might have this:

```
my $command_string = "dmcli -cmd \"UC PROD_DC_17
     /ATTR=(TITLE=\\\"The \@\\\"at\@\\\" symbol \@(\@\@\@)\\\")\"";
```

!

Note that in a Windows batch file, you cannot include double quotation marks in a variable definition if the variable will be used in a command where quotation marks need to be escaped. For example, the following does not work:

```
set FILE1="C:\<dir-name-with-spaces>\<file-name>"
...
call dmcli ... %FILE1%
```

Instead, do it like this:

```
set FILE1=C:\<dir-name-with-spaces>\<file-name>
...
call dmcli ... \"%FILE1%\"
```

An example how to use the "@n" syntax for escaping newlines is given in the discussion of multiline attributes below.

Using Comments in Dimensions Command Files

As discussed on page 16, a number of Dimensions commands can be batched together in a text file that is used as input to the Dimensions CMD command (see "CMD – Execute Dimensions Command File" on page 100).

To aid readability of this text file, you can enter comment lines. You do this by putting an exclamation point as the first non-blank character of a line where a command could start – this means you *cannot* place the exclamation point in the middle of a set of continuation lines for a single Dimensions command.

Multivalue and Multiline Attributes

A multivalue attribute—such as *OPS* with valid values *Sun*, *HP*, *DEC* and *IBM*—would be handled in command mode as follows:

```
/ATTR=(OPS=["Sun","HP","DEC","IBM"])
```

A multiline attribute – such as *DOC* with value

```
Hello world
     First line
     Second line
```

would be handled in command mode as follows:

```
/ATTR=(DOC="Hello world@nFirst line@nSecond line")
```

Compound Fields in Dimensions Commands

NOTE In certain circumstances, it is possible to omit some fields when coding compound parameters. When doing so, follow these rules:

- The <product-id> and the colon (:) following it can never be omitted.
- Apart from <item-id>, which can be optional, the second field (<part-id>, <baseline-id> or <release-id>) is also always required.
- If the <item-id> field is omitted, no other punctuation is omitted with it.
- If any other field is omitted, the immediately preceding punctuation character is also omitted, e.g. dot (**.**) when omitting <variant>; hyphen (–) when omitting <item-type>; and semicolon (;) when omitting <pcs> or <revision>.

In the syntax diagrams there are five parameters, *<project-spec>*, *<part-spec>*, *<item-spec>*, *<baseline-spec>* and *<release-spec>*, where the required substitution is a set of values or fields in a specific syntax format. This syntax is as follows:

<project-spec>

This fully identifies a specific project and has the following syntax:

```
<product-id>:<project-id>
```

<part-spec>

This fully identifies a specific design part and has the following syntax:

```
<product-id>:<part-id>.<variant>;<pcs>
```

<item-spec>

This fully identifies a specific item and has the following syntax:

```
<product-id>:<item-id>.<variant>-<item-type>;<revision>
```

The <revision> field can optionally have the syntax:

```
<branch-id>#<version>
```

where <branch-id> identifies the development branch to which this item revision belongs, and <version> identifies its revision within this branch. For example:

```
PROD:"QUERY RELEASE".AAAA;maint#3
```

If the field is **not** of the above form (i.e. it does not contain the **#** character), then the entire field is the revision number, and the item revision is not in a named branch.

<baseline-spec>

This fully identifies a specific baseline and has the following syntax:

```
<product-id>:<baseline-id>
```

<release-spec>

This fully identifies a specific release and has the following syntax:

```
<product-id>:<release-id>
```

Examples

An example of <part-spec>, omitting <variant> and <pcs>:

```
PROD:"RELEASE MANAGEMENT"
```

An example of <item-spec>, omitting <item-id> and <variant>:

```
PROD:-SRC;1
```

Running Commands from a Dimensions Client

Running commands from a Dimensions client or server is exactly the same—both are accessed by running the command DMCLI. For a Dimensions client this replaces the former CMDCLIENT command. In the remainder of this reference *client* should be read to include *server* installations as well unless specifically mentioned otherwise.

In order to run DMCLI from a Dimensions client, a network connection needs to be established first before you can run DMCLI commands as described later in this chapter. This connection can be achieved in any of the ways described below. Examples of client commands are given on page 28.

Connection Processes for Windows Dimensions Clients

Connecting Using the Remote Login Dialog Box

Typing

```
dmcli
```

at the Windows Command Prompt will launch the same remote login dialog box as that used for the Dimensions desktop client (see the *User's Guide* for details).

Once successful connection has been established, the login dialog box will be dismissed and a

```
Dimensions>
```

prompt will be displayed in the Command Prompt window. You can now enter commands on the Dimensions client.

NOTE All the Windows connection mechanisms described below will also default to the above mechanism if they fail to successfully establish a connection.

Connecting from a Server Using the DMDB Environment Variable

When running Dimensions CM operations from the command line you will normally be required to set the DMDB variable, unless you access the command line through the Dimensions CM GUI login dialog in which case it will be set for you. The DMDB variable has to be set to the value:

```
<base_database_id>/<base_database_passwd>@<db_connection>
```

For example, in Dimensions CM for Windows:

```
set DMDB=intermediate/intermediate@dim10
```

Connecting Using the -con Command-Line Parameter

If you assign (or have already assigned) a "Previous Connections" name to the connection details in a remote login dialog box, then you can "fast track" the login process and directly access the `Dimensions>` prompt by typing

```
dmcli -con <connect-name> -user <user-id> -pass <password>
```

The following sub-section describes this mechanism in detail (also, if `-pass` is excluded, you will be prompted for it in a similar manner to that described below for a new connection).

NOTE This is the recommended connection mechanism.

Connecting Using the Command-Line

Syntax

```
dmcli
   [-con <connect-name>]
   -user <user-id>
   -pass <user-pswd>
   -host <server-name>
   -dbname <db-name>
   -dsn <dsn-name>
   -param <param-file>
   -file <cmd-file>
   -cmd <dm-cmd>
   -help
   -version
```

where:

`-con <connectname>` specifies either:

- an existing connection string associated with stored connection parameters (created either by an earlier invocation of this command or by use of the "Previous Connections" field in the remote login dialog box);
- a connection string to be created at this invocation of DMCLI.

In the first case, you would normally only use the `-con` and `-pass` parameters, as described in the previous sub-section.

In the second case, you either specify the connection parameters required on the same command line

```
dmcli -con <new-connect-name> -user <user-name> …
```

or you simply type

```
dmcli -con <new-connect-name>
```

upon which you will be prompted for the relevant connection details in turn:

```
A connection named "<new-connect-name>" was not found. A new
    connection will be created.
Please enter values as requested or Ctrl+D to cancel.
Host: <server-name>
<server-name> login: <user-id>
Password: <user-pswd>
Database: <db-name>
```

This connection is then saved to file and will be used the next time `–con <new-connect-name>` is specified, when you will be prompted only for your password.

NOTE Unless `-con` is specified, none of the other DMCLI parameters will be stored for future use.

`-user <user-id>` specifies the operating system user name of your account on the server.

`-pass <userpswd>` specifies the password of your operating system user name account on the server.

`-host <server-name>` specifies your host name of the server.

`-dbname <db-name>` specifies your database identifier.

`-dsn <dsn-name>` specifies the data source name for connecting to your remote database.

`-param <param-file>` specifies a file containing the above parameters. This file has a format similar to the following example:

```
-user dmsys
-pass xxx
-host server1
-dbname intermediate
-dsn PC50
```

`-file <cmd-file>` specifies a file containing several Dimensions commands to be executed.

`-cmd <dm-cmd>` specifies a single Dimensions command to be executed.

`-help` displays help for DMCLI (the syntax described here).

`-version` displays the Dimensions release version.

Further Detail The parameter file or connection parameters can be followed by a Dimensions command using the `-cmd` option or in a command file containing a list of Dimensions commands

each on a separate line using the `-file` option. If no command or command file is specified, then commands are read and executed from standard input until an EOF (CTRL+Z) character is detected, or the pseudo-command `exit` is encountered.

NOTE

- The parameter file and command file must be specified using the full directory path contained within double quotation characters (").
- If the command you are running contains double quotation marks in the qualifiers, you must wrap the entire command in double quotations, and 'escape' the double quotation marks in the command. For example:

```
dmcli -user dmsys -pass dmsys -host myhost -dbname intermediate -dsn
mydsn -cmd "CI \"PAYROLL:LICENSE2 DAT TXT.A-SRC;1\" /
USER_FILENAME=G:\license.dat.txt /FILENAME=license2-dat-01.txt /
PART=PAYROLL:PAYROLL.A;1 /WS_FILENAME=license2.dat.txt /
DESCRIPTION=\"test test\" /FORMAT=TXT /
ATTRIBUTES=(COMPLEXITY=lowish) /COMMENT=\"This is a test\" /
CHANGE_DOC=(\"PAYROLL_CR_21\") /KEEP"
```

- Dimensions commands requiring quotation characters within the double quotations characters referred to above will require:
  - For Windows: three double quotation characters before the quoted string and three double quotation characters after the quoted string.
  - For UNIX: single quotation characters before the command containing the quoted string.
- Windows example:

```
dmcli -con _tabuilder -cmd "EI """TA_DESKTOP:TEST TXT.BASE-
    SOURCE_INT""" /USER_FILENAME="""c:\temp\test.txt""""
    /PROJECT=TA_DESKTOP:INTERNAL /NOOVERWRITE"
```

- UNIX example:

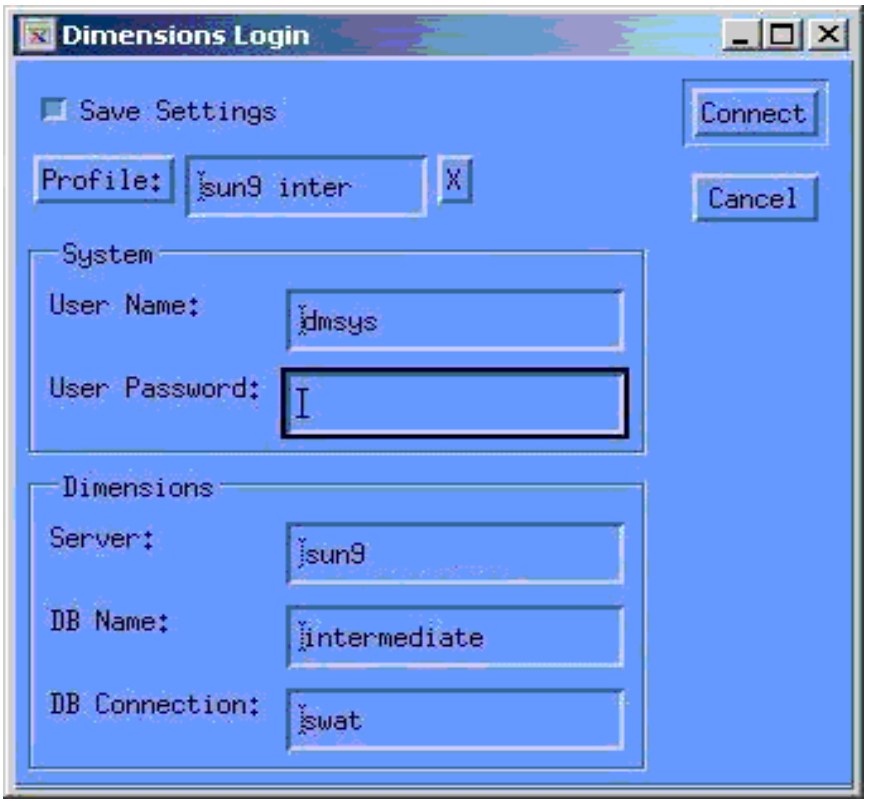
```
dmcli -con _tabuilder -cmd 'EI "TA_DESKTOP:TEST TXT.BASE-
    SOURCE_INT"' /USER_FILENAME='"/usr/temp/test.txt"'
    /PROJECT=TA_DESKTOP:INTERNAL /NOOVERWRITE"
```

Connection Processes for UNIX Dimensions Clients

Connecting Using the Remote Login Dialog Box

If you have an X Windows-based GUI environment installed on your UNIX Dimensions client (for example, Motif) and have the the X Windows environment `DISPLAY` set appropriately, then typing

```
dmcli
```



at the operating system prompt in a terminal window will launch a remote login dialog box (see above) that is functionally identical to that used for the Dimensions desktop client.

Populating this dialog box can be done in the following ways:

- Entering values into blank dialog box fields. This will look and feel the same as logging in to the Dimensions desktop client for the first time (see *User's Guide* for details).

NOTE The user name and host name are initially inherited from the client operating system, and are used to populate the User Id and Host Name fields. These will need to be replaced in those cases where your Dimensions server and client are not physically located on the same machine.

Entered dialog box fields can be stored by assigning a name in the Previous Connections field. This connection name will then be available for selection at subsequent invocations of the login dialog box or use of the `dmcli -con` command, and will be entered into the `.dimensions.rc` file described below.

Previous connection details can be deleted by use of the X button to the right of the Previous Connections field. This will also delete the information from the `.dimensions.rc` file.

- Loading values from the file `.dimensions.rc` in your home directory.

  This file is created following a successful log in attempt. All of the fields in the login dialog box – with the exception of the password – are saved to the file under the heading of a Connection Name. Once there exist stored connections in the file, then by default the most recently used connection is loaded by the dialog box.

  An example `.dimensions.rc file` is shown below:

```
<?xml version="1.0" encoding="UTF-8" standalone="no" ?>
<DimensionsConnections>
   <conname id="test1">
          <user>devlop</user>
          <dmdb>devlop</dmdb>
          <dsn>dev8</dsn>
          <host>aix4</host>
          <auto>yes</auto>
          <dflt>yes</dflt>
   </conname>
</DimensionsConnections>
```

Once successful connection has been established, the login dialog box will be dismissed and a

```
Dimensions>
```

prompt will be displayed in the terminal window. You can now enter commands on the Dimensions client.

NOTE All the UNIX connection mechanisms described below will also default to the GUI login dialog mechanism if they fail to successfully establish a connection.

Connecting from a Server Using the DMDB Environment Variable

On a Dimensions server *only*, you can get a local connection from the command line by setting the UNIX environment variable DMDB. Dimensions will search for any existing occurrences of the PCMSDB environment for backward compatibility with any scripts you have.

The syntax for DMDB is:

```
<base_db_name>@<connection_string>
```

for example,

```
intermediate@dim9
```

Connecting Using the -con Command-Line Parameter

This connection mechanism is functionally exactly the same as "Connecting Using the -con Command-Line Parameter" on page 23. Refer to that description for details.

Connecting Using the Command-Line

To support UNIX Dimensions client connections from non-Motif supporting terminals there is a command-line login interface to the saved `.dimensions.rc` connection file. The determining factor in these cases is the value of the X-Windows DISPLAY environment variable, if it is set the GUI login dialog box is used.

This connection mechanism is functionally exactly the same as "Connecting Using the Command-Line" on page 23. Refer to that description for details.

Examples of Client Commands

```
dmcli -param "c:\connection.txt" -cmd SCWS
```

connects to a Dimensions server using the parameters specified in c:\connection.txt and executes a SCWS command.

```
dmcli -user pcms -pass XXXX -host server1 -dbname intermediate -dsn dim9
    -cmd "FI FS:CABIN REPORT.A-SRC;1 /USER_FILENAME=c:\report.c"
```

connects to the Dimensions server node server1 as user pcms and executes a Get (Fetch) Item command of a text file.

```
dmcli
```

Without any parameters an interactive login box is invoked and you connect interactively to the Dimensions server. Commands can then be typed at the Dimensions client command prompt.

Selecting Item Revisions by Default

When `<revision>` is not specified, it defaults to the *latest revision in the user's current project*. Note that this **may not** necessarily be the latest revision of the item in the database, since only the revisions in the user's *project* are considered. *Latest revision* means that version file content *has most recently been created or updated* – which **might not** be the one with the highest entry in the revision field. This algorithm does not take into account the branch names or whether the branches are locked or owned remotely (via replication). For items which had previously been checked out, the time of creation/update is to be regarded as the time of the check in (RI command), *not* the earlier time of the check out (EI command).

You should bear in mind that criteria other than the above are *not* used in selecting a revision by default. The best way to make this point clear is to consider an example. Suppose that Revision 2 of an item is the latest revision (by the above criteria), and also that Revision 1 of this item has been related to request A_B_1 as *Affected*. Now an Extract (check out) Item (EI) command, to create Revision 3 as *in-Response-to* request A_B_1, *must specify Revision 1* in the `<item-spec>`, if Revision 3 is to start off as identical to Revision 1. Otherwise, by default, Revision 3 would start off as identical to Revision 2, despite the fact that it was not Revision 2 which was cited as *Affected*.

Updating the Content of an Item Without Changing the Item Revision

If the PRODUCT-MANAGER has setup the process model (control plan) so as to allow you to update the *file content* of an item revision residing at the initial lifecycle state *without* having to change the item revision, you should bear the following in mind if you do such an edit.

A particular item revision may have been imported into several projects either manually or as a result of project replication (it should be remembered that a project is basically a logical group of item revisions). In such situations, if you edit the

content of an item revision in one project *without* changing its revision, then in *all* projects that reference that item revision the content of the associated item file will be updated. Conversely, if you edit the content of an item revision in one project *and* change its revision, the changes in content will **only** be reflected in that project and any project replicated from it.

Project Assignment

Each user must have a (mandatory) default project, assigned as follows:

- When the Tool Manager initially registers the Dimensions user concerned, the user is automatically assigned to the default global project called `$GENERIC:$GLOBAL`. This project assignment enables the user to reference any item revision in any product in the base database to which they are connected.
- Subsequently, the user can then use the command SCWS (Set Current Project) or the `/WORKSET` qualifier found on certain commands to reference a specific project; e.g., one created from a baseline representing some development activity (this will then enable the user to reference item revisions that are pertinent to that development activity only).

  SCWS command qualifiers can be used to reassign (or not) the default project as described below:

  - `/DEFAULT` to specify that the current project assigned by SCWS will remain the default for all future sessions until respecified. An example of such a command is:

    ```
    SCWS PROD_X:MAINT /DEFAULT
    ```

  - `/NODEFAULT` to specify that the current project assigned by SCWS is for the duration of the present process/session only, and that the current project will revert to its former default setting once the session is exited. If neither the `/DEFAULT` nor `/NODEFAULT` qualifier is specified, then SCWS behaves as if `/DEFAULT` was specified. An example of such a command is:

    ```
    SCWS PROD_X:MAINT
    ```

  The /WORKSET qualifier found on certain commands is used in most cases to specify the project to be used for the *duration of the command* concerned.

Each project, when opened, must have a (mandatory) top level "working location" assigned to it. This "working location" defines a point in the directory hierarchy structure below which (or relative to which) the project file name is placed e.g. in UNIX `<dir>/<ws_filename>`. This is assigned as follows:

- By, where applicable, the `/DIRECTORY` command qualifier.
- The user's current working directory if `/DIRECTORY` is not specified or is not applicable.

The project file name as used in commands such as get or build consists of the relative directory path from the working location `<directory-spec>` and the file name from `<ws_filename>` concatenated together.

Request Attributes

Request Attribute 1 must always be defined in the process model. It must be given the variable name TITLE and be declared as single-valued. Its length must be less than or equal to 80 characters.

Request Attributes 2 and 3 must be defined in the process model and they must be defined as single-valued. These attributes appear in the report when users are e-mailed as the result of requests being actioned to new states.

Attributes used to define a block (table) must satisfy the following conditions.

- They must all be multiple-valued.
- They must all be declared as visible.

The maximum length of a default value for an attribute is restricted to 240 characters.

There is no facility to specify a list of values as default value for a multiple-valued attribute. However, a single value attribute can be specified as the default value for such an attribute.

Issuing Certificates

You can issue certificates through the DMCLI with the `-cert` option. This is most useful in conjunction with the REXEC or RSTAT command.

The following sample code is from a batch file in the `templates` directory:

```
echo RSTAT %DMJOBID. /STATUS=SUCCEEDED /RC=0 > c:\temp\dmcli.in
echo quit >> c:\temp\dmcli.in
dmcli -cert %DMCERTIFICATE. -file c:\temp\dmcli.in >
    c:\temp\%DMJOBID..log
```

This generates the following batch file:

```
echo RSTAT R-4195789 /STATUS=SUCCEEDED /RC=0 > c:\temp\dmcli.in
echo quit >> c:\temp\dmcli.in
dmcli -cert
    81C4AC3983A8AB3CD847B6BA185BF8C6853B7102BB0ADA1B1BF420FE96EFB90E2F9
    4F008AB99435FCE153EA40EE8C6F7C159E58BC61E01725EE6E7C491A88FE78C8DCE
    58F7A2824FCE00EBF0A2169C073873DD825430316B341A8A5C7F0B38EBAD677FF81
    53F7E5F < c:\temp\dmcli.in
```

For more information on certificates as well as the RSTAT and REXEC commands, see the *Serena Dimensions CM Developer's Reference*.

Operating System Differences

Spaces in File Names

UNIX and Windows operating systems support the use of spaces in file names.

Dimensions may allow the creation of files with leading and trailing spaces but some tools may not be able to access these files.

Windows UNC Paths

Dimensions allows the use of UNC (Universal Naming Convention) paths for work areas; e.g., working locations or getting items. As with all directory specifications, if a working location is set as a UNC path and that project is opened by the same user on UNIX, the directory specification will not be recognized.

Specifying a Project File Name in z/OS Item Operations

When running Dimensions item operations from a z/OS platform, you need to specify a 'backslash' character (\) in place of any parenthesis within a project file name. For example, if the project file name is TEST.COBOL(STAFF), to get (fetch) the item using the project file name you would need a command such as:

```
                    fi cv3prod:.-src /filename="TEST.COBOL\STAFF.CBL"
    /user_file="cvuser3.test.cobol(staff)"
```

Command-Line Logging and Usage Analysis

The Dimensions server provides functionality to support the concepts of command-line logging and usage analysis. This functionality can be used to help you perform command-line auditing and general usage analysis.

This logging functionality is available in three parallel forms:

- For all users, you can log a summary of all the commands that a server has processed to the database for viewing via a Dimensions published view. See "Audit Trail of Commands" on page 32 for more details.
- For all users, you can log the full details of the commands that a server has processed to a file. This includes client machine details, user details, and full commands run. See "Logging All Commands Run by All Users" on page 33 for more details.
- Each user also has the ability to log all their command details to a file as well, through the use of the *SET* command. See "SET – Set DIR, PRINTER, OVERWRITE, or CMD_TRACE Environment" on page 328 for more details.

Audit Trail of Commands

The "Audit Trail of Commands" functionality is an extension to the Dimensions server that allows a summary of all the commands processed by the server to be written to the Dimensions database.

This summary information is available through the published view PCMS_COMMAND_STATISTICS, which contains the following information:

- The Dimensions command run.

NOTE Only the command name, for example, CBL, CRB, CC is logged—not command qualifiers or parameters.

- The user who ran the command.
- The last date the command was run.
- How many times the user has *successfully* run this command.
- How many times the user has *unsuccessfully* run this command.

This functionality is enabled by the definition of

```
DM_AUDIT_CMD_USAGE      true
```

in the Dimensions DM.CFG configuration file or the operating-system environment.

To display the data logged, you can connect to the Dimensions database via a valid report user and run an SQL query like

```
SELECT * FROM pcms_command_statistics
```

using an SQL tool like sqlplus or db2.

An example of the data provided is shown below:

COMMAND	USER_NAME	NO_ SUCCESSES	NO_FAILURES	DATE_LAST_RUN
LNN	REICHANADMIN	2	0	2003-12-12 16:11:02
CNN	REICHANADMIN	0	2	2003-12-12 16:11:02
DWP	REICHANADMIN	0	1	2003-12-12 16:11:02
DNP	REICHANADMIN	1	0	2003-12-12 16:11:07
AUTH	REICHANADMIN	1	0	2003-12-12 16:11:07
DPL	REICHANADMIN	1	0	2003-12-12 16:11:07
DWS	REICHANADMIN	1	0	2003-12-12 16:11:07
SCWS	REICHANADMIN	1	0	2003-12-12 16:11:07
DVB	REICHANADMIN	1	0	2003-12-12 16:11:08
SWS	REICHANADMIN	1	0	2003-12-12 16:11:08
UUA	REICHANADMIN	1	0	2003-12-12 16:11:08
UREG	REICHANADMIN	1	0	2003-12-12 16:11:10
AUR	REICHANADMIN	10	0	2003-12-12 16:11:11

COMMAND	USER_NAME	NO_ SUCCESSES	NO_FAILURES	DATE_LAST_RUN
CP	REICHANADMIN	9	0	2003-12-12 16:15:41
CC	REICHANADMIN	104	0	2003-12-12 16:15:38
CI	REICHANADMIN	1264	0	2003-12-12 16:18:49

Logging All Commands Run by All Users

The "Logging All Commands Run by All Users" functionality is an extension to the server that allows a log file of all Dimensions server commands (plus parameters and session information) to be maintained. This log file will contain all the commands run by *all* users connecting to that server (except for commands that have a `/PASSWORD` qualifier in them like `AUDIT` for security reasons).

This functionality is enabled by the definition of

```
DM_INTERNAL_AUDIT_CMD_FILE <logFile>
```

in the Dimensions `DM.CFG` configuration file or the operating-system environment—where `<LogFile>` is the absolute path of the logging file. This file will be created and owned by the user running the Dimensions pooled servers.

This log file will contain information such as, for example,

```
** dmappsrv log "Wed Dec 31 22:36:21 2003" (GMT)
    (DMDB=INTERMEDIATE_TESTDB@hotaruchan) pid=3328 user="reichanadmin"
    node="AYANAMI" execution time = 0(s)
   'lwsd' (SUCCESS)
** dmappsrv log "Wed Dec 31 22:36:24 2003" (GMT)
    (DMDB=INTERMEDIATE_TESTDB@hotaruchan) pid=3328 user="reichanadmin"
    node="AYANAMI" execution time = 0(s)
   'lwsd' (SUCCESS)
** dmappsrv log "Wed Dec 31 22:36:55 2003" (GMT)
    (DMDB=INTERMEDIATE_TESTDB@hotaruchan) pid=3328 user="reichanadmin"
    node="AYANAMI" execution time = 0(s)
   'lwsd' (SUCCESS)
** dmappsrv log "Wed Dec 31 22:37:18 2003" (GMT)
    (DMDB=INTERMEDIATE_TESTDB@hotaruchan) pid=3328 user="reichanadmin"
    node="AYANAMI" execution time = 0(s)
   'lws' (SUCCESS)
** dmappsrv log "Wed Dec 31 22:37:27 2003" (GMT)
    (DMDB=INTERMEDIATE_TESTDB@hotaruchan) pid=3328 user="reichanadmin"
    node="AYANAMI" execution time = 2(s)
   'lsbl' (SUCCESS)
** dmappsrv log "Wed Dec 31 22:37:56 2003" (GMT)
    (DMDB=INTERMEDIATE_TESTDB@hotaruchan) pid=3328 user="reichanadmin"
    node="AYANAMI" execution time = 2(s)
   'lsar' (SUCCESS)
** dmappsrv log "Thu Jan 01 00:46:34 2004" (GMT)
    (DMDB=INTERMEDIATE_TESTDB@db2reichan) pid=3740 user="john doe"
    node="AYANAMI" execution time = 2(s)
   'lswd /rec' (SUCCESS)
```

```
** dmappsrv log "Thu Jan 01 00:49:45 2004" (GMT)
    (DMDB=INTERMEDIATE_TESTDB@db2reichan) pid=3740 user="john doe"
    node="AYANAMI" execution time = 2(s)
   'AI "MINAKO:AMI_CHAN0.A-SRC;reichan#2"' (FAILED)
** dmappsrv log "Thu Jan 01 00:49:48 2004" (GMT)
    (DMDB=INTERMEDIATE_TESTDB@db2reichan) pid=2684 user="reichanadmin"
    node="AYANAMI" execution time = 2(s)
   'AC "MINAKO_PR_1"' (FAILED)
** dmappsrv log "Thu Jan 01 00:49:50 2004" (GMT)
    (DMDB=INTERMEDIATE_TESTDB@db2reichan) pid=4016 user="reichanadmin"
    node="AYANAMI" execution time = 2(s)
   'AI "MINAKO:AMI_CHAN0.A-SRC;reichan#2"' (SUCCESS)
** dmappsrv log "Thu Jan 01 00:49:50 2004" (GMT)
    (DMDB=INTERMEDIATE_TESTDB@db2reichan) pid=4016 user="reichanadmin"
    node="AYANAMI" execution time = 2(s)
   'AI "MINAKO:AMI_CHAN1.A-SRC;reichan#2"' (SUCCESS)
```

NOTE All log times are reported in GMT format to allow meaningful comparisons across time zones.

Logging Users Who Connect to Dimensions

In secure environments where you wish to track all users attempting to connect to a Dimensions server, Dimensions offers you the facility to define a file into which the details of all clients connecting to that server will be logged. The location of this log file is defined by the symbol

```
DM_USER_AUDIT_LOG_FILE <file-name>
```

in the `dm.cfg` configuration file, where `<file-name>` is the absolute path on the server of the log file. This log file will contain details such as when connect attempts were made, from which clients, by which user, and if that connection was successful or not.

All connection attempts are split into two types:

- The first type is an operating system user check. This verifies that the user attempting to log in actually exits on that server.
- The second type is a check to verify that the user specified is registered against Dimensions.

An example is as follows:

```
** dmpool connect "Thu Oct 14 19:40:53 2004" (GMT) pid=3332
    user="reichanadmin" node="AYANAMI"
   User attempted to login to Dimensions - OS user check (SUCCESS)
** DMAPPSRV connect "Thu Oct 14 19:40:54 2004" (GMT)
    (DMDB=ENTRY_LEVEL_TESTDB@hotaruchan) pid=3396 user="reichanadmin"
    node="AYANAMI"
   User attempted to login to Dimensions - database check (SUCCESS)
** dmpool connect "Thu Oct 14 20:20:13 2004" (GMT) pid=3332
    user="reichanadmin" node="AYANAMI"
   User attempted to login to Dimensions - OS user check (SUCCESS)
```

```
** DMAPPSRV connect "Thu Oct 14 20:20:20 2004" (GMT)
    (DMDB=ENTRY_LEVEL_TESTDB@hotaruchan) pid=3396 user="reichanadmin"
    node="AYANAMI"
   User attempted to login to Dimensions - database check (SUCCESS)
```

NOTE All log times are reported in GMT format to allow meaningful comparisons across time zones.

Invoking Help at the Dimensions Command-Line

When working at the Dimensions command line, you can invoke text based help for any Dimensions command. When you invoke help, the following information is returned:

- The full name of the command.
- The complete syntax of the command.

To invoke help for a command:

At the Dimensions command line, type:

```
help <Dimensions function mnemonic>
```

For example:

```
Dimensions>help abl
ABL   -   Action Baseline or Items
        <baseline-spec>
        [/ITEM_FILTER=<item-spec>]
        [/STATUS=<status>]
Operation completed

Dimensions>help cbl
CBL   -   Create Baseline
        <baseline-spec>
        /PART=<part-spec>
        [ /TEMPLATE_ID=<template-id>]
        [ /TYPE=<baseline-type>]
        [ /WORKSET=<project-spec>]
        [ /ATTRIBUTES=(<attr1>=<value1>,<attr2>=<value2>,…) ]
        [ /LEVEL=<integer> ]
        [ /CHANGE_DOC_IDS=(<request1>,<request2>,…) ]
        [ /CANCEL_TRAVERSE ]
        [ /INCLUDE_INFO ]
        [ /[NO]INCLUDE_
```

Chapter 2

Command Reference

This chapter contains an alphabetic listing of commands.

NOTE For detailed information on roles, groups, and privileges, see Chapter 1 of the *Serena Dimensions CM User's Guide* and Chapter 4 of the *Serena Dimensions CM Process Modeling User's Guide*.

ABL – Action Baseline or Items

NOTE This command is not available in the Issue Management–only licensed version of Dimensions.

```
<baseline-spec>
[/ITEM_FILTER=<item-spec>]
[/STATUS=<status>]
[/COMMENT=<text>]
```

Example `ABL PROD:"R M VERSION 2 FOR HP" -/STATUS="UNDER TEST"`

- `<baseline-spec>`

  Comprises:

  `<product-id>:<baseline-id>`

- `/ITEM_FILTER=<item-spec>`

  Comprises:

  `<product-id>:<item-id>.<variant>-<item-type>;<revision>`

  Wildcard characters _ (underscore) for "any one" and % (percent) for "zero or more" characters may be used to identify what subset of the item revisions in the baseline are to be actioned to a new *item* lifecycle state.

  If omitted, it is `<baseline-spec>` itself that is actioned. See Description on page 38 for details.

- `/STATUS=<status>`

  Specifies the new status to be given *either* to the baseline itself *or* to every item revision in the subset identified above. Unless you hold the `PRODUCT-MANAGER` role, this status must be reachable from the current status (of each object to be actioned) by a single lifecycle transition.

  If omitted, the new status is the next normal lifecycle state for each object. See Description on page 38 for details.

- `/COMMENT=<text>`

  This is an optional user-defined action-comment.

  Dimensions enters a default comment if this is omitted.

Description

This command actions to a new lifecycle state **either** the specified baseline (if the `<item-spec>` filter is omitted) **or** each of the item revisions in the baseline that match the specified filter. (To action both the baseline itself and (a subset of) the item revisions in it, two instances of the ABL command must be used.)

If `<status>` is omitted, the object(s) to be actioned must each be at a normal lifecycle state, and each will be advanced one state further along the normal lifecycle. Thus it is possible in a single ABL operation to advance all the matching item revisions each by one approval level (i.e. each moves along its normal lifecycle by one transition), even when the start and end states of these transitions are not the same for all the item revisions

actioned. This is particularly convenient when the matching item revisions are of more than one item type, which means that they would probably be following different lifecycles.

Constraints

Unless you have the appropriate management privileges, you must, for each object to be actioned, have been assigned a role authorized to perform its transition (whether `<status>` is specified or not).

AC – Action Request

```
<request-id>
[/STATUS=<status>]
[/ACTION_CHECK] or [/CLOSURE_CHECK]
```

Examples

```
AC PROD_DR_25 /STATUS="CRB APPROVED"
AC PROD_DR_26 /STATUS="CRB APPROVED" /ACTION_CHECK
AC PROD_DR_27 /ACTION_CHECK
AC PROD_DR_28 /CLOSURE_CHECK
AC PROD_HELD_350
```

- `<request-id>`

  This is the identity of the request to be actioned or checked.

- `/STATUS=<status>`

  Specifies the new status to be given to the request (provided `ACTION_CHECK` is omitted). Unless the user has the role of CHANGE-MANAGER (see note below), the new status must be one reachable by a single lifecycle transition from the current status.

  If omitted, Dimensions will action the request to its next state in the normal lifecycle, or if the request is held, save it (which places it at its initial lifecycle state).

NOTE Saving results in a changed value for `<request-id>`, but `$LAST` can be used to refer to it in subsequent commands in an CMD file. (See note on CC command-mode command on page 79.)

CAUTION! `<status>` **cannot be omitted** if the current status is a state not in the normal lifecycle.

- `/ACTION_CHECK`

  Indicates that Dimensions is to check whether the specified request can be either actioned to the state specified by the `/STATUS` qualifier or the next normal state if `/STATUS` is not specified, while conforming to the current rules. This qualifier must not be used with the `/CLOSURE_CHECK` qualifier.

  Mandatory attributes must be set to action next normal state or state defined by `/STATUS`.

- `/CLOSURE_CHECK`

  Indicates that Dimensions is to check whether the specified request can be actioned to the final state in its normal lifecycle, while conforming to the current rules. This qualifier must not be used with the `/STATUS` qualifier or the `/ACTION_CHECK` qualifier.

NOTE A user with the role of CHANGE-MANAGER may action any request to any valid state in its lifecycle. This includes the possibility of re-opening a request which has been closed or rejected (i.e. has reached a final state).

Constraints

Unless you have the appropriate management privileges to action a request, you must have a role required to action the request to a new state. To be able to select any lifecycle state from any stage of the lifecycle, you need CHANGE-MANAGER role or have the role on the lifecycle transition if that transition exists.

Requests that were created in a held state are not considered to have been "created" as far as Process Models where optional sensitive states or attributes have been set up ("electronic signatures") are concerned. The act of entering them into the system by actioning them out of the held state is considered the "authorization point" for such process models. This also applies to held requests that are updated at the held state (using the command UC) before being actioned on.

ACDI – Action Request Items

```
<request-id>
[/[NO]CANCEL_TRAVERSE]
[/LOGFILE=<log-file>]
[/STATUS=<status>]
[/WORKSET=<project>]
```

Example `ACDI PAYROLL_CR1`

Parameters

- `<request-id>`

  The name of a Dimensions request.

- `/CANCEL_TRAVERSE`

  By default, all requests related as dependent to the specified request are processed by this command. This qualifier forces the command to process only the request specified.

- `/LOGFILE`

  Specifies a local log file to which the command is to divert all messages.

- `/STATUS`

  Specifies the status to which items and requests are to be actioned. If this is not specified, the next default state is used.

- `/WORKSET`

  Specifies the project to be processed by this command.

Description

This command actions to a specified state all the items and requests that are related to a specified request.

When multiple revisions of the same item are related to requests processed by this command, only the latest is processed.

If any failure occurs, all actions are rolled back.

ACDWS – Add Request Items to Project

```
<request-id>
[/[NO]CANCEL_TRAVERSE]
[DIRECTORY=<project-directory-filter>]
[/[NO]RECURSIVE]
[/LOGFILE=<log-file>]
[/WORKSET=<project>]
```

Example `ACDWS PAYROLL_CR1`

Parameters

- `<request-id>`

  The name of a Dimensions request.

- `/CANCEL_TRAVERSE`

  By default, all requests related as dependent to the specified request are processed by this command. This qualifier forces the command to process only the request specified.

- `/DIRECTORY`

  Enables you to specify a project directory filter to restrict the number of items processed.

- `/RECURSIVE`

  Used with /DIRECTORY, this specifies that the filter is to be processed recursively; that is, subdirectories are to be processed.

- `/LOGFILE`

  Specifies a local log file to which the command is to divert all messages.

- `/WORKSET`

  Specifies the project to be processed by this command.

Description

This command adds to the current project (or a specified project) all the items that are related to a specified request.

When multiple revisions of the same item are related to requests processed by this command, only the latest is processed.

ACF – Assign Data Formats to Request Types

```
<product-id>
/TYPE=<request-type>
/FORMAT=<format>
[/EXTENSION=<file-extension>]
```

Example `ACF PAYROLL /TYPE=CR /FORMAT=HTM /EXTENSION="html"`

- `<product-id>`

  Specifies the product within which the assignment is to be made.

- `/TYPE=<request-type>`

  Specifies the request type within the specified product to which the format assignment is to be made.

- `/FORMAT=<format>`

  Specifies a valid data format to be assigned to the specified request type. This will then become the valid data format assigned when creating a request of type `<request-type>`.

- `/EXTENSION=<file-extension>`

  Optionally specifies a file extension to be assigned to the specified request type.

Description

This command assigns a data format to a particular request type. The data format must have been previously defined using the Define Data Format (DDF) command (see page 125) or the Administration Console (see the *Process Modeling User's Guide*).

Once assigned, the format and file are used by the Dimensions client applications (web client, desktop client, and IDE) to correctly choose an application/viewer to display the request.

Optionally, you can also specify a file name extension to be assigned to the specified request type.

This function is available *only* in Command Mode.

Constraints

Only users with the appropriate management privileges can run this command.

ADF – Assign Data Formats to Item Types

NOTE This command is not available in the Issue Management–only licensed version of Dimensions.

```
<product-id>
/ITEM_TYPE=<item-type>
[/FORMAT_LIST=(format1,format2,format3,…)]
```

Example `ADF PROD /ITEM_TYPE=DAT /FORMAT_LIST=(C,TXT,CPP)`

- `<product-id>`

  Specifies the product within which the assignment is to be made.

- `/ITEM_TYPE=<item-type>`

  Specifies the item type within the specified product to which the format assignments are to be made.

- `/FORMAT_LIST=<format1,format2,format3,…)`

  Specifies the list of valid data formats to be assigned to the specified item type. This will then become the valid list of data formats from which users must select when creating an item.

  If `/FORMAT_LIST=.` (dot) is specified, then any existing assignments are cleared.

Description

This command assigns data formats to particular item types. The data formats must have been previously defined using the Define Data Format (DDF) command (see page 125) or the Administration Console (see the *Process Modeling User's Guide*). Once these formats are assigned to an item type, the choice of one these formats is compulsory when creating items of that type; whereas, if none has been assigned, then any format can be used at the time of item creation, even one not defined by a user with the role of TOOL-MANAGER.

IMPORTANT! This command overwrites the existing list of formats.

Constraints

Only users with the appropriate management privileges can run this command.

This function is available *only* in Command Mode.

AGRPU – Assign Groups to a User

```
<user-name>
[/GROUPS=(<group-name>,…)]
[/ADD] or [/REMOVE]
```

Example `AGRPU <user-name> /GROUPS=(<group-name>,…) /ADD`

- `<user-name>`

  is the user name for the user to whom you are adding groups or from whom you are removing groups.

- `<group-name>,…`

  is the list of groups to be added or removed.

Description

Use this command to assign groups to a user or to remove group assignments from a user.

Constraints

Only users with the appropriate management privileges can run this command.

AI – Action Item

NOTE This command is not available in the Issue Management–only licensed version of Dimensions.

```
<item-spec>
[/FILENAME=<file-name>]
[/STATUS=<status>]
[/COMMENT=<text>]
[/WORKSET=<project-spec>]
```

Example `AI PROD:"QUERY RELEASE".AAAA;2 /FILENAME=query.c -/STATUS="UNDER TEST"`

- `<item-spec>`

  Comprises:

  ```
  <product-id>:<item-id>.<variant>– <item-type>;<revision>
  ```

`<item-id>`	may be omitted if `<file-name>` is specified.
`<variant>`	may be omitted if only one exists.
`<revision>`	may be omitted if `<file-name>` is specified.

- `/FILENAME=<file-name>`

  Specifies the name of the project file name.

  The project file name identifies the relative path (directory plus file name) from the working location to the item to be used from the current project. The project file name for the same item may *differ* between projects; for example, `src/hello.c`, `hello.c`, or `src/build/hello.c`.

  It may be omitted if `<item-id>` is specified.

- `/STATUS=<status>`

  Specifies the new status to be given to the revision.

  Except as noted below, it must be reachable from the current status by a single lifecycle transition.

  It can be omitted, **only if** the current status is a state in the normal lifecycle, in which case the item will be actioned to its next normal lifecycle state.

- `/COMMENT=<text>`

  An optional user-defined action-comment.

  Dimensions enters a default comment if this is omitted.

- `/WORKSET=<project-spec>`

  Comprises:

  ```
  <product-id>:<project-id>
  ```

  This optionally specifies the project to be used for this command: failing this, the user's current project will be taken.

The project is used to select the revision to action if the revision is not actually specified. If the revision is specified, the project is ignored (as Dimensions assumes reference to the explicit revision).

Constraints

Unless you have the appropriate management privileges, you can run this command only if the current item revision is in your pending list.

NOTE To simplify the transfer of an existing product to Dimensions, a user with role of PRODUCT-MANAGER can action any item revision to any valid state in its lifecycle. This includes permission to re-action a revision which has reached a final state, thereby re-opening it to further ordinary actioning.

Such a user can also action items when no appropriate roles have yet been allocated. This is to facilitate quicker migration of files.

AIWS – Add Item Revision to Project

NOTE This command is not available in the Issue Management–only licensed version of Dimensions.

```
<item-spec>
[/FILENAME=<file-name>]
/WORKSET=<project-spec>
/KEEP_STAGE
```

Example `AIWS PROD_X:"HELLO WORLD".AAAA-SRC;2.6 /WORKSET=PROD_X:"WS MAINT"`

- `<item-spec>`

  Comprises:

  ```
  <product-id>:<item-id>.<variant>– <item-type>;<revision>
  ```

`<item-id>`	may be omitted if `<file-name>` is specified.
`<variant>`	may be omitted if only one exists
`<revision>`	defaults to the latest revision in your current project.

- `/FILENAME=<file-name>`

  Specifies the name of the project file name.

  The project file name identifies the relative path (directory plus file name) from the working location to the item to be used from the current project. The project file name for the same item may *differ* between projects; for example, `src/hello.c`, `hello.c`, or `src/build/hello.c`.

  It may be omitted if `<item-id>` is specified.

- `/WORKSET=<project-spec>`

  Comprises:

  ```
  <product-id>:<project-id>
  ```

  This command will add the item specified to the given project. The specified item and project must exist. If the specified item is already in the project a warning will be given.

  Item revisions may be removed from a project using the RIWS command.

- `/KEEP_STAGE`

  Specify this optional qualifier to avoid resetting the stage of the item revision in the target project to the initial stage, and to instead keep the stage of the item. The default behavior (when this qualifier is not specified) is to reset the stage to the initial stage. Note that this qualifier can only be used when adding items to projects that use the manual deployment model.

Description

The item selected by the `<item-spec> [/filename=<file-name>]` parameters from the current project will be added to the project specified by the `/WORKSET=<project-spec>` qualifier. The baseline project file name is not used.

NOTE Whenever a new revision is added to a project, its stage is reset to DEVELOPMENT, and associated deployment areas and reference areas are updated.

Constraints

Normally, only users with the appropriate management privileges can run this command.

This constraint can, however, be relaxed using the Set Project Permissions (SWSP) command, as described on page 342.

APNO – Allocate Part Numbers

```
<part-spec>
[/GENERIC_NO=<standard-no> [/NOCHECK]]
[/LOCAL_NO=<local-no>]
[/DESCRIPTION=<description>]
```

Example `APNO PROD:"RELEASE MANAGEMENT" /GENER="SQLS 1234"`

- `<part-spec>`

  Specifies the design part to be numbered. It comprises:

  `<product-id>:<part-id>.<variant>;<pcs>`

`<variant>`	may be omitted if only one exists.
`<pcs>`	is ignored. A part-number always applies to all PCSs.

- `/GENERIC_NO=<standard-no>`

  Specifies a standard part number to be allocated.

  It may be omitted provided `<local-no>` is specified.

  - `/NOCHECK`

  Specifies that the standard part number need not be in a range of numbers allocated to the product.

- `/LOCAL_NO=<local-no>`

  Specifies a local part number to be allocated.

  It may be omitted provided `<standard-no>` is specified.

- `/DESCRIPTION=<description>`

  Specifies a new description to be given to the design part.

Constraints

Only users with the appropriate management privileges can run this command.

Each part category that is to use part numbers has to be enabled by the Administration Console.

AUDIT – Audit Project or Area

NOTE This command is not available in the Issue Management–only licensed version of Dimensions.

```
[<project-spec>]
/STAGE=<stage-spec>
/USER_FILENAME=<file-spec>
[/AREA=<area-name>]
[/[NO]FIX]
```

Examples

Audit the UNIT TEST deployment area `"ACME_2.1-WINDOWS-UT"`, which is assigned to project ACME:ACME_2.1, repairing the area and generating a report file:

```
AUDIT "ACME:ACME_2.1"
   /STAGE="UNIT TEST" -
   /AREA="ACME_2.1-WINDOWS-UT" -
   /USER_FILENAME="c:\my_audit_report.txt" -
   /FIX
```

Audit the contents of the DEVELOPMENT deployment area ACCT1.0-ZOS-DEV, which is assigned to project EXEDLL:EXEDLL 2.0, and generate a report file:

```
AUDIT "EXEDLL:EXEDLL 2.0"
   /STAGE="DEVELOPMENT" -
   /USER_FILENAME="D:\temp\audit.log" -
   /AREA="ACCT1.0-ZOS-DEV"
```

- `<project-spec>`

  Comprises `<product id>:<project id>` and specifies the project to be audited.

- `/STAGE=<stage-spec>`

  Specifies the ID of a deployment stage to be audited.

- `/USER_FILENAME=<file-spec>`

  Specifies a file where the audit output is to be saved.

- `[/AREA=<area-name>]`

  Specifies the ID of a deployment area to be audited. If not specified, all areas for the project and stage pair are audited.

- `[/FIX]`

  Repair an area (synchronize it with all items corresponding to the specified deployment stage). Repairing an area ensures that it contains all item revisions from the project that are at the specified deployment stage. Any other files present in the area will not be affected by this feature.

NOTES

- AUDIT works with z/OS mainframes by utilizing Dimensions metadata stored on the mainframe.
- It is possible to audit one area or *all* areas (by not specifying an area on the *command line*).

AUGRP – Assign Users to a Group

```
<group-name>
[/USERS=(<user-name>,…)]
[/ADD] or [/REMOVE]
```

Example `AUGRP <group-name> /USERS=(<user-name>,…) /ADD`

- `<group-name>`

  is the name of the group to which you are adding users or from which you are removing users.

- `<user-name>,…`

  is the list of users to be added or removed.

Description

Use this command to assign users to a group or to remove user assignments from a group.

Constraints

Only users with the appropriate management privileges can run this command.

AUR – Assign User Roles

```
<user-name>
/ROLE=<role>
[/TYPE=<assignment-type>]
/PART=<product-id>:<part-id>.<variant>
[/CAPABILITY=<capability>]
[/ADD] or [/DELETE]
[/WORKSET=<project-spec>]
[/REPORT]
```

Example

```
AUR SMITH
   /ROLE=DEVELOPER -
   /CAPABILITY=P -
   /PART=PROD:"RELEASE MANAGEMENT" -
   /ADD
```

- `<user-name>`

  This is the login user name of the Dimensions user to whom the role is (to be) assigned.

- `/ROLE=<role>`

  Specifies a role to be defined or assigned to a user.

- `/TYPE=<assignment-type>`

  Specifies the type of assignment. It is either C (denoting role candidate definition) or R (denoting actual user role assignment). If omitted, R is assumed.

- `/PART=<product-id>:<part-id>.<variant>`

  Specifies a design part over which this role assignment is applicable.

NOTE If the variant field is left blank, the role assignment applies to **all** variants of the design part, excluding those that have an explicit role.

- `/CAPABILITY=<capability>`

  Specifies that this role assignment is one of the following:

  - L for Leader
  - P for Primary
  - S for Secondary (default).

  Leader (**L**) role function: It is sometimes useful to have more than one user with a particular role with respect to a request or item-spec e.g. so that they can add comments (called Action Descriptions in requests). However, it may also be appropriate to restrict the number of users in this group who can actually action the object to the next stage in its lifecycle. The way to implement this is via the Leader function. When a Leader function is defined in a group of users who have the same role for a given object, only the Leader can update the associated attributes **and** action on the object. All other users with the role may add only Action Descriptions or user comments. The Leader function applies whether rules are used or not. If Leader role function is assigned to a user, then Primary role function (described below)

cannot be assigned to the same user i.e. Leader role and Primary role functions are mutually exclusive.

The **Primary** (**P**) user for a role in the lifecycle of an object is the user regarded in the project as having the main responsibility for that role on the request or item-spec. There cannot be more than one Primary user defined for a role (as applicable to any particular design part or segment of the product structure). If Primary role function is assigned to a user, then Leader role function (described below) cannot be assigned to the same user i.e. Primary role and Leader role functions are mutually exclusive

Secondary (**S**) users are intended to act as deputies for the Primary. They have exactly the same privileges as the Primary: they can add action comments and also, unless the Leader capability is in use, update the object's attributes and action them.

- `/ADD`

  Specifies that this user role assignment is to be added. This is the default.

- `/DELETE`

  Specifies that this user role assignment is an existing one to be revoked.

  If omitted, this assignment is a new one to be granted.

- `/WORKSET=<project-spec> comprises:`

  `<product-id>:<project-id>`

  This is optional. If specified, the role assignment will apply to that particular project. If unspecified, the role assignment will apply to all projects, unless the role is `WORKSET-MANAGER` (in which case it is necessary to assign a specific project for that role-assignment to be effective).

- `REPORT`

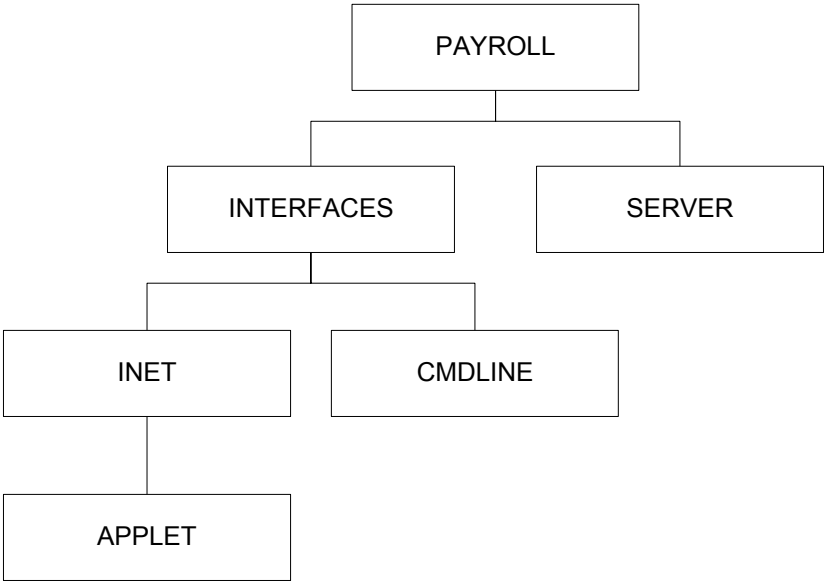
  This is optional. If specified, an on-screen report will be generated detailing what the result of such a proposed role assignment would be, without actually performing the role assignment—a "what if report".

NOTE This option is provided to aid administrators in understanding the impact of making a role assignment change.

Example

Consider the following design part structure and current user role assignments:

- User JOHN has role DEVELOPER on the top part `PAYROLL:PAYROLL`.
- User JILL also has the role DEVELOPER on the top part `PAYROLL:PAYROLL`.
- User CHRIS has the role DEVELOPER on the `PAYROLL:APPLET` design part.



Given the above scenario, entering the following command

```
AUR USER2 /ROLE=DEVELOPER /PART=PAYROLL:INTERFACES /REPORT
```

would generate a report, like the following, detailing how activating this role assignment would override the DEVELOPER role of various existing users who have been assigned that role at various other levels in the design part structure

```
Warning: You are removing a role from other user(s)…
   User JOHN was assigned the role on Design Part PAYROLL:PAYROLL.A;1
   By making this role assignment the above user will no longer have
        the role on part PAYROLL:INTERFACES.A;1 or any of its
        descendants

   User JILL was assigned the role on Design Part PAYROLL:PAYROLL.A;1
   By making this role assignment the above user will no longer have
        the role on part PAYROLL:INTERFACES.A;1 or any of its
        descendants

Warning: You are assigning the role on the following Design Parts…
   The role assignment will be effective on Design Part
        PAYROLL:INTERFACES.A;1
   The role assignment will also be effective on Design Part
        PAYROLL:INET.A;1
   The role assignment will also be effective on Design Part
        PAYROLL:CMDLINE.A;1

Warning: The role assignment will not take effect on Design Part
     PAYROLL:APPLET.A;1 or any of its descendants
   The following user(s) already hold the role:
      CHRIS

Operation completed
```

Constraints

Only users with the appropriate management privileges can run this command.

AUTH – Authorize Access to Node

```
[/NETWORK_NODE=<node-name>
[/USER=<userid>
[/PASSWORD=<password>
[/NEW_PASSWORD=<new-password>]]]]
```

Examples `AUTH /NETWORK_NODE=MYNODE`

requests a list of authenticated users on the node `MYNODE`

`AUTH /NETWORK_NODE=MYNODE /USER=MICKEY /PASSWORD=MOUSE`

requests access to user files on node `MYNODE` for the user `MICKEY`, with password `MOUSE`.

- `/NETWORK_NODE=<node-name>`

  Specifies the name of the node where your user files are stored.

- `/USER=<userid>`

  This is your User ID for the node specified.

- `/PASSWORD=<password>`

  This is the password associated with the User Id specified

- `/NEW_PASSWORD=<new-password>`

  This is the string that you want to change your password to.

Description

The AUTH command enables you to perform tertiary node access to items located on a remote node. All communication across the network of this sensitive information is encrypted.

You can use AUTH to:

- Obtain information about current authenticated users as follows:
  - To obtain a list of all authenticated users for each of the nodes currently available, enter the command with *no* parameters.
  - To obtain a list of authenticated users for one node, enter the command with the parameter `/NETWORK_NODE`.
- Change the current user on a node. Enter the command with the parameters `/NETWORK_NODE` and `/USER`. The user must already have been authenticated.
- Request access for a specified user on a node. Enter the command with the parameters /NETWORK_NODE, /USER and /PASSWORD. You can also change the password at the same time, by specifying the parameter /NEW_PASSWORD.

AWS – Action Project

NOTE This command is not available in the Issue Management–only licensed version of Dimensions.

```
<project-spec>
[/ATTRIBUTES=(<attr>,…)]
[/STATUS=<status>]
[/COMMENT=<text>]
```

Example `AWS <project-spec> /ATTRIBUTES=(<attr>,…)`

- `<attr>,…` is a list of attributes.
- `<text>` is an optional comment.
- If you omit `/STATUS`, AWS uses the next normal lifecycle state.

Description

This command is made necessary by the fact that projects have a user-defined lifecycle.

Constraints

Unless you have the appropriate management privileges, you must, for each object to be actioned, have been assigned a role authorized to perform its transition (whether `<status>` is specified or not).

BC – Browse or Print Request

```
<request-id>
[/FILENAME=<user-filename>]
[/[NO]PRINT]
[/ACTION_NO=<number>]
[/ATTACHMENTS=([FILENAME=<file-id>, USER_FILE=<user-file>],
               [FILENAME=<file-id>, USER_FILE=<user-file>],…)]
```

Examples

```
BC PROD_DR_25 /ACTION=2

BC PAYROLL_CR_21
/ATTACHMENTS=([FILENAME=Figure2.jpg,USER_FILE=c:\temp\attachment1.jpg],
              [FILENAME=Figure4.jpg,USER_FILE=c:\temp\attachment2.jpg])
```

- `<request-id>`

  This is the identity of the request to be browsed or gotten and/or printed.

- `/FILENAME=<user-filename>`

  Specifies the name of the file which will be created in the user area, and into which the request will be gotten. If this qualifier is used, interactive browsing is not invoked.

NOTE This parameter must be specified if you wish to run BC from Dimensions for z/OS.

- `/PRINT`

  Specifies that the request is to be printed. If this qualifier is used, interactive browsing is not invoked.

- `/ACTION_NO=<number>`

  Specifies that the request is to be browsed or gotten and/or printed in its state as it was prior to the action given by `<number>`.

  If this is not specified, the current state of the request is shown.

- `/ATTACHMENTS=([FILENAME=<file-id>, USER_FILE=<user-file>])`

  Specifies an attachment (<file-id>) to be retrieved from a given request (<request-id>) to the file that you specify in the <user-file> parameter. The /FILENAME qualifier is optional for non-interactive use if you use the /ATTACHMENTS qualifier (however <request-id> is still required).

  The FILENAME parameter in the /ATTACHMENTS qualifier is the attachment associated with the request, and the /FILENAME qualifier is the user file where the expanded browse template is saved.

Additional information:

- A request cannot have two attachments with the same <file-id>.
- When you add or delete attachments, or update their descriptions, the action is recorded in the request's history.
- A new request substitution variable called %chdoc_attachments% has been added, which enables you to view details of any attachment linked to a request that you browse.
- When you delete a held request, its attached files and history records are also deleted.

Constraints

This command can be run by any user with a role (any role will suffice) on the product owning the request selected.

BI – Browse Item

NOTE This command is not available in the Issue Management–only licensed version of Dimensions and cannot be run from Dimensions for z/OS.

```
<item-spec>
[/ROOT_PROJECT=<project-spec>]
[/FILENAME=<file-name>]
[/BASELINE=<baseline-spec>]
[/WORKSET=<project-spec>]
```

Example

```
BI PROD:"QUERY RELEASE".AAAA-SRC;
```

- `<item-spec>`

  Comprises:

  ```
  <product-id>:<item-id>.<variant>–<item-type>;<revision>
  ```

`<item-id>`	may be omitted if `<file-name>` is specified.
`<variant>`	may be omitted if only one exists.
`<revision>`	is ignored if `<baseline-spec>` is specified; otherwise, if omitted, the latest revision is used (see Introduction on page 16).

- `/ROOT_PROJECT=<project-spec>`

  Comprises:

  ```
  <product-id>:<project-id>
  ```

  This optionally specifies the root project. Use this when the current project set via SCWS (or the project specified by the /WORKSET qualifier) occurs in more than one project tree.

- `/FILENAME=<file-name>`

  Specifies the name of the project file name. If /ROOT_PROJECT is used to specify a the root project, /FILENAME is interpreted in the scope of that project.

  The project file name identifies the relative path (directory plus file name) from the working location to the item to be used from the current project. The project file name for the same item may *differ* between projects; for example, `src/hello.c`, `hello.c`, or `src/build/hello.c`.

  It may be omitted if `<item-id>` is specified.

- `/BASELINE=<baseline-spec>`

  Specifies a release-baseline which contains the particular revision of `<item-spec>` to be browsed. It comprises:

  ```
  <product-id>:<baseline-id>
  ```

  If this qualifier is omitted, the specified or default `<revision>` (as described above) is browsed.

- `/WORKSET=<project-spec>`

Comprises:

```
<product-id>:<project-id>
```

This optionally specifies the project to be used for this command: if not specified, the user's current project will be taken.

Constraints

This command can be run by a user who has a role on the design part owning the item or a role on one of the ancestor nodes of that design part. For more information see the section 'About Role Responsibilities' in the *Process Modeling User's Guide*.

BLD – Build

NOTE This command is not available in the Issue Management–only licensed version of Dimensions.

```
<project-spec>
[/AREA=<area-name>]
[/[NO]AUDIT]
[/[NO]BATCH]
[/BUILD_CLEAN]
[/BUILD_CONFIG = <build-configuration-name>]
[/BUILD_OPTIONS = (<opt1>=<value1>,<opt2>=<value2>,…)]
[/[NO]CAPTURE]
[/CHANGE_DOC_IDS=(<request1>,<request2>,…)]
[/TARGETS=<targets-list>]
[/USER_FILENAME = <file-name>]
```

Example

```
BLD "ACME_2.1" /AREA="ACME_2.1-WINDOWS-UT"
   /TARGETS=("foo.exe", "win32\bar.exe")
   /NOAUDIT
   /NOBATCH
   /CAPTURE
   /CHANGE_DOC_IDS=(PVCS_CR_1234)
   /USER_FILENAME=D:\temp\bld.log
```

Parameters

- `<project-spec>`

  Specifies the name of a project.

- `/AREA=<area-name>`

  Specifies the Dimensions area to be used for the build. If this is not specified, all areas associated with the build configuration or configurations are used.

- `/AUDIT`

  Specifies that an audit is to be run before the build.

  Default: no audit

- `/BATCH`

  Specifies that the build is to be run in batch mode (the command does not wait for the build to finish).

  Default: no batch.

- `/BUILD_CLEAN`

  Specifies that the area is to be cleaned of targets before the build process begins.

  Default: no clean.

- `/BUILD_CONFIG=<build-configuration-name>`

  Specifies the build configuration. If this is not specified, all build configurations associated with the Dimensions project are used.

- /BUILD_OPTIONS=(<opt1>=<value1>,<opt2>=<value2>,…)

  Specifies any build options. For a list of all build options see *Dimensions Build Predefined Symbols* in *The Templating Language and Processor* chapter of the *Developer's Reference*.

- CAPTURE

  Specifies whether built targets are to be collected. It is not possible to collect default targets when building at the DEVELOPMENT stage.

  Default: no capture.

- /CHANGE_DOC_IDS=(<request1>,<request2>, …)

  <requestN> identifies a request to which the new items created from the built final targets are to be related in-Response-to if required by change management rules.

- /TARGETS=<targets-list>

  Specifies the list of targets to be built. If this is omitted, all targets for the project are built.

- /USER_FILENAME=<file-name>

  This optional qualifier specifies a file that will be created in the user area to store the build results. If this qualifier is not specified, the build result information is returned to the command client as message text. For example:

```
Dimensions>BLD  "REPX:REPX" /BUILD_CONFIG="win 32 buildme;6" /
    AREA="WS_3" /NOCAPTURE /NOBATCH /NOAUDIT
Current status of build job 691 :       SUCCESS
Open this link in a browser for more details
<link>

Operation completed
```

Description

The BLD command:

- Builds targets that are defined for the specified project and that are available at the specified build stage.
- Optionally, collects built targets and creates relationships between them and the sources used to build them.

  By default, the collected targets will be:

  - Created in the development project from which the build request was initiated. The default product-specific Dimensions upload rules will be used to derive Dimensions data format and item type information. For more information about upload rules, refer to the *Process Modeling User's Guide*.
  - Created at the initial lifecycle state, which would normally be associated with the DEVELOPMENT stage. If the lifecycle of the item type of a built target is mapped to build stages (such as UT, ST, and REL), then Dimensions will also promote the built target to the corresponding build stage, and will place the resulting item revision into the corresponding project-stage project.

BLDB – Build Baseline

NOTE This command is not available in the Issue Management–only licensed version of Dimensions.

```
<baseline-spec>
[/AREA=<area-name>]
[/[NO]BATCH]
[/BUILD_CLEAN]
[/BUILD_CONFIG = <build-configuration-name>]
[/BUILD_OPTIONS = (<opt1>=<value1>,<opt2>=<value2>,…)]
[/[NO]CAPTURE]
[/CHANGE_DOC_IDS=(<request1>,<request2>,…)]
[/TARGETS=<targets-list>]
[/USER_FILENAME = <file-name>]
```

Example

```
BLDB "PAYROLL:ACME_2.1_TSTBLN"
   /TARGETS=("aix/foo", "aix/libfoo.so")
   /PARAMETERS=(DMBLDMAKE_OPTIONS="-s -ov")
   /CAPTURE
   /CHANGE_DOC_IDS=(PVCS_CR_1234)
   /USER_FILENAME=D:\temp\bld.log
```

Parameters

- `<baseline-spec>`

  Specifies the baseline to be built.

- `/AREA=<area-name>`

  Specifies the Dimensions area to be used for the build. If this is not specified, all areas associated with the build configuration or configurations are used.

- `/BATCH`

  Specifies that the build is to be run in batch mode (the command does not wait for the build to finish).

  Default: no batch.

- `/BUILD_CLEAN`

  Specifies that the area is to be cleaned of targets before the build process begins.

  Default: no clean.

- `/BUILD_CONFIG=<build-configuration-name>`

  Specifies the build configuration. If this is not specified, all build configurations associated with the Dimensions project are used.

- `/BUILD_OPTIONS=(<opt1>=<value1>,<opt2>=<value2>,…)`

  Specifies any build options.

- `CAPTURE`

  Specifies whether built targets are to be collected. It is not possible to collect default targets when building at the DEVELOPMENT stage.

  Default: no capture.

- /CHANGE_DOC_IDS=(<request1>,<request2>, …)

  <requestN> identifies a request to which the new items created from the built final targets are to be related in-Response-to if required by change management rules.

- /TARGETS=<targets-list>

  Specifies the list of targets to be built. If this is omitted, all targets for the project are built.

- /USER_FILENAME=<file-name>

  This optional qualifier specifies a file that will be created in the user area to store the build results. If this qualifier is not specified, the build result information is returned to the command client as message text. For example:

```
Dimensions>BLD  "REPX:REPX" /BUILD_CONFIG="win 32 buildme;6" /
    AREA="WS_3" /NOCAPTURE /NOBATCH /NOAUDIT
Current status of build job 691 :       SUCCESS
Open this link in a browser for more details
<link>

Operation completed
```

Description

Builds targets from baselines, and optionally collects built targets and creates relationships between the collected targets and sources used to build these targets.

By default, the collected targets will be created in the project from which the build request was issued. The default product-specific Dimensions upload rules will be used to derive Dimensions data format and item type information. For more information about upload rules, refer to the *Process Modeling User's Guide*. The collected targets will be created at the initial lifecycle state.

CA – Create Area

NOTE This command is not available in the Issue Management–only licensed version of Dimensions.

```
<area-name>
[/DESCRIPTION=<area-description>]
/NETWORK_NODE=<node-name>
/DIRECTORY=<HLQ/directory>
[/TYPE=<area-type>]
[/STAGE=<stage-name>]
[/USER=<user-name> [/PASSWORD=<password>]]
[/TRANSFER_SCRIPTS=<script-set>]
[/OWNER=<user-name> or <group-name>]
[/USER_LIST=(<user-or-group>,<another-user-or-group>,...)]
[/STATUS=ONLINE or OFFLINE]
[/FILTER=<area-filter>]
```

Example

```
CA <area-name> /NETWORK_NODE=<host-machine> /DIRECTORY=<area-directory>
    /TYPE=WORK USER_LIST=(<user1>,<user2>,<user3>)
```

Parameters

- `<area-name>`

  Specifies the name of the area. Area names must be unique within the base database.

- `/DESCRIPTION=<description>`

  Optional. Specifies a description for the new area.

- `/NETWORK_NODE=<node-name>`

  Specifies the machine hosting the area.

- `/DIRECTORY=<HLQ/directory>`

  Specifies the directory, or PDS (partitioned data set), where the area is located.

  HLQ is a high-level qualifier; for example, MERVK.WORK. It is a common prefix for all data sets in the area such as MERVK.WORK.C, MERVK.WORK.CBL, and so forth.

- `/TYPE=<area-type>`

  Specifies the type of the area: WORK or DEPLOYMENT.

  Below is a mapping between Dimensions 9 and Dimensions 10 area types:

Dimensions 9	Dimensions 10
Development Area (working location)	Work Area
Managed Development Area	Deployment Area associated with stage DEVELOPMENT
Build Area associated with stage <XXX>	Deployment Area associated with stage <XXX>

- `/STAGE=<stage-name>`

  Applicable only to deployment areas. If the area type is DEPLOYMENT, this qualifier specifies the stage with which the new deployment area is associated.

- /USER=<user-name> [/PASSWORD=<password>]

  Login information for the operating system user account that will own files transferred into the area.

- /TRANSFER_SCRIPTS=<script-set>

  Applicable only to the DEPLOYMENT area type. Specifies the transfer script set. The script set contains a comma-separated list of the names of pre/post/fail transfer scripts in the following format:

  (<pre-script>,<post-script>,<fail-script>)

  If one of the scripts is undefined, CA uses $NONE as a placeholder. The pre-script is executed before an item is transferred into an area, the post script is executed after an item is successfully transferred into an area, and the fail script is executed after a failed transfer of all items into an area.

- /OWNER=<user-name> or <group-name>

  Optional. Specifies the user or group that is to become the owner of the new area. If /OWNER is not specified, the user who created the area is set as its owner. The owner of the area has the right to manage the definition of the area.

- /USER_LIST=(<user-or-group>,<another-user-or-group>,...)

  Applicable only to the WORK area type. Specifies the list of users and/or groups that are granted the right to use this work area. If the user list of an area is empty, any user can specify the area as the working location and can get or check out items into the area and check in items from the area. If the user list of an area is not empty, only the listed user and members of the listed groups can use the work area as a target of get, check-out, and check-in operations.

- /STATUS=ONLINE or OFFLINE

  Applicable only to the DEPLOYMENT area type. Specifies the status of the area. If the area's status is ONLINE, the area may participate in file transfer operations. If the area's status is OFFLINE, the area is automatically excluded from any file transfer operations. If this qualifier is not specified, an area with status ONLINE is created.

- /FILTER=<area-filter>

  Applicable only to the DEPLOYMENT area type. Specifies the name of the area filter to be used when deploying files into this area.

Description

The CA command creates an area definition.

Constraints

To create a work area, you must have the Create Work Areas privilege. To create a deployment area, you must have the Create Deployment Areas privilege.

CAR – Create Archive

NOTE This command is not available in the Issue Management–only licensed version of Dimensions.

```
<archive-id>
/BASELINE=<baseline-spec>
/DEVICE=<device-id>
/TAPE=<tape no.>
/VOLUME=<volume-id>
[/DESCRIPTION=<description>]
[/DIRECTORY=<directory>]
[/[NO]REPORT]
```

Example

```
CAR AA12AB /BASELINE="PRODX:BL12AB" -
   /DEVICE="/dev/rmt0h" /TAPE="aa100" /VOLUME="bb100" -
   /DIRECTORY="/usr/smith/work"-
   /DESCRIPTION="Archive of 12AB - sources"
```

See the *Administrator's Guide* for details.

CBA – Create Build Area

NOTE This command is no longer available; use CA (Create Area) instead.

See the CA command.

CBDB – Register a Base Database Entry

```
/BDB_NAME=<base_db_name>
/PCMS_VER=<dimensions_version>
/CAP_REPLICATE=<project_replication_y/n>
/DB_SERVICE=<base_db_instance>
/NN_NAME=<network_node_name>
[/DESCRIPTION=<base_db_description>]
[/PCMS_ROOT_DIR=<dimensions_root_diectory>]
[/CO_NAME=<contact_name>]
[/SITE_NO=<site_no>]
```

Example

```
CBDB /BDB_NAME=SERENA-PCMS
   /NN_NAME=MACHINE.COMPANY.COM
   /DB_SERVICE=PCMSUDB
   /CAP_REPLICATE=N
   /PCMS_ROOT_DIR="DIMENSIONS ROOT DIRECTORY "
   /PCMS_VER="DIMENSIONS 9.1"
   /DESCRIPTION="DETAILS OF BASE DATABASE FOR NODE MACHINE.COMPANY.COM"
```

This command enables you to register base database entries in an installation's network administration tables. See the *Administrator's Guide* for details.

CBL – Create Baseline

NOTE This command is not available in the Issue Management–only licensed version of Dimensions.

```
<baseline-spec>
[/PART=<part-spec>]
[/TEMPLATE_ID=<template-id>]
[/TYPE=<baseline-type>]
/SCOPE=<scope-name>
[/WORKSET=<project-spec>]
[/ATTRIBUTES=(<attr1>=<value1>,<attr2>=<value2>,…)]
[/LEVEL=<integer>]
[/CHANGE_DOC_IDS=(<request1>,<request2>,…)]
[/[NO]CANCEL_TRAVERSE]
[/[NO]INCLUDE_INFO]
[/[NO]INCLUDE_CLOSED]
```

Example

```
CBL PROD:"R M VERSION 2 FOR HP"
   /TEMPLATE=SOURCE_DOC
   /PART=PROD:"RELEASE MANAGEMENT".AAAB
```

- `<baseline-spec>` comprises:

  `<product-id>:<baseline-id>`

  `<baseline-id>` is the identity to be given to the baseline.

- `/PART=<part-spec>` comprises:

  `<product-id>:<part-id>.<variant>;<pcs>`

  `<variant>` may be omitted if only one exists.

  `<pcs>` is ignored; the current PCS is always used.

- `/TEMPLATE=<template-id>`

  This is the identity of the item or request baseline-template. (The latter type of template was introduced in the Dimensions release 9.1, see the sub-section Request Baseline Templates below for more details.)

  This must be:

  - specified when creating a release-baseline
  - omitted when creating a design baseline.

- `/TYPE=<baseline-type>`

  Specifies the type of baseline being created.

  If omitted, the default type is RELEASE if a `<template-id>` has been specified. (If `/TYPE` is not specified, then by default, a design baseline is created, which is simply a snapshot of the current stage of product development, and is not therefore expected to need an approval lifecycle.)

- /SCOPE=<scope-name>

  If /SCOPE=WORKSET is specified, a project baseline is created. If /SCOPE=PART is specified, or if /SCOPE is omitted, a design part scoped baseline is created. A project baseline is always owned by the product that owns the project.

- /WORKSET=<project-spec>

  comprises:

  <product-id>:<project-id>

  and is optional. If specified, only the items in the project are considered for baselining; otherwise, only the items in the user's current project are considered. See also, Dimensions LCK command (on page 202) concerning comment about locking the project when baselining.

  If /SCOPE=WORKSET is specified, this qualifier specifies the project to be baselined, not the project to be used to constrain part-based revision selection.

- /ATTRIBUTES=(<attr1>=<value1>,<attr2>=<value2>,…)

<attrN>	is the Variable Name defined for one of the user-defined attributes for baselines, which has also been declared as usable for this <product-id> and <baseline-type>.
<valueN>	is the substitution value to be given to this attribute.

- /LEVEL=<integer>

  Optionally, you may elect to restrict the baseline to a given number of levels in the design tree structure.

  The default of 0 (zero) signifies all levels below the design part selected by /PART. For example, 1 signifies that items related to the selected design part only will be processed.

- /CHANGE_DOC_IDS=(<request1>,<request2>,…)

<requestN>	identifies a request to which the new baseline is to be related InResponseTo. If the template specified is a request template, the above request identification will be overridden to specify the list of parent requests that will be used for the CBL command.

- /CANCEL_TRAVERSE or /NOCANCEL_TRAVERSE

  /CANCEL_TRAVERSE halts the traversal of dependent requests. By default, requests that are related as dependent will be processed by the CBL command.

  If /SCOPE=WORKSET is specified, this qualifier controls whether child collections are baselined. The default is /NOCANCEL_TRAVERSE.

- /INCLUDE_INFO or /NOINCLUDE_INFO

  Optionally allow the inclusion of items related to requests via an Info relationship. By default, only items that are related to requests via an InResponseTo relationship will be included.

  If /SCOPE=WORKSET is specified, this qualifier controls whether child collections with INFO relationships are baselined. A child is considered to have an INFO relationship if the relationship does not specify a relative location.

The default is /NOINCLUDE_INFO.

- `[/[NO]INCLUDE_CLOSED]`

  Include closed requests when processing requests for request baselines. The default is /NOINCLUDE_CLOSED—do not include requests.

Request Baseline Templates

Overview

Request baseline templates enable you to specify rules for selecting those requests that will be used as input for creating a baseline. They comprise one or more rules that are made up from the following:

- Request type.
- Request status.
- Baseline status code, which itself comprises one of the following keys:
  - EQS – specified state only.
  - SUP – specified state or next existing state upward.

NOTE Baseline collective codes such as *MADE_OF and *LATEST, which are used in item baseline templates, are not relevant to requests baselines and are thus not provided.

A request baseline template consists only of request template rules – it will not allow the addition of rules using item types. Conversely, an item baseline template does not allow the addition of any request baseline template rules. To enforce this separation, the same template identifier **cannot** be used to create an item baseline template and a request baseline template.

Using Request Baseline Templates with CBL

When a baseline is created using the CBL command specifying a request baseline template and a set of starting parent requests, then all the requests that:

- are related to those parent requests, and
- match the template rules

will be collected together for processing.

The baseline template rules will be processed in exactly the same way that they are for item templates, that is, requests will be selected based on the type, status, and the baseline status code that was specified. For example, if a template had a rule that said that:

- all requests of type PR,
- at status ACCEPTED,
- with the baseline status code EQS

were to be considered, then all the requests of type PR, at the status ACCEPTED only, would be used for inclusion into the baseline.

Once this list of requests has been determined, then **only** those items that have been related to those requests with either an InResponseTo or, optionally, an Info relationship will be included into the baseline. However, because the baseline that is being created is a release baseline, only **one** revision of each item will be included into that baseline. This means, that even though the requests being selected may contain multiple revisions of the same item, the final baseline will only contain one revision of all these possible items.

To ensure that only one revision of an item is included in the final baseline in circumstances where multiple item revisions are related to requests, only the latest item revision will be selected using that item's pedigree. If item revisions are in conflict, that is, no common successor to these items are found, then CBL will fail with an appropriate error message.

When the baseline has been created, the requests that were used to create it will be related as InResponseTo that new baseline.

Constraints

1 Generally, to create a baseline you must have the appropriate management privileges for the baseline's specified top design part that are required to action it from its initial lifecycle state to a new state. However, if a user with the role `PRODUCT-MANAGER` has assigned the top design part role `$ORIGINATOR` to the first transition in the lifecycle for this baseline type, any Dimensions user can create a baseline of this type provided they have a role (any will suffice) on the top design part on which the baseline is being created.

2 To enforce a consistent model of behavior in respect to item baselines, the following additional constraints apply to request baselines:

a Requests that are either at a closed or off-norm lifecycle state will not be processed by the SUP baseline code.

b Only requests in the primary catalog will be processed.

c Only requests related through a dependency relationship, that is, DEPEND, will be included in traversal scans.

d If a request related through a dependency relationship does not fulfill the criteria specified in the request template rules, then that request will be ignored—as will every other request that is a child of that request.

This means, that for a situation where N levels of requests are related together in a chain through dependent relationships, for example, CR_1 → CR_5 → CR_7 → CR_9 → CR_12, if CR_7 fails to match a template rule, then it, and CR_9 and CR_12 will be ignored from any further processing.

e Only requests that are owned by the product on which the baseline is being created will be processed. Any requests owned by other products will be ignored—this includes any children that such requests might have.

f Only parts or items that are:

- owned or
- that have usage relationships to the parent part specified in the CBL command

will be included in the final baseline.

This means that if a request refers to affected parts and/or items that may be out of scope, then these parts and items will be ignored. Out of scope parts and items are parts or items not owned by or related to the parent part or any of its children.

CC – Create Request

```
<product-id>
<request-type>
[/WORKSET=<project-spec>]
[/BASED_ON=<request-id>]
[/DESCRIPTION=<desc-file>]
[/AFFECTED_PARTS=(<part-spec>,<part-spec>,…)]
[/ATTRIBUTES=(<attr1>=<value1>,<attr2>=<value2>,…)]
[/RELATIONSHIP=<rel_name>]
[/[NO]HOLD]
[/ATTACHMENTS=([USER_FILE=<user-file>, FILENAME=<file-id>,
    DESCRIPTION=<description-text>], …)]
[/BASELINE_LIST=(<baseline1>,<baseline2>,…)]
[/[NO]EXCLUSIVE_LOCK]
```

Example

```
CC PROD DR
   /DESC=qrel_subdir.desc
   /AFFECT=PROD:"RELEASE MANAGEMENT".AAAA
   /ATTRIB=(TITLE="QREL Subdir problem",SEVERITY=3)Sub
   /ATTACHMENTS=([USER_FILE=C:\Attachments\Figure1.jpg,
                    FILENAME=Figure1.jpg,DESCRIPTION="first page"])
```

- `<product-id>`

  Specifies the product that is to own the new request.

- `<request-type>`

  Specifies the type of request to be created.

- `/WORKSET=<project-spec>`

  Comprises `<product-id>:<project-id>` and specifies a project with which the request will be associated. If /WORKSET is not specified, the current project is used. When a request is created with an owning project, the Stage ID for the request is set to the initial stage in the Global Stage Lifecycle.

- `/BASED_ON=<request-id>`

  This is used to base (i.e. prime) the creation of the new request on the attributes of the request given by the identity `<ch_doc_id>`.

  If omitted, the new request's data is derived solely from whatever other parameters are specified in this command.

- `/DESCRIPTION=<desc-file>`

  Specifies a file containing the text body to be used as the detailed description of the request.

- `/AFFECTED_PARTS<part-spec>` comprises:

  ```
  <product-id>:<part-id>.<variant>;<pcs>
  ```

`<variant>`	may be omitted if only one exists.
`<pcs>`	is ignored; the current PCS is always used.

  This is used to specify one or more design parts to be related to the new request.

  If /AFFECTED_PARTS is omitted, the product's top design part is related by default.

- /ATTRIBUTES=(<attr1>=<value1>,<attr2>=<value2>,…)

<attrN>	is the Variable Name defined for one of the 220 user-defined attributes for requests, which has also been declared as usable for this <product-id> and <request-type>.
<valueN>	is the substitution value to be given to this attribute.

- /RELATIONSHIP=<rel_name>

  This is valid only if the qualifier /BASED_ON is used. It specifies the relationship type between the newly created request and the base request. The relationship is a bi-directional link with the new request as child and the base request as parent.

- /HOLD

  Specifies that the new request is to be placed on the Held List for possible modification before it is entered into the system.

  The default is /NOHOLD meaning that the new request is saved; that is, it is immediately entered into the system.

NOTE

The <request-id> generated by CC may be referenced as $LAST by a subsequent command within a CMD command-file.

UNIX users only: This must be preceded by a backslash, thus: \$LAST.

- /ATTACHMENTS=([USER_FILE=<user-file>,FILENAME=<file-id>, DESCRIPTION=<description-text>], …)

  Specifies a file, or files, to be attached to the request when it is created.

USER_FILE=<user-file>	is the name of the user file from where the attachment is to be loaded.
FILENAME=<file-id>	is the name of the file to be attached.
DESCRIPTION=<description-text>	is a description of the attachment.

- /BASELINE_LIST=(<baseline1>,…)

  Identifies one or more existing release baselines that the request will be related to (as Affected) when that request enters the system. This qualifier will have the following properties:

  - When a CC command is run without /AFFECTED_PARTS, then the request will be automatically related to the current open PCS of the part owning the baseline. When multiple baselines are specified, these owning parts will also be related as if multiple affected parts had been specified.
  - When a CC command is run with /AFFECTED_PARTS and /BASELINE_LIST specified, then the parts affected will be a merged list of both those parts listed in the /AFFECTED_PARTS qualifier and those parts which own the baselines.

- /EXCLUSIVE_LOCK

  Specifies that the new request will be "locked" against any request (issue) replication "requests" from users located on other replication sites. A locked request is still available for users to work on normally if they are located on the "owning" replication

site. Users on the owning site where that locked request was created can still continue to use that new request normally. The default is /NOEXCLUSIVE_LOCK meaning that the new request can be "requested" from any authorized replication site.

Constraints

This command can be run (by default) by all users. However, a user with the role of PRODUCT-MANAGER can set a Process Model flag to insist that you must have a role (any role will suffice) on the product concerned before you can create requests.

Requests that were created in a held state are not considered to have been "created" as far as Process Models where optional sensitive states or attributes have been set up ("electronic signatures") are concerned. The act of entering them into the system by actioning them out of the held state is considered the "authorization point" for such process models. This also applies to held requests that are updated at the held state (using the command UC) before being actioned on.

CCO – Create a New Contact

```
/CO_NAME=<contact_name>
[/TITLE=<job_title_of_contact>]
[/EMAIL=<e-mail_address_of_contact>
[/ADDRESS=<postal_addrees_of_contact>
[/CONTACT_TYPE=<additional_information>
```

Example

```
CCO /CO_NAME="SERVER MANAGER" -
   /TITLE="SENIOR SERVER MANAGER"-
   /EMAIL=SERVER.MANAGER@COMPANY.COM -
   /ADDRESS="ABBEY VIEW, ST ALBANS" -
   /CONTACT_TYPE=M

CCO /CO_NAME="MAINFRAME MANAGER" -
   /TITLE="MAINFRAME ARCHITECT" -
   /EMAIL=MAINFRAME.MANAGER@COMPANY.COM -
   /ADDRESS="ABBEY VIEW, ST ALBANS" -
   /CONTACT_TYPE=M

CCO /CO_NAME="DB ADMIN" -
   /TITLE="DBA" -
   /EMAIL=DBA.MANAGER@COMPANY.COM -
   /ADDRSS="ABBEY VIEW, ST ALBANS" -
   /CONTACT_TYPE=M
```

This command enables you to add a new contact to an installation. See the *Administrator's Guide* for details.

CCST – Create a New Codeset

```
/CDST_NUMBER=<codeset_number>
/DESCRIPTION=<codeset-description>
```

Example

```
CCST /CDST_NUMBER="2000" -
   /DESCRIPTION="Description - EBCDIC Ireland (Euro)"
```

This command enables you to add a new codeset to an installation. See the *Administrator's Guide* for details.

CCU – Create Customer

NOTE This command is not available in the Issue Management–only licensed version of Dimensions.

```
<name>
/LOCATION=<location>
/PROJECT=<project-spec>
[/COMMENT=<comment>]
[/CONTACT=<contact-details]
```

Example

```
CCU "Brown Finances"
   /LOCATION="Manchester"
   /PROJECT="PAYROLL"
   /CONTACT="Mrs E Green"
```

- `<name>`

  Specifies a name for the customer.

- `/LOCATION=<location>`

  Specifies the customer's physical location.

- `/PROJECT=<project-spec>`

  Specifies the project name.

- `/COMMENT=<comment>`

  for optionally adding more about the customer.

- `/CONTACT=<contact-details>`

  for optionally adding customer contact details.

Description

The Dimensions product allows you to maintain a list of customers and a record of which Dimensions releases have been sent to each customer.

The CCU command enables you to add customers to the list when you are ready to forward a release to that customer.

Constraints

The combination of customer name, location, and project-spec must be unique in the Dimensions database.

CFS – Create a File System

```
/FS_NAME=<file_system_name>
[/DESCRIPTION=<description>]
```

Example `CFS /FS_NAME=NTFS /DESCRIPTION="NT FILE SYSTEM"`

This command enables you to define specific file systems definitions for each registered installation operating system. See the *Administrator's Guide* for details.

CGRP – Create Group

```
<group-name>
[/DESCRIPTION=<description>]
```

Example `CGRP "Contractors" /DESCRIPTION="Contract employees"`

- `<group-name>`

  The name of the group.

- `<description>`

  An optional description for the new group.

Description

The CGRP command creates a group.

Constraints

Only users with the appropriate management privileges can run this command.

CI – Create Item

NOTE This command is not available in the Issue Management–only licensed version of Dimensions.

```
<item-spec>
/PART=<part-spec>
[/ROOT_PROJECT=<project-spec>]
/FILENAME=<file-name>
[/WS_FILENAME=<ws_filename>]
[/COMMENT=<comment text>]
[/FORMAT=<format>]
[/USER_FILENAME=<user-filename>]
[/[NO]KEEP]
[/DESCRIPTION=<description>]
[/EXTRA_VARIANTS=(<var1>,<var2>,…)]
[/ATTRIBUTES=(<attr1>=<value1>,<attr2>=<value2>,…)]
[/CHANGE_DOC_IDS=(<request1>,<request2>,…)]
[/STATUS=<status>]
[/WORKSET=<project-spec>]
[/CODEPAGE=<code-page>| DEFAULT]
[/CONTENT_ENCODING=<file-encoding>]
[/NOMETADATA]
```

Example

```
CI PROD:"QUERY RELEASE"-SRC
   /FILENAME=qr.c
   /USER_FILE=qr.c
   /WS_FILENAME="src/qr.c"
   /PART=PROD:"RELEASE MANAGEMENT".AAAA
   /EXTRA=AAAB
   /COMMENT="created for CRB 66"
```

- `<item-spec>`

  Comprises:

  ```
  <product-id>:<item-id>.<variant>–<item-type>;<revision>
  ```

`<item-id>`	identifies the new item within the product.
`<variant>`	if omitted, the default specified when the product was defined is used.
`<revision>`	defaults to 1, if omitted

NOTE If auto-id generation is enabled for this item type (see the related document *Process Modeling User's Guide*), then the Item-id must **not** be specified.

- /PART=<part-spec>

  Identifies the design part that will own the item.

  It comprises: <product-id>:<part-id>.<variant>;<pcs>

<product-id>	must be the same as that for <item-spec>
<variant>	may be omitted if only one exists.
<pcs>	is ignored. The item is always OWNED by the current PCS.

- /ROOT_PROJECT=<project-spec>

  Comprises:

  <product-id>:<project-id>

  This optionally specifies the root project. Use this when the current project set via SCWS (or the project specified by the /WORKSET qualifier) occurs in more than one project tree.

- /FILENAME=<file-name>

  Specifies the name of the file which is to contain the **item** in the item library. It should (but need not necessarily) be of the form: <name>.<type> (see <format> below for the use of <type>). <file-name> **may** include subdirectory name(s) but must **not** specify an absolute path; thus:

  - **UNIX** ⇒ <file-name> **must not** begin with **/** (forward slash).
  - **Windows** ⇒ <file-name> **must not** begin with **** (backslash) or <drive:>.

  If /ROOT_PROJECT is used to specify a the root project, /FILENAME is interpreted in the scope of that project.

- /WS_FILENAME=<ws_filename>

  Identifies the relative path (directory plus file name) of the file to be used when the item created here is subsequently checked out or gotten as an item from the current project.

<ws_filename>	may include sub-directory name(s), but must not specify an absolute path, as above.

- /COMMENT=<comment text>

  Comment text to explain the reason for the creation of this item revision. The comment text can be up to 1978 characters long, and can be made available within the item header.

- /FORMAT=<format>

  A user with the role of TOOL-MANAGER may have defined (DDF) and assigned (ADF) a list of valid data formats to particular item types. Uses for such a list include:

  - Validation on item creation, specifying file types for the Dimensions desktop client application and specifying MIME types for the Dimensions web client.
  - Dimensions Make. The format field allows items of the same item type to be distinguished on the basis of language (for program sources) or of execution

platform (for executable program files) and so on. In this way different build processes can be defined for items of the same type but different format. For example, an item of type SRC and format C will be compiled using a C compiler, whereas an item of the same type but format PAS will be complied using a Pascal compiler.

If a list of valid file formats have been assigned to an item type, then the use of one of those formats is compulsory when creating items of that type; whereas, if a list of format types has not been assigned, then any format can be used at the time of item creation, even one not defined by a user with the role of TOOL-MANAGER.

If <type> is specified in <file-name> then <format> will default to that and does not need to be explicitly specified (but if <file-name> does **not** include <type>, then <format> **cannot** be omitted).

- /USER_FILENAME=<user-filename>

  Specifies the name of the file which holds the item in the user-area.

  If omitted, then either a skeletal document (from a format-template) or a null file is created.

- /KEEP

  Specifies that the <user-filename> which is normally deleted once the item has been placed under Dimensions control, is to be left intact.

- /DESCRIPTION=<description>

  This is optional text that will be displayed by Dimensions applications and reports.

  Dimensions supplies default text if this is omitted.

- /EXTRA_VARIANTS=<varN>

  Identifies another variant of the OWNER design part which also USES the item.

- /ATTRIBUTES=(<attr1>=<value1>,<attr2>=<value2>,…)

  <attrN> is the Variable Name defined for one of the user-defined attributes for items, which has also been declared usable for the <product-id> and <item-type> specified in **<item-spec>**.

  <valueN> is the value to be given to this attribute.

- /CHANGE_DOC_IDS=(<request1>,<request2>,…)

  <requestN> identifies a request to which the new item is to be related in-Response-to.

- /STATUS=<status>

Specifies the state of the created item.

NOTES

If you specify a state that has been set to 'sensitive' in your process model, the sensitive state is ignored and the item is created at the initial state of the lifecycle. For example, assume that you have three states, *A, B*, and *C*, and *B* has been set to sensitive. If you try to create a new item at status B, it is created instead at the initial state, *A*.

The only equivalent to this parameter in interactive mode is CI followed by AI. The status, if specified, must be one which would be valid if AI had been used separately. If omitted, the initial state (in the lifecycle defined for `<item-type>`) is assigned.

- `/WORKSET=<project-spec>` comprises:

  `<product-id>:<project-id>`

  and is optional. If specified, the new item will be placed in that project. If unspecified, the new item will be placed in the user's current project.

- `/CODEPAGE=<code-page>|DEFAULT`

  Specifies the *code page* to be associated with the item. The code page defines the method of encoding characters. It encompasses both the different ways characters are encoded on different platforms (EBCDIC on z/OS and ASCII on Windows and UNIX) and differences between human languages. Every item in Dimensions has a code page associated with it, this being defined or derived for the connection setting or an individual item.

  The `/CODEPAGE` parameter defaults to the code page specified when the connection between the database server and the logical node on which the user file resides was created. Whenever the item moves between platforms, for example, on a check-out from the mainframe to a PC, if the code page for the target platform is different to the item's code page, Dimensions automatically converts the item.
  `/CODEPAGE` is relevant only for text files, because whenever a text file is checked out or gotten, it must be in the right code page for the target platform, so that it displays correctly. Binary files are moved between platforms with no conversion.

  For further details concerning code pages and logical nodes see the *Distributed Development Guide*.

  You are advised to let the parameter default to the code page for the item type or the platform.

  The `/CODEPAGE` options available are:

`<code-page>`	Specify one of the code page values listed in the text file `codepage.txt`, located on your Dimensions server in the `codepage` subdirectory of the Dimensions installation directory. This file also provides more information about translation between code pages.
`DEFAULT`	Use the code page specified for the target node connection.

- `<file-encoding>`

  Specifies the content encoding for new item revisions to be created. Supported encodings are the Microsoft codepages, the ISO-8859 variants (1–10), UTF-8, UTF-16, UTF-16BE, UTF-16LE, UTF32, UTF32BE, and UTF32LE.

- `/NOMETADATA`

  This parameter disables creation and usage of metadata files in the local work area.

Note on Areas

The CI command results in a new revision being created in the project. If DEVELOPMENT deployment areas are in use, CI automatically updates the corresponding areas.

Constraints

Generally, this command can be run by users who have one of the roles required to action the item from the initial lifecycle state to a new state. However, if a user with the role of `PRODUCT-MANAGER` has assigned `$ORIGINATOR` role to the first transition in the lifecycle for this item type, any Dimensions user can create an item of this type provided they have a role (any will suffice) on the design part owning the item.

A user with the role `PRODUCT-MANAGER` can also create items where no appropriate roles have yet ben allocated. This is to facilitate quicker migration of files.

IMPORTANT! The maximum size of any item in the database is 2GB.

CINS – Register a Database Instance Entry

```
/DB_SERVICE=<base_db_instance>
/NN_NAME=<network_node_name>
[/DB_TRANSPORT=<db_spefic_transport>]
[/DB_NAME=<db_name>]
[/DB_TWO_TASK=<Oracle_specific_remote_connection_string>]
[/DB_HOME_DIR=<home_directory_of_db_instance>]
[/DB_ACTIVE=<reseved_(actively_used_y/n)>]
[/CO_NAME=<contact_name>]
```

Example

```
CINS /NN_NAME=MACHINE.COMPANY.COM
   /DB_SERVICE=PCMSUDB
   /DB_HOME_DIR=.
   /DB_NAME=PAYROLL
```

This command enables you to register database instances in an installation's network administration tables. See the *Administrator's Guide* for details.

CIU – Cancel Item Update

NOTE This command is not available in the Issue Management–only licensed version of Dimensions.

```
<item-spec>
[/ROOT_PROJECT=<project-spec>]
[/FILENAME=<file-name>]
[/WORKSET=<project-spec>]
[/[NO]KEEP]
[/NOMETADATA]
```

Example `CIU PROD:"QUERY RELEASE".AAAA-SRC;1`

- `<item-spec>`

  Comprises:

  `<product-id>:<item-id>.<variant>-<item-type>;<revision>`

`<item-id>`	may be omitted if `<file-name>` is specified.
`<variant>`	may be omitted if only one exists.
`<revision>`	may be omitted if you have checked out only one revision of this item.

- `/ROOT_PROJECT=<project-spec>`

  Comprises:

  `<product-id>:<project-id>`

  This optionally specifies the root project. Use this when the current project set via SCWS (or the project specified by the /WORKSET qualifier) occurs in more than one project tree.

- `/FILENAME=<file-name>`

  Specifies the name of the project file name.

  The project file name identifies the relative path (directory plus file name) from the working location of the file to be used when the item is checked out from the current project. The project file name for the same item may *differ* between projects; for example, `src/hello.c`, `hello.c`, or `src/build/hello.c`.

  It may be omitted if `<item-id>` is specified.

- `/WORKSET=<project-spec>` comprises:

  `<product-id>:<project-id>`

  A project is normally specified on item check out, so it would not normally need to be specified on cancel item update.

  If not specified, the user's default project will be used.

- /KEEP or /NOKEEP

  /KEEP prevents CIU from deleting the extracted (checked out) file. This is the default. Specify /NOKEEP to delete the extracted file.

- /NOMETADATA

  This parameter disables creation and usage of metadata files in the local work area.

Constraints

This command can be run only by the user who checked out the item with the EI command, or by a user with the appropriate management privileges.

CLCA – Create Library Cache Area

NOTE This command is not available in the Issue Management–only licensed version of Dimensions.

```
<area-name>
[/DESCRIPTION=<area-description>]
/NETWORK_NODE=<node-name>
/DIRECTORY=<HLQ/directory>
[/USER=<user-name> [/PASSWORD=<password>]]
[/OWNER=<user-name> or <group-name>]
[/STATUS=ONLINE or OFFLINE]
```

Example

```
CLCA <area-name> /NETWORK_NODE=<host-machine> /DIRECTORY=<area-
    directory>
```

Parameters

- `<area-name>`

  Specifies the name of the area. Area names must be unique within the base database.

- `/DESCRIPTION=<description>`

  Optional. Specifies a description for the new area.

- `/NETWORK_NODE=<node-name>`

  Specifies the machine hosting the area.

- `/DIRECTORY=<HLQ/directory>`

  Specifies the directory, or PDS (partitioned data set), where the area is located.

  HLQ is a high-level qualifier; for example, MERVK.WORK. It is a common prefix for all data sets in the area such as MERVK.WORK.C, MERVK.WORK.CBL, and so forth.

- `/USER=<user-name> [/PASSWORD=<password>]`

  Login information for the operating system user account that will own files transferred into the area.

- `/OWNER=<user-name> or <group-name>`

  Optional. Specifies the user or group that is to become the owner of the new area. If /OWNER is not specified, the user who created the area is set as its owner.

- `/STATUS=ONLINE or OFFLINE`

  Specifies the status of the library cache area. If the area's status is ONLINE, the area may participate in file transfer operations. If the area's status is OFFLINE, the area is automatically excluded from any file transfer operations. If this qualifier is not specified, an area with status ONLINE is created.

Description

The CLCA command creates a library cache area definition.

A library cache area is a named location on a network node, which must have a dimensions listener running.

The purpose of a library cache area is to improve file fetch performance by caching file contents on a network node that is "close" to the user's computer. This avoids transferring the file repeatedly over a potentially slow connection between the item library node and the user's network.

Constraints

To create an area, you must have the Create Library Cache Areas privilege.

CMB – Create Merged Baseline

NOTE This command is not available in the Issue Management–only licensed version of Dimensions.

```
<new-baseline-spec>
/PART=<part-spec>
[/TYPE=<baseline-type>]
/BASELINE_LIST=(<baseline1>,<baseline2>,…)
 [/ATTRIBUTES=(<attr1>=<value1>,<attr2>=<value2>,…)]
```

Example

```
CMB PROD:"MERGED VER 2A FOR HP"
   /PART=PROD:"RELEASE MANAGEMENT".AAAB
   /BASE=("QUERY RELEASE MOD 2A","R M VERSION 2 FOR HP")
```

- `<new-baseline-spec>` comprises:

  `<product-id>:<baseline-id>`

`<baseline-id>`	is the identity to be given to the merged baseline being created.

- `/PART=<part-spec>` comprises:

  `<product-id>:<part-id>.<variant>;<pcs>`

`<variant>`	may be omitted if only one exists.
`<pcs>`	is ignored; the current PCS is always used.

- `/TYPE=<baseline-type>`

  Specifies the type of the merged baseline being created.

  If omitted, `RELEASE` is used as a default type.

- `/BASELINE_LIST=(<baseline1>,…)`

  Identifies one or more existing release-baselines (usually at least two), whose items are to be merged into the new baseline. All these baselines must be for the same product as is specified for the merged baseline; so each `<baselineN>` can be specified either as `<baseline-spec>` or simply as `<baseline-id>` i.e. the syntax is:

  `[<product-id>:]<baseline-id>`

`<product-id>`	can be omitted (the colon (:) also being omitted), but if present must be the same as that which was specified in `<new-baseline-spec>`.
`<baseline-id>`	is the identity of one of the existing baselines to be used in the creation of the merged baseline.

- /ATTRIBUTES=(<attr1>=<value1>,<attr2>=<value2>,…)

  sets values for one of more attributes of the new baseline, where each **<attrN>** is the Variable Name defined for one of the user-defined attributes for baselines, which has also been declared as usable for this <product-id> and <baseline-type>, and <valueN> is the substitution value to be given to this attribute.

Description

1 This command creates a new baseline, using <part-spec> as the top design part, to include exactly the same list of design parts (i.e. to have the same scope) as a baseline created by the CBL command would have; but initially this new baseline contains no items.

2 The baselines in /BASELINE_LIST are then considered in turn **in the order specified in that list**; and each of the items in each baseline is checked as follows and ignored:

- if it is not a relevant item in the new baseline
- if any revision of the same item has already been added to the new baseline.

If these checks are passed, each item revision is added to the new baseline.

This continues until all items in all the baselines in the list have been dealt with.

This processing means that, for any item in the merged baseline, the revision added will be the one found in the first baseline in the list which contains that item. Therefore, in order to obtain a merged baseline with satisfactory contents, it will generally be necessary to list the input baselines in increasing order of antiquity: the most recent baselines first and the oldest last.

Merged baselines can be useful in several different ways, and the following is just one possibility (illustrated in the command example given above). A complete system or subsystem is baselined, tested and released. Later a component of it is modified, and its design parts are baselined by themselves and tested. However, the Dimensions function **REL – Release** must always be executed from a single baseline. In order to re-release the same system with just that component modified, these two baselines are merged, with the later baseline first in the baseline-list. A fresh baseline of the whole system might have introduced other modifications done elsewhere in the meantime, which it is undesirable to include in the present re-release.

NOTE Dimensions collates the merged baseline completely automatically according to the specification above; it is entirely the Dimensions user's responsibility to specify baseline-lists that will create sensible and meaningful merged baselines. In any case, the contents will probably not be readily traceable to a consistent set of template rules, and as a reminder of this, it is wise to use some distinctive naming convention for the baseline-ids of merged baselines.

Constraints

Each of the input baselines, as specified in /BASELINE_LIST, must be either a release-baseline which was created using a template with **no** *ALL rules, or else an earlier merged baseline or a revised baseline. They must all be baselines for (any design structure within) the same product as is specified for the new baseline.

The new baseline-id must be unique within the product.

Generally, to create a merged baseline you must have one of the roles for the top design part in the new merged baseline that are required to action the merged baseline from its initial lifecycle state to a new state. However, if a user with the role of PRODUCT-MANAGER has assigned $ORIGINATOR role to the first transition in the lifecycle for this baseline type, any Dimensions user can create a merged baseline of this type provided they have a role (any will suffice) on the top design part on which the merged baseline is being created. There are no role requirements for the top-level design part of any input baselines.

CMD – Execute Dimensions Command File

```
<command-filename>
[/LOGFILE=<log_filename>]
```

Example `CMD items.setup /log=items_setup.log`

- `<command-filename>`

  Specifies the name of a file containing commands which are to be executed (see Introduction on page 16 for more information).

- `/LOGFILE=<log_filename>`

  Specifies a log file name into which to redirect the output from a command script. If no `/LOGFILE` parameter is specified, all output will be logged to the screen.

NOTE

CMD cannot be run from Dimensions for z/OS.

CMD is foreground replacement for the background XCMD command available in previous versions of Dimensions.

CMP – Compare Structures or Baselines

NOTE CMP cannot be run from Dimensions for z/OS.

```
<file-name>
[/PART=<part-spec> or /BASELINE=<baseline-spec>]
[<file-name> [/PART1=<part-spec> or /BASELINE1=<baseline-spec>]]
```

Example

```
CMP * /BASELINE=PROD:"R M VERSION 2 FOR HP" -prm3_01.export
   /PART1=PROD:"RELEASE MANAGEMENT".AAAB
```

- `<file-name>`

  Specifies where the first or second exported structure is stored, when `<part-spec>` or `<baseline-spec>` is not specified.

  Otherwise, `<file-name>` specifies the name of the file where the exported structure is to be stored. If specified as an asterisk (*), then a temporary file is created which is deleted at the end of the operation.

  If no file name is specified, `compare.out` is created by default.

- `/PART=<part-spec>`

  Specifies the top design part of the current product structure which is to be included in the export file.

  It comprises: `<product-id>:<part-id>.<variant>;<pcs>`

`<variant>`	must be specified, even if only one variant exists.
`<pcs>`	is ignored; the current PCS is always used.

- `/BASELINE=<baseline-spec>`

  Specifies the baseline on which the structure to be included in the export file is to be based.

  It comprises: **`<product-id>:<baseline-id>`**

  If the second `<file-name>` is not specified, the specified structure is exported but no report is generated.

Constraints

This command can be run only by the user initiating the report who must have a valid role for the top design part in the structure to be reported.

CNC – Create a Network Node Connection

```
/SERVER_NAME=<server_node_name>
/CLIENT_NAME=<client_node_name>
/NWO_NAME=<network_object_name>
[/FS_NAME=<file_system_name>]
[/CDST_NUMBER=<code_set_number>]
[/DIRECT_FILE_COPY]
[/FILE_COMPRESSION]
```

This command enables you to add a new network node connection to an installation. For details, see the *Administrator's Guide*.

CNDO – Create a Node Object

```
/NN_NAME=<network_node_name>
/NWO_NAME=<network_object_name>
```

This command enables you to add a new node object to an installation. See the *Administrator's Guide* for details.

CNN – Create a Network Node

```
/NN_NAME=<network_node_name>
/OS_NAME=<operating-system-name>
/LOGICAL=<y|n>
[/PHYSICAL_NAME=<physical_node_name>]
[/CO_NAME=<contact-name>]
[/DESCRIPTION=<description>]
[/RSD_NAME=<resident_software_definition>]
```

Example

```
CNN /NN_NAME=MACHINE.COMPANY.COM
   /LOGICAL=N
   /OS_NAME=XP
   /DESCRIPTION="SOURCES HELD ON DFS DIMENSIONS XP SERVER"

CNN /NN_NAME=TEST_MACHINE.COMPANY.COM
   /LOGICAL=N
   /OS_NAME=XP

CNN /NN_NAME="MY_MAINFRAME"
   /PHYSICAL_NAME=MF390.COMPANY.COM
   /LOGICAL=Y
   /OS_NAME=MVS
   /DESCRIPTION="MAINFRAME SERVER SITUATED AT HEAD OFFICE"
```

This command enables you to create new physical or logical nodes. See the *Administrator's Guide* for details.

CNWO – Create a Network Object

```
/NWO_NAME=<network_object_name>
/PROTOCOL=<communication_protocol>
[/DESCRIPTION=<description>]
[/PROCESS=<network_object_process_name>]
```

Example

```
CNWO /NWO_NAME=PCMS_SDP
   /PROTOCOL=SDP
   /DESCRIPTION="NETWORK OBJECT USING STANDARD DIMENSIONS PROTOCOLS"
```

This command enables you to add a new network object to an installation. See the *Administrator's Guide* for details.

COS – Create an Operating System

```
/OS_NAME=<operating_system_name>
/CASE=<operating_system_case_convention>
/LIB_PROTECTION=<type_of_library_protection>
```

Example

```
COS /OS_NAME=XP
   /CASE=NN
   /LIB_PROTECTION=RWX,RX,R
```

This command enables you to add a new operating system to an installation. See the *Administrator's Guide* for details.

CP – Create Design Part

```
<part-spec>
/CATEGORY=<category-name>
/PARENT_PART=<parent-part-spec>
/DESCRIPTION=<description>
[/ATTRIBUTES=(<attr1>=<value1>,<attr2>=<value2>,…)]
```

Example

```
CP PROD:"RELEASE MANAGEMENT"
   /PARENT=PROD:PROD -
   /CAT=SUBSYSTEM
   /DESC="Release Support Sun Version"
```

- `<part-spec>` comprises:

  `<product-id>:<part-id>.<variant>;<pcs>`

  `<variant>` if omitted, the default (specified when the product was defined) is used.

  `<pcs>` if omitted, the PCS for new variants (specified when the product was defined) is used.

- `/CATEGORY=<category-name>`

  Specifies the category of design part.

- `/PARENT_PART=<parent-part-spec>`[1] comprises:

  `<product-id>:<part-id>.<variant>;<pcs>`

  `<variant>` may be omitted if only one exists.

  `<pcs>` is ignored; the current PCS is always used.

- `/DESCRIPTION=<description>`

  This is mandatory text to describe the function of the design part.

- `/ATTRIBUTES=(<attr1>=<value1>,<attr2>=<value2>,…)`

  `<attr1>` is the Variable Name defined for one of the user-defined attributes for design parts, which has also been declared as usable for this `<product-id>` and `<category-name>`. `<valueN>` is the substitution value to be given to this attribute.

Constraints

Only users with the appropriate management privileges can run this command.

CPV – Create Design Part Variant

```
<part-spec>
/NEW_VARIANT=<new-variant>
[/PARENT_VARIANT=<parent-variant>]
[/DESCRIPTION=<description>]
[/ATTRIBUTES=(<attr1>=<value1>,<attr2>=<value2>,…)]
```

Example

```
CPV PROD:"RELEASE MANAGEMENT".AAAA
   /NEW_VAR=IBM
   /DESC="Release Support - IBM Version"
```

- `<part-spec>`

  Specifies an already existing design part. It comprises:

  `<product-id>:<part-id>.<variant>;<pcs>`

  `<variant>` may be omitted if only one variant has existed up to now.

  `<pcs>` is ignored; the current PCS is always used.

- `/NEW_VARIANT=<new-variant>`

  Specifies the identity of the new variant, e.g**.** AAAB**.**

- `/PARENT_VARIANT=<parent-variant>`[1]

  Specifies the variant code of the parent design part to which the new variant is to be related.

  It may be omitted if the father design part has only one variant.

- `/DESCRIPTION=<description>`

  This is optional text describing the function of the variant.

  It defaults to the description associated with `<part-spec>`, if it is omitted.

- `/ATTRIBUTES=(<attr1>=<value1>,<attr2>=<value2>,…)`

  `<attrN>` is the Variable Name defined for one of the user-defined attributes for design parts, which has also been declared as usable for this `<product-id>` and design part's category.

  `<valueN>` is the substitution value to be given to this attribute for the new variant.

Constraints

Only users with the appropriate management privileges can run this command.

CRB – Create Revised Baseline

NOTE This command is not available in the Issue Management–only licensed version of Dimensions.

```
<new-baseline-spec>
/BASELINE1=<existing-baseline-spec>
[/TYPE=<baseline-type>]
[/UPDATE_CHANGE_DOC_IDS=(<ucd1>,<ucd2>,…)]
[/REMOVE_CHANGE_DOC_IDS=(<rcd1>,<rcd2>,…)]
[/CANCEL_TRAVERSE]
[/WORKSET=<project-spec>]
[/ATTRIBUTES=(<attr1>=<value1>,<attr2>=<value2>,…)]
```

Example

```
CRB PROD:"REVISED RM VER 2B FOR HP"
   /BASE=PROD:"MERGED R M VER 2A FOR HP"
   /UPDATE=(PROD_DR_25, PROD_DC_16)
   /REMOVE=PROD_DC_16
   /CANCEL_TRAVERSE
```

- `<new-baseline-spec>` comprises:

  `<product-id>:<baseline-id>`

  `<baseline-id>` is the identity to be given to the revised baseline being created.

- `/BASELINE1=<existing-baseline-spec>` comprises:

  `<product-id>:<baseline-id>`

  `<product-id>` must be the same as that for `<new-baseline-spec>`.

  `<baseline-id>` is the identity of the existing release baseline to be used to create the revised baseline.

- `[/TYPE=<baseline-type>]`

  Specifies the type of the revised baseline being created.

  If omitted, the type will be the same as that of **<existing-baseline-spec>**.

- `/UPDATE_CHANGE_DOC_IDS=(<ucd1>,…)`

  Identifies one or more request trees (see Description below) within which there are item revisions related `in-Response-to`. Each such revision replaces the revision of the same item which was previously in the baseline. If there was no such item already in the baseline, then the in-response-to revision is now added, *provided that* it is a relevant item (i.e. falls within the scope of the baseline).

  When multiple revisions of the same item are `in-Response-to` one or more requests, the latest revision by pedigree will be selected. If two or more item

revisions are on different branches, CRB will issue a warning and continue processing without changing the revision of that item in the baseline.

NOTE For those requests that result in an item revision being added to the revised baseline, an `in-Response-to` relationship will also be automatically created between that baseline and the appropriate request.

These relationships, however, will only be created if that request has actually been used to revise the baseline. For example, if two requests with exactly the same relationships are used to revise the baseline, then only the one which is actually used by CRB to generate the new baseline will be related.

- `/REMOVE_CHANGE_DOC_IDS=(<rcd1>,…)`

  Identifies one or more request trees (see Description below) within which there are item revisions related as `Affected`. Each such revision, if it is found in the revised baseline, it is deleted from it. Although each of the above qualifiers `/UPDATE` and `/REMOVE` is optional, at least one of them must be specified.

NOTE For those requests that result in an item revision being removed from the revised baseline, an `Affected` relationship will automatically be created between that baseline and the appropriate request.

These relationships, however, will only be created if that request has actually been used to revise the baseline. For example, if two requests with exactly the same relationships are used to revise the baseline, then only the one which is actually used by CRB to generate the new baseline will be related.

- `/CANCEL_TRAVERSE`

  Indicates that, in the above processing for `/UPDATE` and `/REMOVE`, traversal of tree structures is not to be performed, and that only the requests cited in the lists are to be inspected for item revisions that have been related, respectively, in-Response-to and as `Affected`.

  If omitted, the processing is of tree-structures as mentioned above (and explained below).

- `/WORKSET=<project-spec>` comprises:

  `<product-id>:<project-id>`

  This optionally specifies the project to be used for this command: failing this, the user's current project will be taken. When an item has been selected for addition to the baseline, the project file name will be taken from this project if another revision is not already in the project.

- `/ATTRIBUTES=(<attr1>=<value1>,<attr2>=<value2>,…)`

  sets values for one of more attributes of the new baseline, where each `<attrN>` is the Variable Name defined for one of the user-defined attributes for baselines, which has also been declared as usable for this `<product-id>` and `<baseline-type>`, and `<valueN>` is the substitution value to be given to this attribute.

Description

1 This command creates a new baseline as an exact copy of an existing baseline. So not only will the new baseline have the same top design part, but *it will also have the same list of included design parts* (i.e. it will have the same scope) as the existing baseline.

NOTE A new baseline created by the CBL command, with the same top design part, might well have a different scope, because the breakdown and/or usage relationships in the design structure could now be different.)

2 The /UPDATE list is processed.

- A candidate list of requests is constructed from the specified list. Each specified request is considered as the head of a family tree of requests with relationships to it in the Dependent class.

NOTE The number of requests is limited by the maximum command-line-list length of 16383 characters. The number of requests this translates into depends on the character lengths of the baseline and product name specifications used. If these specification names correspond to the maximum character lengths allowed by Dimensions, then the request limit will be 496, whereas, if they are shorter, the request limit will correspondingly be greater than 496.

- The specified request and its Dependent-related children are added to the candidate list, then any Dependent-related children of these (i.e. grandchildren) are added (if not already present), and so on until all progeny have been included. (The reason some requests could be already in the candidate list is that they could belong to more than one family tree: i.e. they can be Dependent-related to more than one parent.) Thus the candidate list contains the whole population of requests in all the tree structures specified by the /UPDATE list; unless the /CANCEL_TRAVERSE option was used, in which case the candidate list is just the specified list itself.

3 All the item revisions related as In-Response-To(R) any of the requests in the candidate list are added to the baseline, provided that the items are within the baseline's scope, and that those revisions are neither checked out nor suspended. During this process, any revisions of those same items, which were already in the baseline, are displaced and are removed from it. Thus these In-Response-To revisions can be a combination:

- both of new additions to the baseline (where the item was not already included);
- and of substitutions for item revisions which were already in the baseline.

4 The /REMOVE list is processed. A new candidate list of requests is constructed from this specified list in exactly the same way as was described above for the /UPDATE list.

5 All item revisions related as Affected(A) in any of the requests in this candidate list, and which are still included in the baseline, are deleted from it.

(If any, or even all, of these revisions are not found in the baseline, they are simply ignored — no message is issued; but see "Error Conditions" below for the occasions when all the processing, both /UPDATE and /REMOVE, achieves nothing.)

The principal purpose of the /REMOVE processing is to clear out from the baseline those items which are now redundant, i.e. those for which no In-Response-To

replacement was required. Thus the /REMOVE qualifier is likely to be needed less frequently, because the normal process of removing superseded revisions has already been achieved in the /UPDATE processing. It follows that, when /REMOVE is used, it is quite likely that the requests cited will be the same as (some of) those in the /UPDATE list, but this is not a constraint.

NOTE The revised baseline thus created is likely to be less self-consistent than one created normally using a baseline-template. For example, the original baseline may include some item revisions by using a *BUILT template rule. Also, the sources, from which that revision was built, could have been displaced from the revised baseline, but the original built revisions will remain in the baseline, unless they too have been specifically displaced by the /UPDATE or /REMOVE processing. As a reminder of this potential hazard, it is wise to use some distinctive naming convention for the baseline-ids of revised baselines.

Error Conditions

If any of the revisions, due to be deleted as a result of the /REMOVE processing, is the same as one which was added or substituted as a result of the /UPDATE processing, this conflict is reported as an error and the revised baseline is not created.

If, on completion of both /UPDATE and /REMOVE processing, it is found that no change has been effected at any point (i.e. the revised baseline would in fact be identical in content to the existing baseline), this is reported as an error, and Dimensions automatically deletes the new baseline.

Constraints

The existing baseline must be either a release baseline which was created using a template with **no** *ALL rules, or else an earlier revised or merged baseline.

The new baseline-id must be unique within the product.

Generally, this command can be run by users who have one of the roles for the top design part in the existing baseline (which will also be the top part for the new baseline) that are required to action the merged baseline from its initial lifecycle state to a new state. However, if a user with the role of PRODUCT-MANAGER has assigned $ORIGINATOR role to the first transition in the lifecycle for this baseline type, any Dimensions user can create a revised baseline of this type provided they have a role (any will suffice) on the top design part on which the revised baseline is being created.

CRSD – Create a Resident Software Definition

NOTE This command is not available in the Issue Management–only licensed version of Dimensions.

```
/RSD_NAME=<name_RSD>
/RSD_VERSION=<version_number_of_RSD>
[/RSD_DATA=<RSD_data_details>]
```

This command enables you to register an installation Resident Software Definition (RSD) to be associated with the build environment in which Dimensions Make operates. See the *Administrator's Guide* for details.

CUSR – Register User

```
<user-id>
[/WORKSET=<project-spec>]
[/[NO]PASSWORD_SAVE]
[/ATTRIBUTES=(site=<site>,
    group_id=<group-id>,Dept=<dept>,
    full_name=<full-name>,phone=<phone>,
    <attribute-id>=<value>,email_addr=<email-addr>)]
```

Description

This command is the same as UREG.

For details, see the *Serena Dimensions CM Administrator's Guide*.

CVS – Create Variant Structure

```
<part-spec>
/NEW_VARIANT=<new-variant>
[/PARENT_VARIANT=<parent-variant>]
[/DESCRIPTION=<description>]
[/ATTRIBUTES=(<attr1>=<value1>,<attr2>=<value2>,…)]
```

Example

```
CVS PROD:"RELEASE MANAGEMENT".AAAA
   /NEW_VAR=IBM
   /DESC="Release Support - IBM Version"
```

- `<part-spec>`

  Specifies an already existing design part. It comprises:

  `<product-id>:<part-id>.<variant>;<pcs>`

  `<variant>` may be omitted if only one variant has existed up to now.

  `<pcs>` is ignored; the current PCS is always used

- `/NEW_VARIANT=<new-variant>`

  Specifies the identity of the new variant, e.g**.** AAAB**.**

- `/PARENT_VARIANT=<parent-variant>`[1]

  Specifies the variant code of the parent design part to which the new variant is to be related.

  It may be omitted if the parent design part has only one variant.

- `/DESCRIPTION=<description>`

  This is optional text describing the function of the variant.

  It defaults to the description associated with `<part-spec>`, if it is omitted.

- `/ATTRIBUTES=(<attr1>=<value1>,<attr2>=<value2>,…)`

NOTE To use the `/ATTRIBUTES` qualifier with the CVS command, the attributes specified must be valid for every design part in the structure to be copied. If the attributes do not meet this requirement, then either CPV commands must be entered individually or UP commands must be issued after issuing a CVS command without the `/ATTRIBUTES` qualifier.

`<attrN>` is the Variable Name defined for one of the user-defined attributes for design parts, which has also been declared as usable for this `<product-id>` design part's category.

`<valueN>` is the substitution value to be given to this attribute for the new variant.

NOTE CVS is similar to CPV except that it creates a whole new part variant structure (by including every variant in the structure from the part you specify) i.e. it is the equivalent of a whole series of CPVs.

Roles may then be assigned to specific design part variants.

Constraints

Only users with the appropriate management privileges can run this command.

This command fails if the structure already contains a variant.

CWSD – Create Project Directory

NOTE This command is not available in the Issue Management–only licensed version of Dimensions.

```
<directory-path>
[/WORKSET=<project-spec>]
```

Example `CWSD src`

- `<directory>`

  The CWSD command creates a project-directory *<directory>* in the database project structure relative to the working location for subsequent use in project operations

- `/WORKSET=<project-spec>` comprises:

  `<product-id>:<project-id>`

  This optionally specifies the project to be used for this command: failing this, the user's current project will be taken.

Constraints

Normally, only users with the appropriate management privileges can run this command.

This constraint can, however, be relaxed using the Set Project Permissions (SWSP) command, as described on page 342.

DAR – Delete Archive

NOTE This command is not available in the Issue Management–only licensed version of Dimensions.

```
<archive-id>
```

Example `DAR AA12AB`

- `<archive-id>`

  This is the identity of an archive which is to be deleted.

See the *Administrator's Guide* for details.

DBDB – Unregister an Existing Base Database Entry

```
/BDB_NAME=<base_db_name>
/DB_SERVICE=<base_db_instance>
/NN_NAME=<network_node_name>
```

Example

```
DBDB /BDB_NAME=SERENA-PCMS
   /DB_SERVICE=PCMSUDB
   /NN_NAME=MACHINE.COMPANY.COM
```

This command enables you to unregister base database entries in an installation's network administration tables. See the *Administrator's Guide* for details.

DBL – Delete Baseline

NOTE This command is not available in the Issue Management–only licensed version of Dimensions.

```
<baseline-spec>
```

Example

```
DBL PROD:"R M VERSION 2 FOR HP"
```

- `<baseline-spec>` comprises:

  `<product-id>:<baseline-id>`

Constraints

This command can be run only by the user who created the baseline or by a user with the appropriate management privileges.

A baseline used as a child collection cannot be deleted.

A baseline that has been actioned can be deleted only by a user with the appropriate management privileges.

A baseline cannot be deleted if it has been used to make a release or archive.

DBPROJ – Create a Dimensions Build Project

NOTE This command is no longer available. Refer to the *Dimensions Build User's and Administrator's Guide* for more information on how to define build projects.

```
Command no longer available.
```

DCH – Delete Request

```
<request-id>
```

Example `DCH PROD_HELD_349`

- `<request-id>`

  This is the identity of the held request to be deleted.

Constraints

Requests which have been saved in the other types of request lists cannot be deleted.

This command can be run only on requests that are pending (in the Held list). They can be deleted by the user who created them or by a user with the appropriate management privileges.

DCO – Delete an Existing Contact

```
/CO_NAME=<contact_name>
```

Example

```
DCO /CO_NAME="SERVER MANAGER"
DCO /CO_NAME="MAINFRAME MANAGER"
DCO /CO_NAME="DB ADMIN"
```

This command enables you to a delete an existing contact from an installation. See the *Administrator's Guide* for details.

DCST – Delete an Existing Codeset

```
/CDST_NUMBER=<codeset_number>
```

Example `DCST /CDST_NUMBER="2000"`

This command enables you to a delete an existing codeset from an installation. See the *Administrator's Guide* for details.

DDF – Define Data Formats

```
<format>
[/DESCRIPTION=<format-description>]
[/CLASS=<class-no>]
[/MIME_TYPE=<mime-type>]
[/COMPRESSION_LEVEL=<level>]
```

Example
```
DDF TXT /DESCRIPTION="Plain text"
   /CLASS=1 /MIME_TYPE="TEXT/PLAIN"
```

- `<format>`

  the format being defined.

- `/DESCRIPTION=<format-description>`

  descriptive name for the format.

- `/CLASS=<class-no>`

  Specifies the file types where:

  1=TEXT
  2=BINARY
  3=OpenVMS
  4=Macintosh
  5=NT

- `/MIME_TYPE=<mime-type>`

  Specifies the Multipurpose Internet Mail Extension (MIME) type. MIME types comprise seven broad categories, with each category also having subcategories defined by using a forward slash (/) separator. The broad categories are: Application, Audio, Image, Message, Multipart, Text and Video. An example of a subcategory is APPLICATION/WORD.

- `COMPRESSION_LEVEL=<level>`

  Specifies the compression level to be used when getting item revisions assigned this data format. Use a digit from 0 to 9, where 0 indicates no compression, 1 means fastest compression method (but less compression) and 9 indicates slowest compression method (but best compression). If this qualifier is omitted, text file formats will use fastest compression method (level 1) while all other file formats will use no compression.

- `No parameters or qualifiers`

  If DDF is executed without parameters or qualifiers, it prints a list of existing data formats together with their descriptions.

Description

This command enables a user to define data formats to be assigned to particular item types or request types:

- **For items**, a user with the role of TOOL-MANAGER can define a list of valid data formats for particular item types. These defined data formats can then, where appropriate, be subsequently assigned by a user with the role of TOOL-MANAGER to

particular item types using the Assign Data Formats to Item Types (ADF) command (see page 45).

- **For requests**, a user with the role of TOOL-MANAGER or CHANGE-MANAGER can define a valid data format for a particular request type. This defined data format can then, where appropriate, be subsequently assigned by a user with the role of PRODUCT-MANAGER or CHANGE-MANAGER to a particular request type using the Assign Data Formats to Request Types (ACF) command (see page 44).

This function is also available from the Administration Console, Data Formats and MIME Types option.

Uses for such a list of valid data formats include:

- Validation on item or request creation.
- Specifying file types: All items or requests that have formats not defined with "CLASS=1" (text) will be transferred to and from clients in binary mode. There is no need to assign the format to an item or request type.
- Specifying MIME types for the Dimensions web client. Dimensions web client will use the MIME type associated with the item's or request's format to display the item's or request's content within the user's browser.

If a list of valid data formats is subsequently assigned to an item type, then the use of one of those formats is compulsory when creating items of that type; whereas, if a list of format types has not been assigned, then any format can be used at the time of item creation, even one not defined by a user with the role of TOOL-MANAGER.

If a valid data format is subsequently assigned to a request type, the choice of this format is compulsory when creating requests of that type. If no format has been assigned, binary is assumed.

Existing defined data formats can be removed by the use of the Remove Data Format Definitions (RMDF) command (see page 301).

NOTE For items only, the data format for existing items can also be updated using the Update Item Attributes (UIA) command (see page 366).

See also "SDF – Set Data Format Flags" command on page 326.

Constraints

Only users with the appropriate management privileges on the product can run this command.

DEPLOY – Invoke a Deployment on the Serena Automated Deployment Server

NOTE This command is not available in the Issue Management–only licensed version of Dimensions.

```
/DEPLOY_MAP=<name>
/USER=<user_name>
/PASSWORD=<password>
/SERVER=<name>
/TASK=<text tag>
/PROJECT=<name>
/PORT=<number>
```

Example

```
DEPLOY /DEPLOY_MAP=64 /SERVER=deploy-server -
   /USER=admin /PASSWORD=admin -
   /PROJECT="test project1"-
   /TASK="first remote deploy" /PORT=80
```

This command invokes a deployment on the optional Serena automated deployment server. Refer to the associated user's guide that accompanies this optional software for details.

DFS – Delete an Existing File System

```
/FS_NAME=<file_system_name>
```

This command enables you to remove specific file systems definitions for each registered installation operating system. See the *Administrator's Guide* for details.

DGRP – Delete Group

```
<group-name>
```

Example `DGRP <group-name>`

where `<group-name>` is the name of the group to be deleted.

Description

The DGRP command deletes a group.

Constraints

Only users with the appropriate management privileges can run this command.

DI – Delete Item

NOTE This command is not available in the Issue Management–only licensed version of Dimensions.

```
<item-spec>
[/FILENAME=<file-name>]
[/WORKSET=<project-spec>]
```

Example

```
DI PROD:"QUERY RELEASE".AAAA-SRC;1
DI PROD:"QUERY RELEASE".AAAA-SRC;*
```

- `<item-spec>`

  Comprises:

  ```
  <product-id>:<item-id>.<variant>-<item-type>;<revision>
  ```

`item-id>`	may be omitted if `<file-name>` is specified.
`<variant>`	may be omitted if only one exists.
`<revision>`	may be specified as ***** to delete all revisions of the product item that are in the project used for this command. If omitted, the latest revision is deleted (see note in Introduction on page 16)

- `/FILENAME=<file-name>`

  Specifies the name of the file containing the item in the item-library.

  It may be omitted if **<item-id>** is specified.

- `/WORKSET=<project-spec>`

  Comprises:

  ```
  <product-id>:<project-id>
  ```

  This optionally specifies the project to be used for this command: failing this, the user's current project will be taken. All item revisions to be affected by the command must be within the project.

  Item revisions to be affected by the command may be specified explicitly, or they will be selected from the project.

Notes

When it deletes an item revision from a project, DI automatically deletes the corresponding file in all areas that previously contained this file and repopulates these areas with new active revisions (if any).

Constraints

This command can be run only by a user with the appropriate management privileges or, if the item revision is at the initial lifecycle state, by users if it is in their pending list.

Items cannot be deleted if they are included in a baseline.

DINS – Unregister an Existing Database Instance Entry

```
/DB_SERVICE=<base_db_instance>
/NN_NAME=<network_node_name>
```

Example

```
DINS /NN_NAME=MACHINE.COMPANY.COM -
   /NN_NAME=MACHINE.COMPANY.COM -
   /DB_SERVICE=PCMSUDB
```

This command enables you to unregister database instance entries in an installation's network administration tables. See the *Administrator's Guide* for details.

DIR – Define Item Relations

NOTE This command is not available in the Issue Management–only licensed version of Dimensions.

```
<relationship-id>
/DESCRIPTION=<description>
/SRC_TYPE=<item-type-name>
/DST_TYPE=<item-type-name>
[/[NO]INHERIT_FROM]
[/[NO]INHERIT_TO]
```

Example

```
DIR "INCLUDES"–
   /DESCRIPTION="C source/header relationship"–
   /SRC_TYPE="PROD_X:C" –
   /DST_TYPE="PROD_X:H"
```

- `<relationship_id>`

  Specifies the identifier of relationship to be created.

- `/DESCRIPTION=<description>`

  Specifies a description for the definition of the relationship type and is restricted to 240 characters.

- `/SRC_TYPE=<product-id><item-type>`

  Specifies the source product and item type from which the link will start.

- `/DST_TYPE=<product-id><item-type>`

  Specifies the destination product and item type at which the link will end or point.

- `/INHERIT_FROM`

  Specifies that a new item revision inherits the previous revision's children.

- `/INHERIT_TO`

  Specifies that a new item revision inherits the previous revision's parents.

Description

The command DIR is used to define a relationship between Dimensions item types. This relationship may be across products. Once a relationship has been defined it can be used by the RII and XII commands to create and delete relationships. (The RIR command is used to remove a relationship definition.)

Constraints

Only users with the appropriate management privileges can run this command.

DLGC – Delegate Request

```
<request-id>
[/SITE=<site_id> or /ROLE=<role> /USER_LIST=(<user1>,<user2>,…)]
[/CAPABILITY=<capability>]
[/ADD or /REPLACE or /DELETE]
[/[NO]DELEGATE_ITEMS]
```

Examples

```
DLGC PROD_DR_25 /ROLE=REVIEWER -
   /CAPABILITY=P /USER=SMITH

DLGC PAYROLL_TDR_1 /SITE="earth:intermediate@@dim9"
```

- `<request-id>`

  Identifies the request for which the delegation is to be made.

- `/SITE=<site-id>`

NOTE If `/SITE` is specified, then neither `/ROLE` nor `/USER_LIST` is applicable and will be rejected.

allows the request to be delegated to a replication site rather than a specific user, where `<site_id>` is either:

- `LOCAL`, a keyword which can be used to set the ownership to the local base database, or
- `<node_name>:<dbname>@@<dsn>`

NOTE @@ is used because @ is the default Dimensions escape character for the command line.

`<node_name>`	=	the node name
`<dbname>`	=	the base database
`<dsn>`	=	the database data source name

The DLGC command itself does not perform an immediate replication, rather it just determines who will be the next owner when the next scheduled replication occurs.

- `/ROLE=<role>`

NOTE If `/ROLE` is specified, then `/SITE` will be rejected.

Identifies the role title to be delegated.

- /USER_LIST=(<user1>,…)

NOTE If /USER_LIST is specified, then /SITE will be rejected.

Identifies by login user name one or more Dimensions users to whom delegation of the role is to be made for this request.

- /CAPABILITY=<capability>

  Specifies that this role delegation is to be **P** for Primary, **S** for Secondary (default) or **L** for Leader. (See AUR command for description of these role types.)

  Only one user and only /ADD or /REPLACE are valid for delegation of Primary capability.

NOTE /CAPABILITY is ignored when /SITE is specified.

- /ADD

  Specifies that the user(s) are to have this role for this request in addition to those who already have it.

  This is the default, if none of /ADD, /REPLACE, /DELETE is specified.

- /REPLACE

  Specifies that the user(s) are to have this role for this request instead of those who already have it.

- /DELETE

  Specifies that the user(s) are to be removed from the list of users who have this role for this request.

- /DELEGATE_ITEMS

  when delegating a request to a user, also automatically delegate those items related as Affected and InResponseTo to that user as well. This qualifier effectively invokes the DLGI command logic for all items related to that request using the qualifiers passed down to DLGC.

Constraints

This command can be run only by a user with the appropriate management privileges or by users for whom the request to be delegated is in their pending list.

DLGI – Delegate Item

NOTE This command is not available in the Issue Management–only licensed version of Dimensions.

```
<item-spec>
[/FILENAME=<file-name>]
/ROLE=<role>
/USER_LIST=(<user1>,<user2>,…)
[/WORKSET=<project-spec>]
[/CAPABILITY=<capability>]
[/ADD or /REPLACE or /DELETE]
```

Example

```
DLGI  PROD:"QUERY RELEASE"-SRC -
   /ROLE=REVIEWER /CAPABILITY=P -
   /USER=SMITH /FILENAME=qr.c
```

- `<item-spec>`

  Comprises:

  ```
  <product-id>:<item-id>.<variant>–<item-type>;<revision>
  ```

`<item-id>`	identifies the new item within the product.
`<variant>`	if omitted, the default (specified when the product was defined) is used.
`<revision>`	defaults to **1**, if omitted.

- `/FILENAME=<file-name>`

  Specifies the name of the project file name.

  The project file name identifies the relative path (directory plus file name) from the working location of the file to be used when the item is checked out from the current project. The project file name for the same item may *differ* between projects; for example, `src/hello.c`, `hello.c`, or `src/build/hello.c`.

  It may be omitted if `<item-id>` is specified.

- `/ROLE=<role>`

  Identifies the role title to be delegated.

- `/USER_LIST=(<user1>,…)`

  Identifies by login user name one or more Dimensions users to whom delegation of the role is to be made for this item.

- `/WORKSET=<project-spec>`

  Comprises:

  ```
  <product-id>:<project-id>
  ```

  Optionally specifies the project to be used for this command: failing this, the user's current project will be taken.

Item revisions to be affected by the command may be specified explicitly, or they will be selected from the project.

- /CAPABILITY=<capability>

  Specifies that this role delegation is to be **P** for Primary or **S** for Secondary (default) or **L** for Leader. (See AUR command for description of these role types.)

  Only one user and only /ADD or /REPLACE are valid for delegation of Primary capability.

- /ADD

  Specifies that the user(s) are to have this role for this item in addition to those who already have it.

  This is the default, if none of /ADD, /REPLACE, /DELETE is specified.

- /REPLACE

  Specifies that the user(s) are to have this role for this item instead of those who already have it.

- /DELETE

  Specifies that the user(s) are to be removed from the list of users who have this role for this item.

Constraints

This command can be run only by a user with the appropriate management privileges or by users for whom the item to be delegated is in their pending list.

DMNR – Delete Mail Notification Rule

```
<rule-id>
```

Example `DMNR <rule-id>`

where `<rule-id>` is the name of the rule to be deleted.

Description

The DMNR command deletes a mail notification rule.

Constraints

Only users with the appropriate management privileges can run this command.

DNC – Delete an Existing Network Node Connection

```
/SERVER_NAME=<server_node_name>
/CLIENT_NAME=<client_node_name>
/NWO_NAME=<network_object_name>
```

Example

```
DNC /CLIENT_NAME=TEST_MACHINE.COMPANY.COM -
   /SERVER_NAME=MACHINE.COMPANY.COM -
   /NWO_NAME=PCMS_SDP
```

This command enables you to delete an existing network node connection from an installation. See the *Administrator's Guide* for details.

DNDO – Delete an Existing Network Node Object

```
/NN_NAME=<network_node_name>
/NWO_NAME=<network_object_name>
```

This command enables you to delete an existing network node from an installation. See the *Administrator's Guide* for details.

DNN – Delete an Existing Network Node

```
/NN_NAME=<network_node_name>
```

Example `DNN /NN_NAME=MACHINE.COMPANY.COM`

This command enables you to delete an existing installation network node. See the *Administrator's Guide* for details.

DNP – Define New Product

```
<product-id>
[/BASED_ID=<based-on-product>]
[/STRUCTURE=ALL]
[/STRUCTURE=NO]
/DESCRIPTION=<description>
/PRODUCT_MANAGER=<product-manager>
[/PARTS_CONTROLLER=<parts-controller>]
[/CHANGE_MANAGER=<change-manager>]
/VARIANT=<variant>
/PCS=<pcs>
[/ATTRIBUTES=(<attribute_id>=<value>,…)]
```

Example

```
DNP PRODTEST20 /VARIANT=AAAA /PCS=1 -
   /DESC="PROD Rel 2.0 Test Vehicle" -
   /PRODUCT_MANAGER=SMITH
   /ATRIBUTES=(site=dallas, priority=critical,country_orig=germany)
```

- `<product-id>`

  Specifies the identity of the new product.

NOTE Only alpha-numeric and "underscore" (_) characters can be used to specify the product-id.

- `/BASED_ID=<based-on-product>`

  Specifies the `<product-id>` of an existing product on which the new one is to be based.

NOTE If the existing product has upload rules defined for it, then those upload rules will be copied to the new product being defined.

  If omitted, the new product is based on the `$GENERIC` product.

  `/STRUCTURE=ALL`

  Copy structure from the `<based-on-product>` to the new product. The default is `/STRUCTURE=NO`

  `/STRUCTURE=NO`

  Do **not** copy structure from the `<based-on-product>` to the new product. This is the default.

- `/DESCRIPTION=<description>`

  Specifies a description of the product.

- `/PRODUCT_MANAGER=<product-manager>`

  assigns the role of `PRODUCT-MANAGER` to the specified user. (By default a user with the role `PRODUCT-MANAGER` is also automatically assigned the role of `PCMS-ROLE-MANAGER` and `PCMS-PART-MANAGER` for the associated project – other users can also be assigned the `WORKSET-MANAGER` role via the DUR command.)

- /PARTS_CONTROLLER=<parts-controller>

  Assigns the role of PARTS-CONTROLLER to the specified user.

  It defaults to <product-manager>, if omitted.

- /CHANGE_MANAGER=<change-manager>

  assigns the role of CHANGE-MANAGER to the specified user.

  It defaults to <product-manager>, if omitted.

- /VARIANT=<variant>

  Specifies the default <variant> which is to be used by Dimensions for this product, whenever new design parts and items are created.

- /PCS=<pcs>

  Specifies the PCS which is to be used by Dimensions for this product, whenever new design part variants are created.

- /ATTRIBUTES=(<attribute_id>=<value>,…)

  Specifies a value for a new mandatory product level attribute. If you do not specify a value, an error message is issued.

<attribute_id>	is the name of the new attribute.
<value>	is the attribute value.

Constraints

Only users with the appropriate management privileges can run this command.

DNWO – Delete an Existing Network Object

```
/NWO_NAME=<network_object_name>
```

Example `dnwo /nwo_name=pcms_sdp`

This command enables you to delete an existing network object from an installation. See the *Administrator's Guide* for details.

DOS – Delete an Existing Operating System

```
/OS_NAME=<operating_system_name>
```

This command enables you to delete an existing operating system from an installation. See the *Administrator's Guide* for details.

DOWNLOAD – Download Project or Baseline

NOTE This command is not available in the Issue Management–only licensed version of Dimensions.

```
[<file-spec> or /DIRECTORY=<directory-spec> or
    /USER_FILELIST=<filelist-file> or /USER_ITEMLIST=<itemlist-file>]
[/[NO]RECURSIVE]
[/[NO]EXPAND]
[/[NO]TOUCH]
[/[NO]OVERWRITE]
[/LOGFILE=<file-spec> or /SCRIPTFILE=<file-spec>]
[/CODEPAGE=<code-page> or DEFAULT]
[/WORKSET=<project-spec> or /BASELINE=<baseline-spec>]
[/USER_DIRECTORY=<directory-path>]
[/[NO]CONFLICT_CHECK]
[/FILTER=<filter-name>]
[/TRANSFER_SCRIPTS=<script-set>]
[/NOMETADATA]
```

Examples

- `DOWNLOAD`

  Downloads the tip of the current project into the working location.

- `DOWNLOAD /DIRECTORY="build"`

  Downloads the tip of "build" into the working location.

- `DOWNLOAD /DIRECTORY=java_build /BASELINE="pvcs:dm10_build_5_11" /USER_DIRECTORY="C:\projects\dm10"`

  Downloads the `java_build` directory from a baseline into the specified directory.

- `DOWNLOAD "C:\temp\build\build.mk" /TOUCH /BASELINE="PVCS:DM10 TIER1 FINAL"`

  Assuming that the project user directory is set to `C:\temp`, this command will update (if necessary) the `C:\temp\build\build.mk` file with the baseline item revision with file name `build\build.mk` from the PVCS:DM10 TIER1 FINAL baseline into the `C:\temp` directory. The modification time of the updated file will be set to the current system time.

- `DOWNLOAD /DIRECTORY="build\include" /TOUCH /WORKSET="PVCS:DM10"`

  All files found in the project directory `build\include` and in any directories below it will be considered for download into the current working location. If the user has locally modified any matching files in the current working location, these files will not be updated.

Parameters

- `<file-spec>`

  This parameter specifies the name of the file to be downloaded. (The Dimensions `node::` syntax is also valid.)

- <directory-spec>

  This parameter specifies a directory path. It is matched to the corresponding project or baseline directory, the files in that project or baseline directory are enumerated, and each one that has not been been modified is downloaded.

  You can specify the directory using the Dimensions node:: syntax. It can also be a path relative to the user working location.

- /USER_FILELIST=<filelist-file>

  This optional qualifier must specify a file containing a list of file names to be downloaded from the project or baseline. Each file name must be on a separate line.

  File names may be specified as either relative or absolute paths. If the path is absolute, it is interpreted as a full project or baseline file path; otherwise, Dimensions obtains the project or baseline path by mapping the file name to the operation root directory, which is the directory specified by the /USER_DIRECTORY qualifier (if any) or the current working location as specified by the last SCWS command. If such a mapping is not possible, the file name is ignored.

- /USER_ITEMLIST=<itemlist-file>

  This optional qualifier must specify a file containing a list of item specifications to be downloaded from the project or baseline. This qualifier allows you to efficiently download an arbitrary set of items from Dimensions using the complete item specifications. Each item specification must be on a separate line. There is no need to use double quotes with item specifications.

- /NORECURSIVE

  If /DIRECTORY is specified and this qualifier is not present, all files that have not been modified in all directories beneath the one specified are downloaded. /NORECURSIVE specifies that only files at the specified directory level are downloaded.

- /EXPAND

  Specifies substitution variable expansion.

- /TOUCH

  Sets the modification time of the user file to the current system time.

- /OVERWRITE

  By default, DOWNLOAD does not overwrite files in the operation root directory that have no metadata, are locally modified, are checked out to the operation root directory, or correspond to an item different from the one being fetched (that is, files that have different <product>:<item-id>.<variant>-<type> pairs).

  If /OVERWRITE is specified, DOWNLOAD overwrites such files with the content of corresponding project or baseline item revisions.

- /LOGFILE=<file-spec>

  This qualifier specifies that a log file be generated at the given file location containing the results of all the individual Dimensions operations executed through this command.

- /SCRIPTFILE=<file-spec>

  This qualifier specifies that a script file be generated at the given file location containing the individual Dimensions operations that would otherwise be executed through this command. The operations are not executed.

- <code-page>

  The code page to be associated with the item.

- <project-spec>

  The project from which to download the files. If this parameter is not specified, files are downloaded from the current session project.

- <baseline-spec>

  The baseline from which to download the files. If this parameter is not specified, files are downloaded from the current session baseline.

- <directory-path>

  Use /USER_DIRECTORY=<directory-path> to specify a download directory other than the current working location. For example, the following command downloads the project into C:\temp regardless of what the current working location is:

  ```
  DOWNLOAD /USER_DIRECTORY="C:\temp"
  ```

  The following command downloads the project into the /tmp directory on host "hostname":

  ```
  DOWNLOAD /USER_DIRECTORY="hostname::/tmp"
  ```

  The following command downloads the project into the src directory inside the area_name area:

  ```
  DOWNLOAD /USER_DIRECTORY="area_name::src"
  ```

- /CONFLICT_CHECK

  Specifies that DOWNLOAD will search for unresolved merge conflicts for each item revision to be fetched. If any unresolved conflicts are found, Dimensions issues a warning and ignores the corresponding revision. By default, DOWNLOAD does not perform this check.

- /FILTER=<filter-name>

  Specifies that DOWNLOAD will get only files that satisfy the criteria specified in the <filter-name> area filter.

- /TRANSFER_SCRIPTS=<script-set>

  Specifies the transfer script set to be executed as items are fetched during the command's execution. The script set contains a comma-separated list of the names of pre/post/fail transfer scripts in the following format:

  (<pre-script>,<post-script>,<fail-script>)

  If one of the scripts is undefined, use $NONE as a placeholder.

- /NOMETADATA

  This parameter disables creation and usage of metadata files in the local work area.

Description

The DOWNLOAD command downloads repository content (a project or a baseline) to the developer's workspace.

This command compares each item revision selected by the passed parameters with the corresponding on-disk files. If the disk file has been locally modified (by the use of optimistic locking), or does not have Dimensions metadata, or has been locally checked out, then the command issues a warning and skips the file. Otherwise, DOWNLOAD overwrites the disk file with the corresponding item revision.

The DOWNLOAD command represents the most efficient way of getting multiple item revisions from the Dimensions repository. It is particularly optimized for transferring many relatively small files across high-latency wide area network (WAN) connections, where it could be several times faster than an equivalent script of separate FI commands.

DPB – Deploy Baseline

NOTE This command is not available in the Issue Management–only licensed version of Dimensions.

```
<baseline-spec>
[/WORKSET=<project-spec>]
[/STAGE=<new-stage>]
[/COMMENT=<comment>]
```

Example

```
DPB "PVCS:DM10 COMMON TOOLS" /STAGE=APPROVED
```

Parameters

- `<baseline-spec>`

  Baseline specification.

- `<project-spec>`

  Specifies the root project to be used for this command. If the /WORKSET qualifier is omitted, the DPB command uses the current project.

- `<new-stage>`

  Specifies the target stage. This must be a stage from the global stage lifecycle, and there must be a transition from the current stage to the new stage in the stage lifecycle in order for the DPB command to succeed.

- `<comment>`

  Textual comment.

Description

The DPB command deploys the specified baseline to the specified stage in the context of the specified project. If the project has deployment areas assigned to it, these areas are updated as the baseline is deployed from one stage to another.

If a deployment area has an area filter, the deployment area is updated only with item revisions that match the area filter's rules. If a deployment area has transer scripts associated with it, these scripts are executed before and after all item revisions are transferred to and / or removed from the deployment area. When transfer scripts are expanded for execution in an area, a number of predefined template variables are set by Dimensions. In particular, when items are deployed as part of the DPB command, the `DMBLNPRODUCT` and `DMBLNID` template variables will contain the product and ID of the baseline that contains the items.

See "DPI – Deploy Item" on page 151 for details on the template variables.

The project in which the baseline is deployed must either be a standalone project or a root project with sub-projects (in other words, the project in which the baseline is to be deployed may not be a sub-project attached to another project). In order for the baseline to be deployable in this project, each item revision included in the baseline must also be present in the project (or the namespace of the root project in case the specified project is a root project with sub-projects). The project must follow the manual deployment model.

Constraints

Only users with the "Deploy Baseline to Next Stage" or "Deploy Baseline to Any Stage" privileges can run this command.

DPI – Deploy Item

NOTE This command is not available in the Issue Management–only licensed version of Dimensions.

```
<item-spec>
[/FILENAME=<file-name>]
[/WORKSET=<project-spec>]
[/STAGE=<new-stage>]
[/COMMENT=<comment>]
```

Example `DPI PROD:"QUERY RELEASE".AAAA;2 /FILENAME=query.c /STAGE=APPROVED`

Parameters

- `<item-spec>`

  Item specification. Comprises:

  `<product-id>:<item-id>.<variant>– <item-type>;<revision>`

`<item-id>`	may be omitted if `<file-name>` is specified.
`<variant>`	may be omitted if only one exists.
`<revision>`	may be omitted if `<file-name>` is specified.

- `<file-name>`

  Specifies the name of the project file name.

  The project file name identifies the relative path (directory plus file name) from the working location to the item to be used from the current project. The project file name for the same item may *differ* between projects; for example, `src/hello.c`, `hello.c`, or `src/build/hello.c`.

  It may be omitted if `<item-id>` is specified within the `<item-spec>` argument.

- `<project-spec>`

  Specifies the project to be used for this command. If the /WORKSET qualifier is omitted, the DPI command uses the current project.

- `<new-stage>`

  Specifies the target stage. This must be a stage from the global stage lifecycle, and there must be a transition from the current stage to the new stage in the stage lifecycle in order for the DPI command to succeed.

- `<comment>`

  Textual comment.

Description

The DPI command deploys the specified item revision to the specified stage and updates deployment areas accordingly. If a deployment area has an area filter, the deployment area is updated only if the item revision matches the area filter's rules. If a deployment area has transfer scripts associated with it, these scripts are executed before and after the item revision is transferred to or removed from the deployment area.

When transfer scripts are expanded for execution in an area, a number of predefined template variables are set by Dimensions. The following is a list of such variables:

```
DMSERVER =<Dimensions server hostname>
DMBLNPRODUCT = <product of the baseline being deployed, if any>
DMBLNID = <ID of the baseline being deployed, if any>
DMREQUEST = <the id of the request being deployed, if any>
DMAREA = <area-id>
DMDIR = <full path of directory the file is being transferred to or
    removed from>
DMFILENAME = <file-name>
DMCOMMENT = <the /COMMENT qualifier value from the current operation>
DMTRANSFERTYPE = <operation type>, "c" - for copy into area, "r" - for
    "removal from area
DMREVISION = <revision string of the deployed item revision>
DMBRANCH = <branch name portion of the DMREVISION variable>
DMFORMAT = <data format of the deployed item revision>
DMPREFIX = <file name less extension of the deployed item revision>
DMSUFFIX = <file extension of the deployed item revision>
DMWSPRODUCT = <the product of the current project>
DMWSID = <the id of the current project>
DMPRODUCT = <the product of the deployed item revision>
DMID = <the item-id of the deployed item revision>
DMVARIANT = <the variant of the deployed item revision>
DMTYPE = <the item type of the deployed item revision>
DMCTIME = <the date and time of the transfer>
```

For example, say that the following command is executed:

```
DPI "DEPLOYMENT:COMMONS LOGGING API JAR.A-DAT;main#1" /STAGE="UNIT TEST"
    /COMMENT="Item deployed from DEVELOPMENT to UNIT TEST" /
    WORKSET=DEPLOYMENT:DMNET
```

and say that the Dimensions administrator had assigned the DMNET-UT deployment area to the DEPLOYMENT:DMNET project and that the area is located in the `/tmp/dpr_test/areas/ut` directory. The above template variables will have the following values:

```
DMSERVER = stal-dev-lx1
DMAREA = DMNET-UT
DMDIR =
DMCOMMENT = Item deployed from DEVELOPMENT to UNIT TEST
DMFILENAME = commons-logging-api.jar
DMTRANSFERTYPE = c
DMREVISION = main#1
DMBRANCH = main
DMFORMAT = ZIP
DMPREFIX = commons-logging-api
DMSUFFIX = jar
DMWSPRODUCT = DEPLOYMENT
DMWSID = DMNET
DMPRODUCT = DEPLOYMENT
DMID = COMMONS LOGGING API JAR
DMVARIANT = A
DMTYPE = DAT
DMCTIME = Tue May 30 15:35:03 2006
```

```
DMBLNPRODUCT = <empty string>
DMBLNID = <empty string>
DMREQUEST = <empty string>
```

In this example, the DMBLNPRODUCT, DMBLNID, and DMREQUEST variables contain empty strings. This occurs because neither a request nor a baseline is deployed.

For detailed information about writing deployment area scripts, and an overview of the associated templating language, see the *Developer's Reference*.

Constraints

Only users with the "Deploy Item to Next Stage" or "Deploy Item to Any Stage" privileges can run this command.

DPL – Define Product Libraries

NOTE This command is not available in the Issue Management–only licensed version of Dimensions.

```
<product-id>
/ITEM_TYPE=<item-type>
[/DELTA]
[/LIBRARY=<directory-name>]
[/NETWORK_NODE=<network-node-name>]
[/PROTECTION=<protection>]
[/ADD or /UPDATE or /DELETE]
```

NOTE For information about managing Dimensions item libraries on a z/OS mainframe for access through Dimensions for z/OS, see the *Dimensions for z/OS User's and Administrator's Guide*.

Example

```
DPL PROD /ITEM_TYPE=DATA /NET=eldarmar -
/LIB="/usr/PROD/library/item/data/"
```

- `<product-id>`

  Identifies the product in which an item library is (to be) defined.

- `/ITEM_TYPE=<item-type>`

  Specifies that the library is for items of this type.

  `/ITEM_TYPE=*` may be used to specify a default item library.

- `/DELTA`

  Indicates that the item-library is (to be) a delta library.

  If omitted, the item-library is (to be) a normal library.

NOTE

- If `/UPDATE` is specified, `/DELTA` must be specified if and only if the existing item library is a delta library. It cannot be specified or omitted to change a normal library to a delta library or vice versa.
- The compress storage option is not available for delta library storage (see the *Process Modeling User's Guide*).

- `/LIBRARY=<directory-name>`

  names the directory to hold the specified library as follows:

UNIX	the absolute directory path, ending with the character / e.g. `/usr/PROD/lib/`
Windows	the absolute directory path, ending with the character \ e.g. `c:\PROD\lib\`

The library <directory name> is **not** required when a library definition is being deleted.

- /NETWORK_NODE=<network-node-name>

  Identifies to which computer and file system in a network <directory-name> is applicable.

  The <Network_Node> must be stated even if a local machine is being used.

- /PROTECTION=<protection>

  Specifies the protection code for the items library directory in the standard operating system format in UNIX.

  If omitted, the defaults used are:

  UNIX: rwx,rx,r

  For Windows this qualifier **must** be omitted. Once a library directory has been created (and before it is used), the owner should specify its protection by creating an ACL (Access Control List) for it. See separate sub-section below for additional information.

- /ADD or /UPDATE or /DELETE

  Indicates whether this library definition is being added, altered or removed.

  The default is /ADD.

Protecting the Item Library Files from Unauthorized Changes on Windows

The Dimensions item libraries are protected from unauthorized changes by setting an ACL on each directory which is defined to hold an item library. This must be done manually (i.e. as a supplementary operation external to Dimensions processing), using the Windows Explorer. Because ACLs are allowed only on files on a disk with an NTFS file system, it is recommended that item libraries are not defined on disks with FAT file systems, as there would be no way to protect the item libraries from unauthorized changes.

An ACL with the following attributes is recommended:

- System: Full Control
- Administrators: Read Access
- Owner: Read Access

This will ensure that only the Dimensions Listener Service is able to write files into these directories.

Some additional users could be granted Read access to the Item files by adding a group or users (using the Windows Explorer Security | Permissions menu option).

Permissions Only the 'System' user is permitted to Write, Change and Delete ACLs.

Constraints

You cannot define operating system directories for request libraries. Requests are automatically stored by Dimensions in the RDBMS database.

This command can be run only by a user with the appropriate management privileges but **not** while other Dimensions users are using the libraries. **Such operations could cause fatal library access errors**.

Dimensions does not maneuver the contents of the library to correspond with any revisions made—it issues a warning message advising the user with the role of PRODUCT-MANAGER to perform this task.

DPR – Deploy Request

NOTE This command is not available in the Issue Management–only licensed version of Dimensions.

```
<request-id>
[/WORKSET=<project-spec>]
[/STAGE=<new-stage>]
[/COMMENT=<comment>]
[/[NO]CANCEL_TRAVERSE]
```

Example `DPR PVCS_EC_100 /STAGE=APPROVED`

Deploys all revisions related to PVCS_EC_100 as IRT to the APPROVED stage.

Parameters

- `<request-id>`

  Request identifier.

- `<project-spec>`

  Specifies the project to be used for this command. If the /WORKSET qualifier is omitted, the DPR command uses the current project.

- `<new-stage>`

  Specifies the target stage. This must be a stage from the global stage lifecycle, and there must be a transition from the current stage to the new stage in the stage lifecycle in order for the DPR command to succeed.

- `<comment>`

  Textual comment.

- `CANCEL_TRAVERSE or NOCANCEL_TRAVERSE`

  Specifies whether the hierarchy of child requests is traversed.

  The DPR command traverses the child requests by default. CANCEL_TRAVERSE reverses the default behavior.

Description

This command finds all items related as IRT ("in response to") to the specified request and all of its child requests and deploys these items to the specified stage in the context of the current project. If the current project includes deployment areas, these areas are updated as the request is deployed from one stage to another.

In order to be deployable, a request must be related to a project. The request is then deployable only in the context of that project. When the hierarchy of child requests is traversed, only such child requests that are related to the same project as the main request or that are related to a project in the same project tree as the main request's project will be considered for deployment. If a request has any items checked out against it, it cannot be deployed.

If a deployment area has an area filter, the deployment area is updated only with item revisions that match the filter's rules. If a deployment area has transfer scripts associated with it, these scripts are executed before and after all item revisions are transferred to

and / or removed from the area. When transfer scripts are expanded for execution in the area, a number of predefined template variables are set by Dimensions. In particular, when items are deployed as part of the DPR command, the DMREQUEST template variable will contain the request contributing the item being deployed.

See "DPI – Deploy Item" on page 151 for details on the template variables.

Constraints

Only users with the "Deploy Request to Next Stage" and "Deploy Request to Any Stage" privileges can run this command.

DPROJ – Define a Dimensions Project

NOTE This command is no longer available. Use DWS to define a new project and Dimensions Build to define a build project.

See the DWS command and the *Dimensions Build User's and Administrator's Guide*.

DPRP – Define Preservation Rules Policy

NOTE This command is not available in the Issue Management–only licensed version of Dimensions.

```
<policy-spec>
[/DESCRIPTION=<description>]
[/DEFAULT_RULE={NORMAL|EXTERNAL|PLACEHOLDER}]
[/ITEM_TYPE={SOURCE|LISTING|etc} /RULE={NORMAL|EXTERNAL|PLACEHOLDER}]]
[/ADD | /DELETE | /UPDATE]
```

IMPORTANT! Depending on whether or not the qualifier /ITEM_TYPE is specified, the /ADD, /DELETE, and /UPDATE qualifiers behave differently. See the qualifier descriptions later in this section for details.

Examples

1 The following commands:

```
DPRP PAYROLL:DEFAULT_POLICY -
/DESCRIPTION="Default site policy" -
/DEFAULT_RULE=NORMAL

DPRP PAYROLL:DEFAULT_POLICY -
/ITEM_TYPE=LISTING /RULE=EXTERNAL

DPRP PAYROLL:DEFAULT_POLICY -
/ITEM_TYPE=OBJ /RULE=PLACEHOLDER /ADD
```

define a new preservation policy that conjunctionally specifies:

- That the default rule for all item types is to be preserved as normal item revisions in an item library, and
- that item revisions of type LISTING are to be preserved as external items, and
- that item revisions of type OBJ are to be preserved as placeholder items.

2 The following command deletes a rule for item type PAYROLL:OBJ

```
DPRP PAYROLL:DEFAULT_POLICY -
/ITEM_TYPE=OBJ /DELETE
```

3 The following command updates a rule for item type PAYROLL:LISTING

```
DPRP PAYROLL:DEFAULT_POLICY -
/ITEM_TYPE=LISTING /RULE=NORMAL /UPDATE
```

4 The following command updates the description of a preservation policy:

```
DPRP PAYROLL:DEFAULT_POLICY -
/DESCRIPTION="default site preservation policy" -
/DEFAULT_RULE=PLACEHOLDER /UPDATE
```

5 The following command deletes the preservation policy:

```
DPRP PAYROLL:DEFAULT_POLICY /DELETE
```

- `<product-spec>`

  comprises `<product-id>:<policy-id>` where:

  - `<product-id>`

    Specifies the product for which the policy is defined.

  - `<policy-id>`

    Specifies the policy identifier.

- `/DESCRIPTION=<description>`

  Specifies the description.

- `/DEFAULT_RULE={NORMAL|EXTERNAL|PLACEHOLDER}`

  Specifies the default preservation rule for this policy; namely:

  - `/DEFAULT_RULE=NORMAL` for specifying normal items (this is the default and may be omitted), or
  - `/DEFAULT_RULE=EXTERNAL` for specifying external items, or
  - `/DEFAULT_RULE=PLACEHOLDER` for specifying placeholder items.

NOTE Only build targets generated outside a build area can be preserved as external items.

- `/ITEM_TYPE={SOURCE|LISTING|etc}`

  Specifies the item type name (within the product in which the policy is defined) for which a preservation rule is defined. For example, SOURCE specifies the `<product-id>:SOURCE` item type.

- `/RULE-{NORMAL|EXTERNAL|PLACEHOLDER}`

  Specifies whether built targets of the above type are to be preserved in a Dimensions item library, namely:

  - `/RULE=NORMAL` for specifying normal items (this is the default and may be omitted), or
  - `/RULE=EXTERNAL` for specifying external items, or
  - `/RULE=PLACEHOLDER` or specifying placeholder items.

NOTE Only build targets generated outside a build area can be preserved as external.

- `/ADD`

  if `/ITEM_TYPE` is not specified, specifies whether to add a new preservation policy. This is the default, and may be omitted. Otherwise, if `/ITEM_TYPE` is specified, this qualifier specifies whether to add a preservation rule for the specified item type.

- /DELETE

  if /ITEM_TYPE is not specified, specifies whether to delete the preservation policy and all associated rules. Otherwise, if /ITEM_TYPE is specified, this qualifier specifies whether to delete a preservation rule for the specified item type.

- /UPDATE

  if /ITEM_TYPE is not specified, specifies whether to update the description of the preservation policy. Otherwise, if /ITEM_TYPE is specified, this qualifier specifies whether to update a preservation rule for the specified item type.

Description

The DPRP command defines a preservation rules policy within a Dimensions product. The policy has an identifier, a description, and a default rule; optionally, it can contain a list of additional preservation rules (exceptions to the default rule).

A preservation rule is conceptually similar to an upload rule, and defines how built targets are preserved in the Dimensions product. The built targets, which are mapped by upload rules to a particular item type, can be preserved as:

- normal item revisions stored in an item library, or
- "external" item revisions stored externally, outside of the control of the Dimensions product, or
- "placeholder" item revisions stored as zero-byte assets in the Dimensions item library.

Each built target is mapped by upload rules to a particular Dimensions item type. The rules in a preservation policy define whether each built target of this item type is to be preserved as a normal item revision or either as a "placeholder" or "external" item revision. The difference between "placeholder" and "external" item revisions is:

- External item revisions

  External item revisions represent versioned files whose actual content is stored outside of a Dimensions item library. In other words, an external item revision has external storage. The actual location of an external revision, which is comprised of a network node name and a full path to a file on that network node, is stored in the Dimensions base database. Typically, one would use external item revisions to represent compile and link listings generated as a result of a build on z/OS.

  Physically, an external item revision is represented as a zero-byte file in the item library. An external item revision can only be created for a build output (such as listing) that was generated outside a build area, and each external item revision must represent a unique file. External item revisions can be actioned and promoted from one stage to another; however, the file referred to by the external file revision is not promoted between build areas occurs – promotion is reduced to a status change. AUDIT will ignore external item revisions.

  External item revisions can be gotten (fetched) and checked out (extracted). If you use a Dimensions client to revise an external item revision (by executing RI or UI commands), then a normal item revision will be created. External item revisions are only creatable by build outputs collection.

- External item revisions

  Placeholder item revisions represent versioned files with no content in the Dimensions item library. In other words, a placeholder item revision has no storage at all. Typically, one would use placeholder item revisions to represent intermediate targets and made-of relationships, and thus avoiding the overhead of preserving in Dimensions every build output generated as a result of a build job.

  Physically, a placeholder item revision is represented as a zero-byte file in the item library, and can be created for any build output regardless of its location – inside or outside a build area. Placeholder item revisions can be actioned and promoted from one stage to another; however, since a placeholder item revision does not refer to any files at all, no file promotion between build areas occurs – promotion is reduced to a status change. AUDIT will ignore placeholder item revisions.

  Placeholder item revisions can neither be gotten (fetched) nor checked out (extracted). If you use a Dimensions client to revise a placeholder item revision (by executing RI or UI commands), then a normal item revision will be created. Placeholder item revisions are only creatable by build outputs collection.

A build administrator will be able to assign a preservation rule to a build stage within a project. By default, if no preservation rules are assigned to a build stage within a project, then the Dimensions product will default to capturing all built targets as normal Dimensions items.

Constraints

Only users with the appropriate management privileges can run this command.

DPV – Delete Design Part Variant

```
<part-spec>
```

Example `DPV PROD:"RELEASE MANAGEMENT".IBM`

- `<part-spec>` comprises:

  `<product-id>:<part-id>.<variant>;<pcs>`

  `<variant>` may be omitted if only one exists.

  `<pcs>` is ignored. All PCSs of the specified variant are deleted.

Constraints

Only users with the appropriate management privileges can run this command.

The design part must meet the following criteria:

- it is not related to any request (regardless of the status of the request)
- it has no child design parts
- it owns no items
- it is not in any baseline
- it is not the 'top' design part

DREL – Delete Release

NOTE This command is not available in the Issue Management–only licensed version of Dimensions.

```
<release-spec>
```

Example `DREL PROD:"R M 2.0 FOR HP"`

- `<release-spec>` comprises:

  `<product-id>:<release-id>`

Constraints

This command can be run only by a user with the appropriate management privileges on the product that owns the release or the owner of the baseline on which the release was based.

DRSD – Delete an Existing Resident Software Definition

NOTE This command is not available in the Issue Management–only licensed version of Dimensions.

```
/RSD_NAME=<name_RSD>
```

This command enables you to unregister an installation Resident Software Definition (RSD). See t he *Administrator's Guide* for details.

DUR – Define User Roles

```
/ROLE=<role>
[/DESCRIPTION=<description>]
[/ADD or /DELETE]
```

Example

```
DUR /ROLE=DEVELOPER -
/DESC="Creates and edits items"
```

- `/ROLE=<role>`

  Specifies a role title for use in the lifecycles used in the base database for all products.

- `/DESCRIPTION=<description>`

  This is optional text to explain the purpose of this new role title.

- `/ADD`

  Adds the role definition.

- `/DELETE`

  Indicates that the specified role title is an obsolete one no longer required in this product.

  If omitted, the command identifies a new role-title to be used i.e. `/ADD` is the default.

Constraints

Only users with the appropriate management privileges can run this command.

The role of `PCMS-ROLE-MANAGER` is subordinate to the role of `PRODUCT-MANAGER` and cannot therefore be used to define that superior role.

DUSR – Unregister User

```
<user-id>
[\[NO]KEEP]
```

Description

This command is the same as XREG.

For details, see the *Serena Dimensions CM Administrator's Guide*.

DVB – Define Version Branch

NOTE This command is not available in the Issue Management–only licensed version of Dimensions.

```
<branch-id>
/DESCRIPTION=<description>
[/[NO]LOCK]
```

Example

```
DVB MAINT -
/DESCRIPTION="Principal branch for maintenance work"
```

- `<branch-id>`

  unique branch identifier.

NOTE Only alpha-numeric and "underscore" (_) characters can be used to specify the branch-id.

- `/DESCRIPTION=<description>`

  brief description of the purpose for the branch.

- `/LOCK`

  Optional flag to specify that the branch is locked and further development along it cannot take place.

- `/NOLOCK`

  Optional flag, negation of LOCK and is the default.

- `No parameters or qualifiers`

  if DVB is executed without parameters or qualifiers, it prints a list of defined branches together with their description and lock status.

Description

The DVB command is used to define a new branch-id within a Dimensions database for use with version commands. This function is available only in command mode.

Once branch-ids are defined, a valid list (subset) of them is assigned to a particular **existing** project by use of the Set Project Attributes (SWS) command. Alternatively, a valid list of branch-ids can be associated to a *new* project at the time the project is defined using the Define New Project (DWS) command.

The description and/or lock status of an existing `branch-id` definition can be modified (set) using the Set Version Branch Flags (SVBF) command.

A `branch-id` definition can be removed by the Remove Version Branch (RMVB) command.

Constraints

Only users with the appropriate management privileges can run this command.

DWP – Delete Whole Product

```
<product-id>
```

Example `DWP PRODTEST20`

The DWP command deletes all the information about a product (items, requests, design parts, any existing product-specific upload rules, etc) from the Dimensions database.

CAUTION! This command must be used with great care as there is no undo operation for it.

NOTE

Although the product is deleted from the database, Dimensions will *not* remove the contents of the item libraries. (However, because requests are stored in the database they, and any associated attachments, *will* be deleted.)

The contents of the item libraries will still be available, and you will be able to back them up or move them to other directories. It is your responsibility to delete their contents.

Any references within a project to a product's items will be automatically removed from that project when the product is deleted.

Constraints

Only users with the appropriate management privileges can run this command.

DWS – Define New Project

NOTE This command is not available in the Issue Management–only licensed version of Dimensions.

```
<project-spec>
/DESCRIPTION=<description>
[/WORKSET=<project-spec>]
[/BASELINE=<baseline-spec>]
[/ATTRIBUTES=(<attr1>,attr2,…)]
[/TYPE=<type-name>]
[/STATUS=<status>]
[/BRANCH|/TRUNK]
[/COPY_ON_DEPLOY]
[/[NO]AUTO_REV]
[/VALID_BRANCHES=(<branch-id1>,<branch-id2>,…)]
[/DEFAULT_BRANCH=<branch-id>]
[/DEPLOYMENT_MODEL=MANUAL or /DEPLOYMENT_MODEL=AUTOMATIC]
[/KEEP_STAGE]
```

Example

```
DWS "PROD_X:TEST_WS" -
/DESCRIPTION="Project for portation work" –
/BASELINE="PROD_X:prod 2.3 beta"
```

- `<project-spec>`

  Comprises:

  `<product-id>:<project-id>`

- `/DESCRIPTION=<description>`

  Specifies the description to be attached to the project definition.

- `/WORKSET=<project-spec>`

  Optionally specifies the project on which to base the new project.

- `/BASELINE=<baseline-spec>`

  Optionally specifies the Dimensions baseline on which to base the new project.

- `/ATTRIBUTES=(<attr1>,attr2,…)`

  Specifies the user-defined attribute values for this project.

- `/TYPE=<type-name>`

  Specifies the type of the project. If this qualifier is not specified, the type name WORKSET is used.

- `/STATUS=<status>`

  Allows the new project to be created in either a `LOCKED` or `UNLOCKED` (default) state. The `LOCKED` state prevents new Dimensions items being added to the project (baselining is likely to occur in this state).

- /BRANCH

  Optional qualifier to adopt "branching" for the item revision scheme. This means that if an item-revision is at revision maint#5, and the users decide to stay on this maint branch, then subsequent revisions will be maint#5.1, maint#5.2, maint#5.3 etc.

- /TRUNK

  Optional qualifier to adopt "trunking" for the item revision scheme. This means that if an item-revision is at revision maint#5, and the users decide to stay on this maint branch, then subsequent revisions will be maint#6, maint#7, maint#8 etc.

- /AUTO_REV

  Optional qualifier to tell Dimensions to automatically generate a new revision each time an item-spec is edited/updated.

- /NOAUTO_REV

  Optional qualifier to tell Dimensions **not** to automatically generate a new revision each time an item-spec is edited/updated, and instead request the user to supply a revision.

- /VALID_BRANCHES=(<branch-id1>,<branch-id2>,…)

  Identifies one or more branches – each previously defined in a Define (Item) Version Branches (DVB) command – that are to be valid for new item revisions created in this new project. If this parameter is omitted, an empty list is created – the Set Project Attributes (SWS) command can subsequently be used, if desired, to associate a valid list of branch-ids to the project created here.

  This list defines the branches on which newly created item revisions can be placed.

  If the project is defined with **one or more** valid branches, every **new** item revision in the new project must use one of these branch-ids.

  If the project is defined with *no* valid branches, new revisions with no branch-ids in them can continue to be used

- /DEFAULT_BRANCH=<branch-id>

  Selects, from the valid list of branch-ids, the <branch-id> to be the default branch. If a default branch-id is not defined, the first branch-id in the valid-list of branch-ids is taken as the default.

- /DEPLOYMENT_MODEL=MANUAL or /DEPLOYMENT_MODEL=AUTOMATIC

  Specifies whether the project uses a manual or automatic deployment model.

- /COPY_ON_DEPLOY

  Optional qualifier that specifies whether files that are deployed from an earlier stage to a newer stage will be copied between areas rather than moved (they are moved by default).

- /KEEP_STAGE

  When creating a project based on another project, specify this optional qualifier to avoid resetting the stage of item revisions in the new project to the initial stage and to instead keep the stage of the item revisions from the source project. The default behavior (when this qualifier is not specified) is to reset the stage of all items in the new project to the initial stage. Note that this qualifier can only be used when the new project uses the manual deployment model.

Description

The DWS command is used to create a new project within a Dimensions product. The project can be empty, based on an existing project or based on an existing baseline.

NOTE The /PROJECT, /FILENAME and /POPULATE qualifiers are no longer applicable in DWS. In order to assign deployment areas to a project and populate them, use the RAWS command.

When DWS populates a project using the /WORKSET or /BASELINE qualifiers, it makes a shallow copy of the populating collection; that is, the project or baseline directory structure is copied and equivalent relationships are created to any child collections. The relative locations of child collections are preserved, but per-user collection roots are not copied.

The Remove Project (RWS) command is used to remove a project.

NOTE The /BRANCH, /TRUNK, /AUTO_REV, /NOAUTO_REV, and DEPLOYMENT_MODEL qualifiers may further be used to alter the options associated with the project. The permitted combinations of these qualifiers are:

```
DWS <project-spec> /BRANCH
DWS <project-spec> /TRUNK
DWS <project-spec> /AUTO_REV
DWS <project-spec> /NOAUTO_REV
DWS <project-spec> /BRANCH /AUTO_REV
DWS <project-spec> /BRANCH /NOAUTO_REV /DEPLOYMENT_MODEL=MANUAL
DWS <project-spec> /TRUNK /AUTO_REV
DWS <project-spec> /TRUNK /NOAUTO_REV
```

Constraints

Normally this command can be run only by a user with the appropriate management privileges for the project concerned.

This constraint can, however, be relaxed using the Set Project Permissions (SWSP) command, as described on page 342.

DWSD – Delete Project Directory

NOTE This command is not available in the Issue Management–only licensed version of Dimensions.

```
<directory-path>
[/WORKSET=<project-spec>]
[/[NO]RECURSIVE]
[/[NO]REMOVE_ITEMS]
```

Example `DWSD src`

- `<directory>`

  The DWSD command deletes a project-directory `<directory>` from the database project structure relative to the working location (this subdirectory previously having been used for project operations, but no longer being required).

- `/WORKSET=<project-spec>` comprises:

  `<product-id>:<project-id>`

  This optionally specifies the project to be used for this command: failing this, the user's current project will be taken.

By default, DWSD fails if the specified project directory contains subdirectories or items. The following two qualifiers enable you to remove such a directory from a project.

- `/RECURSIVE`

  Specifies that the DWSD command will be applied first to each subdirectory of the specified project directory. If deletion of any subdirectory fails (for example, if the subdirectory contains items and /REMOVE_ITEMS is not in effect), the overall command fails as well; otherwise, the DWSD command is applied last to the specified project directory itself.

- `/REMOVE_ITEMS`

  Specifies that the DWSD command will remove from the current project each revision of each item in the specified project directory before attempting to delete the project directory itself. If /RECURSIVE is specified also, DWSD will recursively remove item revisions from each subdirectory of the specified project directory as well.

NOTE Whenever a new revision is added to a project, its stage is reset to DEVELOPMENT, and associated deployment areas and reference areas are updated.

Constraints

Normally this command can be run only by a user with the appropriate management privileges for the project concerned.

This constraint can, however, be relaxed using the Set Project Permissions (SWSP) command, as described on page 342.

ECDI – Extract (Check Out) Request Items

```
<request-id>
[/[NO]CANCEL_TRAVERSE]
[CHANGE_DOC_IDS=(<request1>,<request2>,…)]
[DIRECTORY=<project-directory-filter>]
[/[NO]RECURSIVE]
[/LOGFILE=<log-file>]
[/[NO]OVERWRITE]
[/[NO]TOUCH]
[/USER_DIRECTORY=<target-directory>
[/WORKSET=<project>]
[/NOMETADATA]
```

Example `ECDI PAYROLL_CR1`

Parameters

- `<request-id>`

  The name of a Dimensions request.

- `/CANCEL_TRAVERSE`

  By default, all requests related as dependent to the specified request are processed by this command. This qualifier forces the command to process only the request specified.

- `/CHANGE_DOC_IDS=(<request1>,<request2>,…)`

  Specifies requests that are to be related to all extracted items.

- `/DIRECTORY`

  Enables you to specify a project directory filter to restrict the number of items checked out.

- `/RECURSIVE`

  Used with /DIRECTORY, this specifies that the filter is to be processed recursively; that is, subdirectories are to be processed.

- `/LOGFILE`

  Specifies a local log file to which the command is to divert all messages.

- `/OVERWRITE`

  Specifies that Dimensions should overwrite files on disk with files processed by this command.

- `/TOUCH`

  Assigns the current date and time to extracted files.

- `/USER_DIRECTORY`

  Specifies the directory to which files are to be checked out.

- `/WORKSET`

  Specifies the project to be processed by this command.

- /NOMETADATA

  This parameter disables creation and usage of metadata files in the local work area.

Description

This command checks out all the items that are related to a specified request.

When multiple revisions of the same item are related to requests processed by this command, only the latest is processed.

EI – Extract (Check Out) Item for Update

NOTE This command is not available in the Issue Management–only licensed version of Dimensions.

```
<item-spec>
[/ROOT_PROJECT=<project-spec>]
[/FILENAME=<file-name>]
[/BASELINE=<baseline-spec>]
[/USER_FILENAME=<user-filename>]
[/REVISION=<new-revision>]
[/ATTRIBUTES=(<attr1>=<value1>,<attr2>=<value2>,…)]
[/CHANGE_DOC_IDS=(<request1>,<request2>,…)]
[/WORKSET=<project-spec>]
[/[NO]OVERWRITE]
[/CODEPAGE=<code-page>|DEFAULT|SOURCE]
[/TOUCH]
[/NOMETADATA]
```

Examples

```
EI PROD:"QUERY RELEASE".AAAA-SRC;1 /CHANGE=PROD_DC_12
```

```
EI PROD:;1 /FILE=qr.c /CHANGE=PROD_DC_12
```

```
EI "FS:CBEVENT C.A-SRC;b1#4" -
/USER_FILENAME="e:\cpjtest\cbevent.c" -
/REVISION="b1#5" -
/OVERWRITE
```

- `<item-spec>`

  Comprises:

  `<product-id>:<item-id>.<variant>–<item-type>;<revision>`

`<item-id>`	may be omitted if `<file-name>` is specified.
`<variant>`	may be omitted if only one exists.
`<revision>`	defaults to the latest revision in the project specified by /WORKSET. If /WORKSET is unspecified, the user's current project will be assumed.

- `/ROOT_PROJECT=<project-spec>`

  Comprises:

  `<product-id>:<project-id>`

  This optionally specifies the root project. Use this when the current project set via SCWS (or the project specified by the /WORKSET qualifier) occurs in more than one project tree.

- `/FILENAME=<file-name>`

  Specifies the name of the project file name. If /ROOT_PROJECT is used to specify a the root project, /FILENAME is interpreted in the scope of that project.

The project file name identifies the relative path (directory plus file name) from the working location of the file to be used when the item is checked out from the current project. The project file name for the same item may *differ* between projects; for example, `src/hello.c`, `hello.c`, or `src/build/hello.c`.

It may be omitted if `<item-id>` is specified.

- `/BASELINE=<baseline-spec>`

  Specifies a release-baseline which contains the particular revision of `<item-spec>` to be selected. (As it is in a baseline, a new revision must of course be checked out as a copy of it.) `<baseline-spec>` comprises:

  ```
  <product-id>:<baseline-id>
  ```

  If omitted, the specified or default `<revision>` (as described above) is selected for checking out.

- `/USER_FILENAME=<user-filename>`

  Specifies the name of the file which will be created in the user-area, and into which the item will be copied.

  If omitted, it defaults to the project file name – i.e. the file created in the user area (the current directory) will have the same name as that of the item's project file name.

- `/REVISION=<new-revision>`

  Specifies a new revision for the item. This new revision will be placed in the project specified by `/WORKSET`. If `/WORKSET` is omitted, then the new revision will be placed in the user's default project.

  If omitted, Dimensions increments the current revision (the rightmost subfield only), unless the item revision in `<item-spec>` is at its initial lifecycle state. In this case, the revision is unchanged.

- `/ATTRIBUTES=(<attr1>=<value1>,<attr2>=<value2>,…)`

`<attrN>`	is the Variable Name defined for one of the user-defined attributes for items, which has also been declared usable for the `<product-id>` and `<item-type>` specified in `<item-spec>`.
`<valueN>`	is the substitution value to be given to this attribute.

- `/CHANGE_DOC_IDS=(<request1>,<request2>,…)`

`<requestN>`	identifies a request to which the item revision is to be related **in-Response-to**.

- `/WORKSET=<project-spec>` comprises:

  ```
  <product-id>:<project-id>
  ```

  If specified, the new revision of the item will be placed in the project, otherwise it will be placed in the user's current project.

- `/[NO]OVERWRITE`

  When checking out an item revision, specify whether or not Dimensions is allowed to perform this operation depending on:

- The existence of a local file of the same name.
- The status (read-only or writeable) of an existing local file of the same name.

/NOOVERWRITE – which is normally the default but which can be reassigned using the SET OVERWRITE command described on page 328 – results in a file only being successfully checked out by Dimensions if the local (target) file does not already exist or is marked read-only. This is the traditional Dimensions behavior (with respect to the file being read-only, the assumption is that if it is writable then the file could potentially be a more recently modified revision of the item that the user does not want to lose).

/OVERWRITE results in the file being successfully checked out by Dimensions irrespective of the existence or writeable status of any local (target) file.

To clarify the above, consider the following example:

```
EI "FS:CBEVENT C.A-SRC;b1#4" -
        /USER_FILENAME="e:\cpjtest\cbevent.c" -
        /REVISION="b1#5" /OVERWRITE
```

would always overwrite cbevent.c if it existed, irrespective of whether it was marked read-only or not.

- /CODEPAGE=<code-page>|DEFAULT|SOURCE

Specifies the *code page* to be associated with the item. The code page defines the method of encoding characters. It encompasses both the different ways characters are encoded on different platforms (EBCDIC on z/OS and ASCII on Windows and UNIX) and differences between human languages. Every item in Dimensions has a code page associated with it, this being defined or derived for the connection setting or an individual item.

The /CODEPAGE parameter defaults to the code page specified when the connection between the database server and the logical node on which the user file resides was created. Whenever the item moves between platforms, for example, on a check-out from the mainframe to a PC, if the code page for the target platform is different to the item's code page, Dimensions automatically converts the item.
/CODEPAGE is relevant only for text files, because whenever a text file is checked out or gotten, it must be in the right code page for the target platform, so that it displays correctly. Binary files are moved between platforms with no conversion.

For further details concerning code pages and logical nodes see the *Distributed Development Guide*.

You are advised to let the parameter default to the code page for the item type or the platform.

The /CODEPAGE options available are:

<code-page>	Specify one of the code page values listed in the text file codepage.txt, located on your Dimensions server in the codepage subdirectory of the Dimensions installation directory. This file also provides more information about translation between code pages.
DEFAULT	Use the code page specified for the target node connection.
SOURCE	Use whatever code page was associated with the item during check in. This option is relevant only on commands that take an item out of a library i.e. FI and EI.

- /TOUCH

  Sets the modification time of the user file to the current system time instead of the modification time stored in Dimensions for this item revision.

- /NOMETADATA

  This parameter disables creation and usage of metadata files in the local work area.

Constraints

This command can be run by users who have one of the roles required to action the item from the initial lifecycle state to a new state.

By default, users can check out different revisions of an item in parallel unless a user with the role of PRODUCT-MANAGER has disabled this facility to allow only one revision of an item to be checked out at any given time.

IMPORTANT! The maximum size of any item in the database is 2GB.

EXIT – End Dimensions Execution

```
[NO PARAMETERS]
```

NOTE EXIT cannot be run from Dimensions for z/OS.

EXIT may be used (but is not essential) to mark the end of a command file which is specified to CMD.

Constraints

None

FBI – Fetch (Get) Baseline Items

NOTE This command is not available in the Issue Management–only licensed version of Dimensions.

```
<baseline-spec>
[/DIRECTORY=<directory>]
[/USER_DIRECTORY=<user-directory>]
[/[NO]EXPAND]
[/[NO]OVERWRITE]
[/CODEPAGE<code-page>|DEFAULT|SOURCE]
[/[NO]TOUCH]
[/[NO]RECURSIVE]
[/METADATA]
```

Examples: The following command fetches the specified baseline to the specified remote directory on a tertiary node. Each file is touched as it is fetched.

```
FBI "PVCS:DM9000 BUILD 5.3" /TOUCH /USER_DIRECTORY="stal-dev-lx1::/
    build/Sources/Dm9000"
```

The following command fetches all items from the `build` directory in the specified baseline.

```
FBI PVCS:DM9000 /DIRECTORY="build" /
    USER_DIRECTORY="E:\Dimensions\dim90-build.2004"
```

Description

The FBI command fetches (gets) all item revisions from a baseline or a baseline directory, regardless of the current project.

Constraints

For each item that FBI fetches (gets) in the specified baseline, FBI executes the FI command. The constraints for that command are as follows: "This command can be run by users who have a role on the design part owning the item or a role on one of the ancestor nodes of that design part."

FCDI – Fetch (Get) Request Items

```
<request-id>
[/[NO]CANCEL_TRAVERSE]
[DIRECTORY=<project-directory-filter>]
[/[NO]RECURSIVE]
[/[NO]EXPAND]
[/LOGFILE=<log-file>]
[/[NO]OVERWRITE]
[/[NO]TOUCH]
[/USER_DIRECTORY=<target-directory>
[/WORKSET=<project>]
[/NOMETADATA]
```

Example `FCDI PAYROLL_CR1`

Parameters

- `<request-id>`

  The name of a Dimensions request.

- `/CANCEL_TRAVERSE`

  By default, all requests related as dependent to the specified request are processed by this command. This qualifier forces the command to process only the request specified.

- `/DIRECTORY`

  Enables you to specify a project directory filter to restrict the number of items fetched.

- `/RECURSIVE`

  Used with /DIRECTORY, this specifies that the filter is to be processed recursively; that is, subdirectories are to be processed.

- `/EXPAND`

  Specifies expansion of all fetched files.

- `/LOGFILE`

  Specifies a local log file to which the command is to divert all messages.

- `/OVERWRITE`

  Specifies that Dimensions should overwrite files on disk with files processed by this command.

- `/TOUCH`

  Assigns the current date and time to fetched files.

- `/USER_DIRECTORY`

  Specifies the directory to which files are to be fetched.

- `/WORKSET`

  Specifies the project to be processed by this command.

- /NOMETADATA

  This parameter disables creation and usage of metadata files in the local work area.

Description

This command fetches all the items that are related to a specified request.

When multiple revisions of the same item are related to requests processed by this command, only the latest is processed.

FI – Fetch (Get) Item

NOTE This command is not available in the Issue Management–only licensed version of Dimensions.

```
<item-spec>
[/ROOT_PROJECT=<project-spec>]
[/FILENAME=<file-name>]
[/BASELINE=<baseline-spec>]
[/USER_FILENAME=<user-filename>]
[/[NO]EXPAND]
[/WORKSET=<project-spec>]
[/[NO]OVERWRITE]
[/CODEPAGE=<code-page>|DEFAULT|SOURCE]
[/[NO]TOUCH]
[/NOMETADATA]
```

Examples

```
FI PROD:"QUERY RELEASE".AAAA-SRC;1
FI "FS:CBEVENT C.A-SRC;b1#4" -
/USER_FILENAME="e:\cpjtest\cbevent.c"
/NOEXPAND -
/NOOVERWRITE
```

- `<item-spec>`

  Comprises:

  ```
  <product-id>:<item-id>.<variant>-<item-type>;<revision>
  ```

  `<item-id>` may be omitted if `<file-name>` is specified.

  `<variant>` may be omitted if only one exists.

  `<revision>` defaults to the latest revision (see Introduction on page 16).

- `/ROOT_PROJECT=<project-spec>`

  Comprises:

  `<product-id>:<project-id>`

  This optionally specifies the root project. Use this when the current project set via SCWS (or the project specified by the /WORKSET qualifier) occurs in more than one project tree.

- `/FILENAME=<file-name>`

  Specifies the name of the project file name. If /ROOT_PROJECT is used to specify a the root project, /FILENAME is interpreted in the scope of that project.

  The project file name identifies the relative path (directory plus file name) from the working location of the file to be used when the item is gotten from the current project. The project file name for the same item may *differ* between projects; for example, `src/hello.c`, `hello.c`, or `src/build/hello.c`.

  It may be omitted if `<item-id>` is specified.

- /BASELINE=<baseline-spec>

  Specifies a release-baseline which contains the particular revision of <item-spec> to be gotten. It comprises:

  ```
  <product-id>:<baseline-id>
  ```

  If omitted, the specified or default <revision> (as described above in <item-spec>) is gotten.

- /USER_FILENAME=<user-filename>

  Specifies the name of the file which will be created in the user area, and into which the item will be copied.

  If the user file name is omitted, it will default (with the exception described below) to the project file name – i.e. the file created in the user area (the current directory) will have the same name as that of the item's project file name.

NOTE If the FI command is submitted via the Dimensions desktop client command-line interfaces, the /USER_FILENAME qualifier is **compulsory**.

- /NOEXPAND

  When getting the item, request Dimensions **not** to expand any substitution variables (file heading text and/or Dimensions-defined or user-defined attributes) located in the item header. This is analogous to what happens when an item is *checked out*.

  Refer to "Item Format Templates", in the *Process Modeling User's Guide* for a discussion of item header substitution.

  The default is /EXPAND provided item type was defined in the process model to request **item header substitution**.

- /WORKSET=<project-spec> comprises:

  ```
  <product-id>:<project-id>
  ```

  This optionally specifies the project to be used for this command: failing this, the user's current project will be taken.

  Item revisions to be affected by the command may be specified explicitly, or they will be selected from the project.

- /[NO]OVERWRITE

  When getting an item revision, specify whether or not Dimensions is allowed to perform this operation depending on:

  - The existence of a local file of the same name.
  - The status (read-only or writeable) of an existing local file of the same name.

  /NOOVERWRITE – which is normally the default but which can be reassigned using the SET OVERWRITE command described on page 328 – results in a file only being successfully gotten by Dimensions if the local (target) file does not already exist or is marked read-only. This is the traditional Dimensions behavior (with respect to the file being read-only, the assumption is that if it is writable then the file could potentially be a more recently modified revision of the item that the user does not want to lose).

  /OVERWRITE results in the file being successfully gotten by Dimensions irrespective of the existence or writeable status of any local (target) file.

To clarify the above, consider the following example:

```
FI "FS:CBEVENT C.A-SRC;b1#4" -
       /USER_FILENAME="e:\cpjtest\cbevent.c" /NOEXPAND -
       /NOOVERWRITE
```

That would not allow `cbevent.c` to be overwritten if it existed and was not marked read-only.

- `/CODEPAGE=<code-page>|DEFAULT|SOURCE`

  Specifies the *code page* to be associated with the item. The code page defines the method of encoding characters. It encompasses both the different ways characters are encoded on different platforms (EBCDIC on z/OS and ASCII on Windows and UNIX) and differences between human languages. Every item in Dimensions has a code page associated with it, this being defined or derived for the connection setting or an individual item.

  The `/CODEPAGE` parameter defaults to the code page specified when the connection between the database server and the logical node on which the user file resides was created. Whenever the item moves between platforms, for example, on a check-out from the mainframe to a PC, if the code page for the target platform is different to the item's code page, Dimensions automatically converts the item.
  `/CODEPAGE` is relevant only for text files, because whenever a text file is checked out or gotten, it must be in the right code page for the target platform, so that it displays correctly. Binary files are moved between platforms with no conversion.

  For further details concerning code pages and logical nodes see the *Distributed Development Guide*.

  You are advised to let the parameter default to the code page for the item type or the platform.

  The `/CODEPAGE` options available are:

`<code-page>`	Specify one of the code page values listed in the text file `codepage.txt`, located on your Dimensions server in the `codepage` subdirectory of the Dimensions installation directory. This file also provides more information about translation between code pages.
`DEFAULT`	Use the code page specified for the target node connection.
`SOURCE`	Use whatever code page was associated with the item during check in. This option is relevant only on commands that take an item out of a library i.e. FI and EI.

- `/TOUCH`

  Sets the modification time of the user file to the current system time instead of the modification time stored in Dimensions for this item revision.

- `/NOMETADATA`

  This parameter disables creation and usage of metadata files in the local work area.

Constraints

This command can be run by users who have a role on the design part owning the item or a role on one of the ancestor nodes of that design part.

FIF – Find Item File

NOTE This command is not available in the Issue Management–only licensed version of Dimensions.

```
<item-spec>
[/FILENAME=<file-name>]
[/WORKSET=<project-spec>]
```

Example `FIF PROD:"QUERY RELEASE".AAAA-SRC;2`

- `<item-spec>`

  Comprises:

  ```
  <product-id>:<item-id>.<variant>– <item-type>;<revision>
  ```

  `<item-id>` may be omitted if `<file-name>` is specified.

  `<variant>` may be omitted if only one exists.

  `<revision>` may be omitted if the file version is not required.

- `/FILENAME=<file-name>`

  Specifies the name of the project file name.

  The project file name identifies the relative path (directory plus file name) from the working location to the item to be used from the current project. The project file name for the same item may *differ* between projects; for example, `src/hello.c`, `hello.c`, or `src/build/hello.c`.

  It may be omitted if `<item-id>` is specified.

- `/WORKSET=<project-spec>`

  Comprises:

  ```
  <product-id>:<project-id>
  ```

  This optionally specifies the project to be used for this command: failing this, the user's current project will be taken.

  Item revisions to be affected by the command may be specified explicitly, or they will be selected from the project.

Description

This program displays an item's file name, including full directory path and optionally the revision number.

Non-Dimensions operations (e.g. compilation) can sometimes be performed by reading the item directly from the item library (provided that the user has been granted normal operating system read access to the library).

NOTE *For UNIX systems only*. The revision number, which is shown preceded by a semicolon (;), is in fact preceded by a dot (.) in the file name.

Constraints

This command can be run only by users who have a role for the owner part.

The command is not available for items held in delta libraries.

FRC – Forward a Release to a Customer

NOTE This command is not available in the Issue Management–only licensed version of Dimensions.

```
<release-spec>
/CUSTOMER=<name>
/LOCATION=<location>
/PROJECT=<project-spec>
[/DESCIPTION=<description>]
```

Example

```
FRC PROD:"R M 2.0 FOR HP"
/CUSTOMER="Brown Finances -
/LOCATION="Bristol"
/PROJECT="PAYROLL"
```

- `<release-spec>`

  Specifies the releases-spec, which comprises:

  ```
  <product-id.:<release-id>
  ```

- `/CUSTOMER=<name>`

  Specifies the customer's name.

- `/LOCATION=<location>`

  Specifies the customer's physical location.

- `/PROJECT=<project-spec>`

  Specifies the project name.

- `/DESCRIPTION=<description>`

  Optionally associate a description of the release.

Description

The Dimensions product allows you to maintain a list of customers and a record of which Dimensions releases have been sent to each customer.

The FRC command enables you to record the fact that a release has been supplied to a specific customer.

Constraints

Only users with the appropriate management privileges can run this command.

The combination of customer name, location, and project-spec must be unique in the Dimensions database.

You cannot forward the same release to a customer twice.

FWI – Fetch (Get) Project Items

NOTE This command is not available in the Issue Management–only licensed version of Dimensions.

```
<project-spec>
[/DIRECTORY=<directory>]
[/USER_DIRECTORY=<user-directory>]
[/[NO]EXPAND]
[/[NO]OVERWRITE]
[/CODEPAGE=<code-page>|DEFAULT|SOURCE]
[/[NO]TOUCH]
[/[NO]RECURSIVE]
[/NOMETADATA]
```

Examples: The following command recursively fetches (gets) the entire specified project into the user's project directory. Each file is touched as it is fetched.

```
FWI PVCS:DM9000 /TOUCH
```

The following command recursively fetches (gets) all items from the `build` directory into the specified user directory.

```
FWI PVCS:DM9000 /DIRECTORY="build"
    /USER_DIRECTORY="E:\Dimensions\dim90-build.2004"
```

The following command fetches (gets) only items in the `install` directory, excluding any subdirectories, into the specified remote directory on a tertiary node.

```
FWI PVCS:DM9000 /DIRECTORY="install" /NORECURSE /USER_DIRECTORY="stal-
    dev-lx1::/builds/dim90-build.2004"
```

Description

The FWI command fetches (gets) the latest item revisions from an entire project or a project directory. By default, FWI traverses all subdirectories when looking for item revisions to fetch.

Constraints

For each item that FWI fetches (gets) in the specified project, FWI executes the FI command. The constraints for that command are as follows: "This command can be run by users who have a role on the design part owning the item or a role on one of the ancestor nodes of that design part."

GREP – Search and Replace

NOTE This command is not available in the Issue Management–only licensed version of Dimensions.

```
/SEARCH=<text>
[/TEXT_REPLACE=<text>]
[/FILENAME=<text>]
[/COMMENT=<text>]
[/USER_FILENAME=<text>]
[/[NO]IGNORE_CASE]
[/LATEST_REV]
[/[NO]RECURSIVE]
[/PRODUCT=<text>]
[/TYPE_LIST=(type,type)]
[/FORMAT_LIST=(format,format)]
[/WORKSET=<project-spec>]
```

One of the following must also be specified…

```
/WS_DIR=<directory>
/PART=<part specification>
/CHANGE_DOC_ID=<doc id>
```

Examples

```
GREP  /SEARCH="printf"  /PRODUCT="PAYROLL"
    /FILENAME="%.c" /WS_DIR="src\gui"  /LATEST_REV
    /IGNORE_CASE
```

will cause Dimensions to search for all occurrences of `printf` ignoring case, in all `c` files in the project directory `src\gui` owned by product **PAYROLL** looking only at the latest revision.

```
GREP  /SEARCH="printf" /FILENAME="%c,%.h,%.cpp"
    /TYPE_LIST=(DAT) /PART="PETRAY:PETRAY.A;1"
    /RECURSIVE
```

will find all items which are `CPP`, `C` or `H` files of item type `DAT` which contain the word `printf` (the case must match) owned by the design part PETRAY or any of its children (recursively).

```
GREP /SEARCH="strncasecmp" /TEXT_REPLACE="strnicmp"
    /FILENAME="domain%.c" CHANGE_DOC_ID="PAYROLL_CR_1"
    /COMMENT="replace string routines"
    /USER_FILE="C:\output.log"
```

will find all items containing `strncasecmp` and create new revisions of the matching items replacing occurrences of `strncasecmp` with `strnicmp`. The comment `"replace string routines"` will be used as the comment for the creation of the new item revisions. Only items related to request `PAYROLL_CR_1` will be searched and the output of the command will be placed in the log file `"C:\output.log"`.

- `/SEARCH=<text>`

  Specifies the text to find.

- /TEXT_REPLACE=<text>

  Specifies the string with which to replace found values.

- /FILENAME=<text>

  Specifies a filter for matching files.

- /COMMENT=<text>

  Specifies comment used for newly created revisions only valid if REPLACE is specified.

- /USER_FILE=<text>

  Specifies optional file to contain output of GREP command.

- /IGNORE_CASE

  ignore case when searching.

- /LATEST_REV

  look at only the latest revision.

- /RECURSIVE

  causes search to be recursive from the given object.

- /PRODUCT=<text>

  only look at items of this product.

- /TYPE_LIST=(type,type)

  only look at items of these types.

- /FORMAT_LIST=(format,format)

  only look at items of these formats.

- /WORKSET=<project-spec> comprises:

  <product-id>:<project-id>

  This specifies the project to be used for this command: failing this, the user's current project will be taken. The WS_DIR qualifier can be used to further scope the nature of the search.

- /WS_DIR=<directory>

  Specifies the project directory to search

NOTE To search the top directory in a project use only the forward slash character (/).

- /PART=<part specification>

  Specifies the design part to search

- /CHANGE_DOC_ID=<doc id>

  Specifies the request to search from.

  All item revisions that are related, either as **Affected** or **in Response to**, will be considered for this command.

Constraints

This command can be run only by users who have the role to get the relevant items.

The GREP command is only for use with files of an ASCII text format. Binary files such as Microsoft Word documents are not searched, they are simply ignored.

HELP – Help

```
<command-name>
```

Example `HELP download`

- `<command-name>`

  The command for which you want a usage summary.

Description

Displays a usage summary, including required and optional qualifiers, for the specified command.

LA – List Areas

NOTE This command is not available in the Issue Management–only licensed version of Dimensions.

```
<area-name>
[/TYPE=<area-type>]
[/WORKSET=<project-spec> [/STAGE=<stage>]]
[/NETWORK_NODE=<machine-name>]
[/OWNER=<user-or-group>]
[/STATUS=ONLINE or OFFLINE]
```

Example `LA <area-name> /TYPE=DEPLOYMENT`

Parameters

- `<area-name>`

  Specifies the name of the area of which to list details. If this parameter is omitted, the list of areas is potentially filtered by other optional qualifiers.

- `<area-type>`

  WORK or DEPLOYMENT. If this parameter is specified, only areas of this type will be included in the list.

- `<project-spec>`

  If this parameter is specified, only areas related to this project will be included in the list.

- `<stage>`

  If this parameter is specified, only areas associated with this stage will be included in the list.

- `<machine-name>`

  If this parameter is specified, only areas defined for this node will be included in the list.

- `<user-or-group>`

  If this parameter is specified, only areas owned by this user or group will be included in the list.

- `/STATUS=ONLINE or OFFLINE`

  If this parameter is specified, only areas of the specified status will be included in the list.

Description

The LA command lists details of the specified area or all areas matching the specified criteria. If no parameters are provided, the command lists details of all areas.

Constraints

Only users with the Run Admin Reports privilege can run this command.

LBA – List Build Areas

NOTE This command is no longer available; use LA (List Areas) instead.

See the LA command.

LBDB – List Existing Base Database Entries

No parameters.

Example LBDB

This command enables you to list existing registered base databases in an installation's network administration tables. See the *Administrator's Guide* for details.

LBPROJ – List Dimensions Build Projects

NOTE This command is no longer available; use LWS instead.

LCK – Lock Project

NOTE This command is not available in the Issue Management–only licensed version of Dimensions.

```
workset <project-spec>
```

Example `LCK WORKSET PROD_X:TEST_WS`

- `<project-spec>` comprises:

  `<product-id>:<project-id>`

  The specified project must exist.

Description

This command locks the project, with project as a fixed parameter and `<project-spec>` a user-defined parameter.

The locked state prevents the addition of new Dimensions items to the project or the removal of existing item revisions that are in a locked project (baselining is likely to occur in this state). In addition, items that exist in a locked project will not be actionable or updateable from another project unless the user has the role of `PRODUCT-MANAGER` for the item's product.

Users with the `PRODUCT-MANAGER` role can create new items in a locked project.

Constraints

This command can be run only by a user with the appropriate management privileges for the project concerned.

LCO – List Existing Contacts

No parameters.

Example LCO

This command enables you to list an installation's contacts. See the *Administrator's Guide* for details.

LCST – List Existing Codesets

No parameters.

Example LCST

This command enables you to list an installation's codesets. See the *Administrator's Guide* for details.

LFS – List Existing File Systems

No parameters.

Example `LFS`

This command enables you to list an installation's file systems. See the *Administrator's Guide* for details.

LGRP – List Groups

```
[/[NO]DETAIL]
```

Description

This command lists all available groups and, If /DETAIL is specified, group members.

Constraints

Only users with the appropriate management privileges can run this command.

LINS – List Existing Database Instance Entries

No parameters.

Example `LINS`

This command enables you to list existing registered database instances in an installation's network administration tables. See the *Administrator's Guide* for details.

LLCA – List Library Cache Areas

NOTE This command is not available in the Issue Management–only licensed version of Dimensions.

```
<area-name>
[/NETWORK_NODE=<machine-name>]
[/OWNER=<user-or-group>]
[/STATUS=ONLINE or OFFLINE]
```

Example `LLCA <area-name> /TYPE=DEPLOYMENT`

Parameters

- `<area-name>`

  Specifies the name of the area of which to list details. If this parameter is omitted, the list of areas is potentially filtered by other optional qualifiers.

- `<machine-name>`

  If this parameter is specified, only areas defined for this node will be included in the list.

- `<user-or-group>`

  If this parameter is specified, only areas owned by this user or group will be included in the list.

- `/STATUS=ONLINE or OFFLINE`

  If this parameter is specified, only areas of the specified status will be included in the list.

Description

The LLCA command lists details of the specified library cache area or all library cache areas matching the specified criteria. If no parameters are provided, the command lists details of all library cache areas.

Constraints

Only users with the Run Admin Reports privilege can run this command.

LMNR – List Mail Notification Rules

```
(no parameters)
```

Description

This command lists all mail notification rules.

LNC – List Existing Network Node Connections

No parameters.

Example LNC

This command enables you to list an installation's existing network node connections. See the *Administrator's Guide* for details.

LNDO – List Existing Network Node Objects

No parameters.

This command enables you to list existing network nodes. See the *Administrator's Guide* for details.

LNN – List Existing Network Nodes

No parameters.

Example `LNN`

This command enables you to list an installation's existing network nodes. See the *Administrator's Guide* for details.

LNWO – List Existing Network Objects

No parameters.

Example `LNWO`

This command enables you to list an installation's existing network objects. See the *Administrator's Guide* for details.

LOS – List Existing Operating Systems

No parameters.

Example LOS

This command enables you to list an installation's existing operating systems. See the *Administrator's Guide* for details.

LPRIV – List Privileges

```
(no parameters)
```

Description

This command lists all privileges.

LPROJ – List Dimensions Projects and Build Projects

NOTE This command is no longer available; use LWS instead.

LPRP – List Preservation Rules Policies

NOTE This command is not available in the Issue Management–only licensed version of Dimensions.

```
<product-id>
```

Example The following command:

```
LPRP PAYROLL
```

list preservation rules policies defined in the PAYROLL product.

- `<product-id>`

  Specifies the product restricting the list of policies to be displayed.

Description

List preservation rules policies defined in a product (for information about preservation rules policies, see "DPRP – Define Preservation Rules Policy" on page 160).

Constraints

None.

LPRT – List Existing Network Protocols

No parameters.

This command enables you to list existing network protocols used by an installation network object. See the *Administrator's Guide* for details.

LPSP – List Per-Stage Project Properties

NOTE This command is not available in the Issue Management–only licensed version of Dimensions.

```
<project-spec>
```

Example The following command:

```
LPSP "EXEDLL:EXEDLL 2.0"
```

list per-stage project stage properties set in the "EXEDLL:EXEDLL 2.0" project.

- `<project-spec>`

  Comprises `<product-id>:<project-id>` and specifies the project specification.

Description

List per-stage project properties set in a project (for information about per-stage project properties, see "SPSP – Set Per-Stage Preservation Policy" on page 332).

Constraints

None.

LRSD – List Existing Resident Software Definitions

NOTE This command is not available in the Issue Management–only licensed version of Dimensions.

No parameters.

Example LRSD

This command enables you to list an installation's existing Resident Software Definitions (RSDs). See the *Administrator's Guide* for details.

LSAR – List Archives

NOTE This command is not available in the Issue Management–only licensed version of Dimensions.

This command lists all archives created by ART.

LSBL – List Baselines

NOTE This command is not available in the Issue Management–only licensed version of Dimensions.

```
LSBL
[<product-ID>]
```

- `<product-ID>`

  Specifies a product for which baselines are to be listed.

Example Output

```
Dimensions>lsbl
Listing of baselines currently in the database for all products:
        Baseline Id: PAYROLL:BL_DEV_REL_1_A
        Created by DMSYS
        Associated Project: (None)
        Baseline Id: PAYROLL:BL_DEV_REL_1_B
        Created by DMSYS
        Associated Project: (None)
        Baseline Id: PAYROLL:BL_DEV_REL_2_A
        Created by DMSYS
        Associated Project: (None)
        Baseline Id: PAYROLL:BL_VBGUI_REL_1_A
        Created by DMSYS
        Associated Project: (None)
        Baseline Id: PAYROLL:INITIAL
        Created by DMSYS
        Associated Project: (None)
Operation completed

Dimensions>lsbl payroll
Listing of baselines currently in the database for user-specified
    product:
        Baseline Id: PAYROLL:BL_DEV_REL_1_A
        Created by DMSYS
        Associated Project: (None)
        Baseline Id: PAYROLL:BL_DEV_REL_1_B
        Created by DMSYS
        Associated Project: (None)
        Baseline Id: PAYROLL:BL_DEV_REL_2_A
        Created by DMSYS
        Associated Project: (None)
        Baseline Id: PAYROLL:BL_VBGUI_REL_1_A
        Created by DMSYS
        Associated Project: (None)
        Baseline Id: PAYROLL:INITIAL
        Created by DMSYS
        Associated Project: (None)
Operation completed
```

Description

This command supports the ISPF panels client baseline build facility.

LSTG – List Stages

NOTE This command is not available in the Issue Management–only licensed version of Dimensions.

This command does not have any options.

Sample Output

```
Dimensions>lstg
Listing of stages for this database:
        Stage Id: DEVELOPMENT
                Description: Development stage
                Created: 31-JAN-2001 16:24:53 by dmsys
        Stage Id: EMERGENCY
                Description: Emergency stage
                Created: 31-JAN-2001 16:24:53 by dmsys
        Stage Id: RELEASE
                Description: Release stage
                Created: 31-JAN-2001 16:24:53 by dmsys
        Stage Id: SYSTEM TEST
                Description: System test stage
                Created: 31-JAN-2001 16:24:53 by dmsys
        Stage Id: UNIT TEST
                Description: Unit test stage
                Created: 31-JAN-2001 16:24:53 by dmsys
Operation completed
```

Description

Lists the contents of the Global Stage Lifecycle.

LWS – List Projects

NOTE This command is not available in the Issue Management–only licensed version of Dimensions.

```
[/FILENAME=<file-name>]
```

Example `LWS /FILENAME=worklist.txt`

- `/FILENAME=<file-name>`

  Specifies that the list of projects is output into the file specified in your 'home' directory.

  LWS without this qualifier outputs the list to the screen (`stdout` on UNIX systems).

  For each project, the associated product, project-id, project-status, project-owner (the user who created the project), associated project, and users with the role of `WORKSET-MANAGER` are detailed, see example below.

Example Output

```
List of projects currently in use
=================================

        Product:         $GENERIC  WS Id:          $GLOBAL
        Status :         UNLOCKED  Owner:             ##!&
        Project Id : (None)
        Product:          PAYROLL  WS Id:              CON
        Status :             OPEN  Owner:             TIMP
        Project Id : (None)
         - Work Set Manager :             TIMP
        Product:          PAYROLL  WS Id:             CONN
        Status :             OPEN  Owner:             TIMP
        Project Id : (None)
         - Work Set Manager :             TIMP
        Product:          PAYROLL  WS Id:             FRED
        Status :             OPEN  Owner:             TIMP
        Project Id : (None)
         - Work Set Manager :             TIMP
        Product:          PAYROLL  WS Id:          PROJDFD
        Status :             OPEN  Owner:             TIMP
        Project Id : HH1
         - Work Set Manager :             TIMP
        Product:          PAYROLL  WS Id:         PROJDFDC
        Status :             OPEN  Owner:             TIMP
        Project Id : HH1
```

Constraints

None

LWSD – List Project Directories

NOTE This command is not available in the Issue Management–only licensed version of Dimensions.

```
<directory-path>
[/RECURSIVE]
[/LATEST]
[/FILES] or
   [/ITEMS] or
   [/FILES/ITEMS]
[/USER_FILENAME=<file-name>]
[/WORKSET=<project-spec>]
```

Example `LWSD src`

- `<directory>`

  The LWSD command lists:

  - The current project-id or that specified by `/WORKSET`.
  - The identities of the operating system directories of the projects at subdirectory `<directory>` relative to the working location.
  - The project description.
  - The date at which the list was taken.
  - The items the project directories contain. The following is detailed for each item: its owner, its update date, its status, whether or not it is checked out and its specification and/or file name.

- `/RECURSIVE`

  recursively list all project directories from the point defined above.

- `/LATEST`

  list only the latest (tip) revisions present in each project directory (if any).

- `/FILES`

  list items using item library file names (the default).

  - `/ITEMS`

    lists items using (traditional Dimensions) item specifications.

  - `/FILES/ITEMS`

    list items using both item-library file names and (traditional Dimensions) item specifications.

- `/USER_FILENAME=<user-filename>`

  Specifies that the list is to be output to a file `<user-filename>` rather than to the screen.

- `/WORKSET=<project-spec>` comprises:

`<product-id>:<project-id>`

This optionally specifies the project to be used for this command: failing this, the user's current project will be taken.

Constraints

None

MCPC – Move Request To Primary (Main) Catalog

```
[/CHANGE_DOC_IDS=(<request1>,<request2>,…)]
```

Example `MCPC /CHANGE=(PROD_DC_22,PROD_DC_23)`

- `/CHANGE_DOC_IDS=(<request1>,<request2>,…)`

  `<requestN>`

  Identifies the request(s) that are to be moved from the secondary catalog to the primary (main) catalog.

NOTE The secondary catalog is intended mainly for requests that are no longer active, and therefore users are not permitted to update a request in this catalog.

Constraints

Only users with the appropriate management privileges can run this command.

Update functions are available only from the main catalog.

MCSC – Move Request To Secondary Catalog

```
[/CHANGE_DOC_IDS=(<request1>,<request2>,…)]
```

Example `MCSC /CHANGE=(PROD_DC_30,PROD_DC_31)`

- `/CHANGE_DOC_IDS=(<request1>,<request2>,…)`

  `<requestN>`

  Identifies the request(s) that are to be moved from the primary (main) catalog to the secondary catalog.

NOTE The secondary catalog is intended mainly for requests that are no longer active, and therefore users are not permitted to update a request in this catalog.

Constraints

Only users with the appropriate management privileges can run this command.

Update functions are available only from the main catalog.

MDR – Move Design Part Relationship

```
<part-spec>
/PARENT_PART=<part-spec>
```

Example

```
MDR PROD:"RELEASE MANAGEMENT".AAAA -
    /PARENT_PART=PROD:"CONFIG DEF"
```

- `<part-spec>`

  (both for the moved design part and for the new owner parent part[1]) comprises:

  ```
  <prod-id>:<part-id>.<variant>;<pcs>
  ```

`<variant>`	may be omitted if only one variant of that design part exists.
`<pcs>`	is ignored; the current PCS is always used.

Constraints

Only users with the appropriate management privileges can run this command.

MI – Merge Item Revisions

NOTE This command is not available in the Issue Management–only licensed version of Dimensions.

```
<item-spec>
/REVISION_LIST=(<merge-rev-1>,<merge-rev-2>,…)
/USER_FILENAME=<merged-file>
[/REVISION=<new-revision>]
[/COMMENT=<comment-text>]
[/ROOT_PROJECT=<project-spec>]
[/FILENAME=<item-filename>]
[/CHANGE_DOC_IDS=(<request1>,<request2>,…)]
[/ATTRIBUTES=(<attr1>=<value1>,<attr2>=<value2>,…)]
[/WORKSET=<project-spec>]
[/[NO]KEEP]
[/CONTENT_ENCODING=<file-encoding>]
[/NOMETADATA]
```

Example

```
MI PROD:"QUERY RELEASE".AAAA-SRC;main#1 -
/REVISION="main#2" -
/REVISION_LIST=("patch1#1","patch2#2")
/USER_FILENAME="patch3#1"
/KEEP -
/COMMENT="Merge maintenance work into main line"
```

- `<item-spec>`

  Specifies the primary item revision that is to be merged. It comprises:

  ```
  product_id>:<item_id>.<variant>–<item-type>;<revision>
  ```

`<item-id>`	may be omitted if `<file-name>` is specified.
`<variant>`	may be omitted if only one exists.
`<revision>`	defaults to the latest revision in the project specified by /WORKSET. If /WORKSET= is unspecified, the current project will be assumed.

- `/REVISION_LIST=(<merge-rev-1>,<merge-rev-2,…)`

  Specifies each revision to be merged with the primary item revision.

- `/USER_FILENAME=<merged-file>`

  Specifies the name of the file containing the data for the merged item revision.

- `[/REVISION=new-revision>]`

  Specifies the new revision that is to be created.

- `/COMMENT=<comment text>`

  comment text to explain the reason for the creation of this merged item revision. The comment text can be up to 1978 characters long.

- /ROOT_PROJECT=<project-spec>

  Comprises:

  <product-id>:<project-id>

  This optionally specifies the root project. Use this when the current project set via SCWS (or the project specified by the /WORKSET qualifier) occurs in more than one project tree.

- /FILENAME=<file-name>

  Specifies the name of the project file name. If /ROOT_PROJECT is used to specify a the root project, /FILENAME is interpreted in the scope of that project.

  The project file name identifies the relative path (directory plus file name) from the working location to the item to be used from the current project. The project file name for the same item may *differ* between projects; for example, src/hello.c, hello.c, or src/build/hello.c.

  It may be omitted if <item-spec> is specified.

- /CHANGE_DOC_IDS=(<request1>,<request2>,…)

  <requestN> identifies one or more request to which the merged item revision is to be related in-Response-to.

NOTE Mandatory if CM rules are enabled.

- /ATTRIBUTES=(<attr1>=<value1>,<attr2>=<value2>, …)

  <attrN> is the Variable Name defined for one of the user-defined attributes, either applicable to items of all item types, or applicable to those of the <item-type> specified in <item-spec>.

  <valueN> is the value to be given to this attribute.

- /WORKSET=<project-spec> comprises:

  <product-id>:<project-id>

  This **optionally** specifies the project to be used for this command: failing this, the current project will be taken.

- [/KEEP]

  Specifies that the <user-filename> which is normally deleted once the item has been placed under Dimensions control, is to be left intact.

- <file-encoding>

  Specifies the content encoding for new item revisions to be created. Supported encodings are the Microsoft codepages, the ISO-8859 variants (1–10), UTF-8, UTF-16, UTF-16BE, UTF-16LE, UTF32, UTF32BE, and UTF32LE.

- /NOMETADATA

  This parameter disables creation and usage of metadata files in the local work area.

Description

MI is used to merge two or more revisions of the same item.

The <*new-revision*> is created from the revision identified by <*item-spec*> using the specified <*merged-file*> as the user file for <*new-revision*>. Merge records are created showing that each revision specified in /REVISION_LIST has been merged into <*new-revision*>.

Both /REVISION_LIST and /USER_FILENAME must be specified.

If a user file and /KEEP are specified, the local metadata is updated after a successful merge.

Constraints

This command can be run by users who have one of the roles required to action the item from the initial lifecycle state to a new lifecycle state.

MIP – Move Item to Another Part

NOTE This command is not available in the Issue Management–only licensed version of Dimensions.

```
<item-spec>
[/FILENAME=<file-name>]
[/PART=<part-spec>]
[/WORKSET=<project-spec>]
```

Example

```
MIP PROD:"QUERY RELEASE".AAAA-SRC -
/PART=PROD:"RELEASE CONTROL".AAAA
```

- `<item-spec>` comprises:

  `<product_id>:<item_id>.<variant>–<item-type>;<revision>`

  `<item-id>` may be omitted if `<file-name>` is specified.

  `<variant>` may be omitted if only one exists.

  `<revision>` is ignored; all revisions are moved to the specified design part.

- `/FILENAME=<file-name>`

  Specifies the name of the library file name.

- `/PART=<part-spec>` comprises:

  `<product_id>:<part_id>.<variant>;<pcs>`

  `<variant>` may be omitted if only one exists.

  `<pcs>` is ignored; the current PCS is always used.

- `/WORKSET=<project-spec>` comprises:

  `<product-id>:<project-id>`

  This optionally specifies the project to be used for this command: failing this, the user's current project will be taken.

  Item revisions to be affected by the command may be specified explicitly, or they will be selected from the project.

Constraints

1. Only users with the appropriate management privileges can run this command.
2. MIP cannot move an item to another part that belongs to a product that is different from that owing the item, that is, the <product-id> component of the <part-spec> must be identical to the <product-id> component of the <item-spec>.
3. If the item is related to a request, then the following rules apply:
   a. MIP cannot move the item to another part if the item is in an Affected or InResponseTo relationship to the request, except when the request is in a closed, rejected, or held state.
   b. MIP can move the item to another part if the item is in an Info relationship to the request, regardless of the request's state.
   c. MIP can move the item to another part if the item is in a relationship to a request in the secondary catalog.
4. MIP will fail if the part specified is already the owner of the item.
5. MIP will fail if the part specified is already in a USAGE relationship with the item.

MIT – Move (Change) Item Type

NOTE This command is not available in the Issue Management–only licensed version of Dimensions.

```
<item-spec>
/TYPE=<new-item-type>
```

CAUTION! *All* revisions of the item are moved (changed) to the new item type *regardless* of the revision (explicit or implied) that is specified in `<item-spec>`.

Example `MIT PROD:"MAIN".AAAA-SRC /TYPE=CODE`

In this example the item's type is changed from SRC to CODE.

- `<item-spec>` comprises:

  `<product_id>:<item_id>.<variant>-<item-type>;<revision>`

  `<item-id>` may be omitted if `<file-name>` is specified.

  `<variant>` may be omitted if only one exists.

  `<revision>` is ignored; all revisions are moved to the specified design part.

- `/TYPE=<new-item-type>`

  Specifies the new type to be assigned to the item.

Constraints

- Only users with the appropriate management privileges can run this command.
- You *cannot* move an item's type if the item:
  - Is stored in a Delta library.
  - Is stored in an item library on a mainframe (z/OS).
  - Is on a named branch that remotely owned e.g. the item has been replicated (see *Administrator's Guide*).
  - Is in the Dimensions OFFLINE state (see the *Administrator's Guide*).
  - Has been included in a Dimensions build.
  - Is in a Dimensions baseline.
  - Is in a Dimensions release.
  - Is checked out.
  - Is referenced in a request.
  - Has other items related to it.

- If the current state of the item does not match (by name) any state in the new item-type's lifecycle, then the state of the item is reset to the new item-type's initial lifecycle state and a warning is issued.
- If an attribute of the current item-type has a name that does not match one of the attribute names for the new item-type, then the attribute values for that name are not copied and a warning is issued.

MVC – Move Request

```
<top-request-id>
<target-product-id>
[/CH_DOC_TYPE=<target-request-type>]
[/AFFECTED_PARTS=(<target-part-spec1>,…)]
[/CHANGE_DOC_IDS=(<doc1>,<doc2>,…)]
[/[NO]CHECK]
```

Example

```
MVC OBLR_DR_52 PROD
/CHANGE_DOC_IDS=(OBLR_DR_53, OBLR_DR_55, OBLR_DR_56)
/NOCHECK
```

- `<top-request-id>`

  Identifies the (principal) request in the source product. Its dependent-related children can also be specified for moving at the same time (as detailed below).

  More exactly, the request(s) are "cloned" rather than moved: the originals are flagged to a special state `$MOVED`, once clones of them have been created in the target product.

- `<target-product-id>`

  Identifies the target product, where the request(s) is (are) to be moved. It must be another product in the same Dimensions database. (To copy requests to a different database, the baseline transfer facility of Dimensions ART may be used.)

- `/CH_DOC_TYPE=<target-request-type>`

  Specifies the request type that the clone of `<top-request-id>` is to have in the target product. The dependent children moved, if of the same type as `<top-request-id>`, are also translated to this type. (Any other dependent children do not receive a type translation.)

  If omitted, all the cloned requests in the target product are created with the same type(s) as the originals in the source product.

- `/AFFECTED_PARTS=(<target-part-spec1>,…)`

  Specifies one or more design parts in the target product that are to be related to the clone of `<top-request-id>`.

  If omitted, just the target product's top (i.e. product level) design part (its original variant) is related to this cloned request.

- `/CHANGE_DOC_IDS=(<doc1>,<doc2>,…)`

  This is a comma separated list of requests, where each `<doc`*N*`>` is of the form `<source-request-id>`.

  The entries `<source-request-id>` each identify a request that has a relationship to `<top-request-id>` in the Dependent class and that is to be included in the group move.

  If the `/CHANGE_DOC_IDS` qualifier is omitted, then just `<top-request-id>` is moved by itself.

- `/CHECK or /NOCHECK`

  Specifies whether MVC is merely to check and report on the feasibility of moving the specified request(s), or is actually to implement the move.

The default is /CHECK: **the move actually takes place only if *[/NOCHECK](#)* is specified**.

Provided the Constraints are complied with, and /NOCHECK is specified, all the source-product requests specified acquire the status $MOVED, which is regarded as a non-normal final state; i.e. the phase becomes Rejected. The History record of each identifies its clone's request-id in the target product.

The cloned children acquire the relationship Dependent to the cloned parent (regardless of any specific relationship name the children had in the source product). All clones are actioned to their initial lifecycle states, but they are not placed in any users' pending lists. The History record of each identifies the source product's request-id from which it was cloned.

For each request moved, if there are user-defined attributes which have been declared for use by the product request types of both the original and the clone, the values of these are all inherited by the clone.

Constraints

- This command can be run only by a user with the appropriate management privileges for the source product, or by users if all the specified requests are in their Pending list.
- None of these requests can have any items related as Affected or In-Response-To. Info related items are permitted, but they are simply ignored when the clones are created.
- Related requests that are to be moved must all be related to the parent as Dependent and **not** Info.
- None of these requests can be related in the Info relationship class to any other requests, in either direction.
- The <top-request-id> must not be Dependent related to any grandparent. Dependent children not specified in /CHANGE_DOC_IDS are permissible: they are left unaltered as orphans in the source product. The specified children must not be related to any request other than <top-request-id>.
- The <top-request-id> must be at its initial lifecycle state in the source product, but the children can be at any states except final states (i.e. all the requests must still be Open).
- The parameter $LAST (see note on the CC command – page 79) is not set by MVC, so the cloned requests cannot be referenced subsequently in the same CMD command file.
- This command does not copy attribute history information.
- When CM rules are on, only requests at the Create phase can be moved.

MWS – Merge Projects

NOTE This command is not available in the Issue Management–only licensed version of Dimensions.

```
<project-spec1>
<project-spec2>
[/WORKSET=<project-spec3>]
[/ATTRIBUTES=(<attr1>,attr2,…)]
[/TYPE=<type-name>]
[/[NO]REPORT]
[/STATUS=<status>]
[/KEEP_STAGE]
```

Example

```
MWS PROD:WS_001 -
PROD:WS_002
/REPORT
/WORKSET=PROD_X:TEST_WS
```

- `<project-spec1>`

  comprises the specification for project 1:
  `<product-id>:<project-id>`

  This project is one of the inputs to the merge operation, and is also the target project if `/WORKSET` is not specified.

- `<project-spec2>`

  comprises the specification for project 2.

  This project is the second input to the merge operation.

- `/WORKSET=<project-spec3>`

  This optionally specifies the target project, which will be created if it does not already exist.

- `/ATTRIBUTES=(<attr1>,attr2,…)`

  Specifies the user-defined attribute values for the target project.

- `/TYPE=<type-name>`

  Specifies the type of the target project. If this qualifier is not specified, the type name WORKSET is used.

- `/REPORT`

  request that only the Merge Project report be generated. This contains information concerning how the merge is processed. The report is mailed to the user.

- `/STATUS=<status>`

  allows the merged project to be created in either a `LOCKED` or `UNLOCKED` state. The locked state prevents new Dimensions items being added to the project (baselining is likely to occur in this state).

- `/KEEP_STAGE`

avoids resetting the stage of item revisions in the target project to the initial stage, and instead keeps the stage from the source project(s). The default behavior (when this qualifier is not specified) is to reset the stage of all items in the target project to the initial stage. Note that this qualifier can only be used when the target project uses the manual deployment model. This is an optional qualifier.

The MWS command merges the two projects given by `<project-spec1>` and `<project-spec2>`. The merged output is placed in a target project, which is either:

- that given by `<project-spec3>`, or
- that given by `<project-spec1>` (if `<project-spec3>` is not specified).

Constraints

This command can be run only by a user with the appropriate management privileges for the target project concerned.

MWSD – Move Project Directory

NOTE This command is not available in the Issue Management–only licensed version of Dimensions.

```
<directory-path1>
<directory-path2>
[/WORKSET=<project-spec>]
```

Example `MWSD src dst`

- `<directory-pathN>`

  The MWSD command moves the project structure (and items) from the source directory to the destination directory.

  `<directory-path1>` is the source directory.

  `<directory-path2>` is the destination directory.

- `/WORKSET=<project-spec>` comprises:

  `<product-id>:<project-id>`

  This optionally specifies the project to be used for this command: failing this, the user's current project will be taken.

NOTE Whenever a new revision is added to a project, its stage is reset to DEVELOPMENT, and associated deployment areas and reference areas are updated.

Constraints

Normally this command can be run only by a user with the appropriate management privileges for the project concerned.

This constraint can, however, be relaxed using the Set Project Permissions (SWSP) command, as described on page 342.

PA – Populate Areas

NOTE This command is not available in the Issue Management–only licensed version of Dimensions.

```
<project-spec>
[/STAGE=<stage>]
[/AREA=<area-name>]
[/USER_FILENAME=<population-log-file-name>]
```

Example

```
PA DEPLOYMENT:DMNET /STAGE=RELEASE
```

Parameters

- `<project-spec>`

  Specifies the project for which areas are to be populated.

- `<stage>`

  If this parameter is specified, only areas associated with this stage will be populated.

- `<area-name>`

  If this parameter is specified, only this area will be populated. The area must have been previously related to the project with the RAWS command.

- `<population-log-file-name>`

  Specifies the name of the file to contain the log of the area population.

Description

The PA command repopulates online areas associated with a project. If none of the optional qualifiers is specified, all online areas associated with the project are populated.

NOTE The PA command replaces the PBA command.

Constraints

Only users with the "Populate Area from Project" privilege can run this command.

PBA – Populate Build Area

NOTE This command is no longer available; use PA (Populate Area) instead.

PEND – Update Users' Pending Request Lists

Syntax (1st form)
```
PEND CHDOC /PRODUCT=<product-id>
[/CHANGE_DOC_IDS=(<request-id1>,<request-id2>, …)]
```

Syntax (2nd form)
```
PEND CHDOC /PRODUCT=<product-id>
[/STATUS=<status>] [/TYPE=<request-type>]
    [/USER=<user-id>] [/ROLE=<role>]
```

NOTE These two forms of syntax are exclusive-or selections; you cannot mix the various parameters. If the syntax is in the second form, the utility processes every request that meets all the criteria specified. In this form, *wild card % may be used in any of the parameters*; and omission of a parameter is the equivalent of its inclusion with just a value of %.

Example
```
PEND CHDOC /PRODUCT=PAYROLL -/CHANGE_DOC_IDS=(PAYROLL_CR1,PAYROLL_CR_2)

PEND CHDOC /PRODUCT=PAYROLL /STATUS=RAISED /TYPE=CR /USER=FRED -
    /ROLE=DEVELOPER
```

- `/PRODUCT=<product-id>`

  This is a string to be matched by the product-id of each request to be processed. This parameter is not optional, so just specify % if the processing is not to be limited by the value of product-id.

- `/CHANGE_DOC_IDS=(<request-id1>,<request-id2>, …))`

  This is one of a list of request identifiers separated by spaces. Pending lists are recalculated only for these specified requests.

- `/STATUS=<status>`

  This is a string to be matched by the current status (lifecycle state) of each request to be processed.

- `/TYPE=<request-type>`

  This is a string to be matched by the type of each request to be processed.

- `/USER=<user-id>`

  This is a string to be matched by login user names. Each request is not processed unless it is currently in the pending list of at least one matching user name.

- `/ROLE=<role>`

  This is a string to be matched by role-titles of each request to be processed. Each request is not processed unless it is currently in the pending list of at least one user who has been assigned for it a role-title that matches this string.

Description

In the event that a user leaves a project, or change management rules are changed such that the phases are different, or design parts are moved around in the structure that could mean a change to the people responsible for requests, the PEND command allows a change-manager (only) to re-calculate the pending trays for users' requests. It will also recalculate the current phase of the request.

This command also needs to be run whenever rules are enabled (see *Process Modeling User's Guide*) for one or more of the product's request types, if at the time any requests of these types already exist.

IMPORTANT! Whenever this utility is to process more than just a few requests, it is highly recommended that it is executed only at times when database activity is otherwise light.

Constraints

Only users with the appropriate management privileges can run this command.

PEND – Update Users' Pending Item Lists

NOTE This command is not available in the Issue Management–only licensed version of Dimensions.

Syntax

```
PEND ITEM /PRODUCT=<product-id>
[/STATUS=<status>]
[/TYPE=<item-type>]
[/USER=<user-id>]
```

NOTE The utility will process every item that meets all the criteria specified. In this form, wild card % may be used in any of the parameters; and omission of a parameter is the equivalent of its inclusion with just a value of %.

Example

```
PEND ITEM
/PRODUCT=PAYROLL
/STATUS=DEFINED
/TYPE=SRC
/USER=FRED
```

- `/PRODUCT=<product-id>`

  This is a string to be matched by the product-id of each item to be processed. This parameter is not optional, so just specify % if the processing is not to be limited by the value of product-id.

- `/STATUS=<status>`

  This is a string to be matched by the current status (lifecycle state) of each item to be processed.

- `/TYPE=<item-type>`

  This is a string to be matched by the type of each item to be processed.

- `/USER=<user-id>`

  This is a string to be matched by login user names. Each item is not processed unless it is currently in the pending list of at least one matching user name.

Description

In the event that a user leaves a project, or design parts are moved around in the structure that could mean a change to the people responsible for items/files (hereinafter referred to as items for brevity), the PEND command allows a product-manager (only) to re-calculate the pending trays for users' items.

NOTE Whenever this utility is to process more than just a few items, it is highly recommended that it is executed when there is little database activity.

Constraints

Only users with the appropriate management privileges can run this command.

This utility only runs on the currently selected project.

PEND – Update Users' Pending Baseline Lists

NOTE This command is not available in the Issue Management–only licensed version of Dimensions.

Syntax

```
PEND BASELINE /PRODUCT=<product-id>
[/STATUS=<status>]
[/TYPE=<baseline-type>]
[/USER=<user-id>]
```

NOTE The utility will process every baseline that meets all the criteria specified. In this form, wild card % may be used in any of the parameters; and omission of a parameter is the equivalent of its inclusion with just a value of %

Example

```
PEND BASELINE
/PRODUCT=PAYROLL
/STATUS=OPEN
/TYPE=DEVLPMENTSRC
/USER=FRED
```

- `/PRODUCT=<product-id>`

  This is a string to be matched by the product-id of each baseline to be processed. This parameter is not optional, so just specify % if the processing is not to be limited by the value of product-id.

- `/STATUS=<status>`

  This is a string to be matched by the current status (lifecycle state) of each baseline to be processed.

- `/TYPE=<request-type>`

  This is a string to be matched by the type of each baseline to be processed.

- `/USER=<user-id>`

  This is a string to be matched by login user names. Each baseline is not processed unless it is currently in the pending list of at least one matching user name.

Description

In the event that a user leaves a project, or design parts are moved around in the structure that could mean a change to the people responsible for items/files (hereinafter referred to as items for brevity), the PEND command allows a product-manager (only) to re-calculate the pending trays for users' baselines.

NOTE Whenever this utility is to process more than just a few baselines, it is highly recommended that it is executed when there is little database activity.

Constraints

Only users with the appropriate management privileges can run this command.

PEND – Update Users' Pending Project Lists

NOTE This command is not available in the Issue Management–only licensed version of Dimensions.

Syntax

```
PEND WORKSET /PRODUCT=<product-id>
[/STATUS=<status>]
[/TYPE=WORKSET]
[/USER=<user-id>]
```

NOTE The utility will process every project that meets all the criteria specified. In this form, wild card % may be used in any of the parameters; and omission of a parameter is the equivalent of its inclusion with just a value of %.

Example

```
PEND WORKSET
/PRODUCT=QLARIUS
/STATUS=OPEN
/TYPE=WORKSET
/USER=dmsys
```

- `/PRODUCT=<product-id>`

  This is a string to be matched by the product-id of each project to be processed. This parameter is not optional, so just specify % if the processing is not to be limited by the value of product-id.

- `/STATUS=<status>`

  This is a string to be matched by the current status (lifecycle state) of each project to be processed.

- `/TYPE=<item-type>`

  This is a string to be matched by the type of each project to be processed.

- `/USER=<user-id>`

  This is a string to be matched by login user names. Each project is not processed unless it is currently in the pending list of at least one matching user name.

PRIV – Manage Privileges

```
<privilege-id>
/RULE=<rule-id>
/ADD or /DELETE or /REPLACE
[/ROLES=(<list-of-role-names>)]
[/USERS=(<list-of-users-and-groups>)]
[/PRODUCT=<product-id>]
```

Usage

- `PRIV <database-level-privilege-id> /RULE=<rule-id> /USERS=(<list-of-users-and-groups>) {/ADD|/DELETE|/REPLACE}`

  For database-level privileges

- `PRIV <product-level-privilege-id> /PRODUCT=<product-id> /RULE=<product-level-rule-id> {/ADD|/DELETE}`

  For product-level privileges with product-level rules

- `PRIV <product-level-privilege-id> /PRODUCT=<product-id> /RULE=<user-or-group-level-rule-id> /USERS=(<list-of-user-and-group-ids>) {/ADD|/DELETE|/REPLACE}`

  For product-level privileges with user-level or group-level rules

- `PRIV <product-level-privilege-id> /PRODUCT=<product-id> /RULE=<role-level-rule-id> /ROLES=(<list-of-role-names>) {/ADD|/DELETE|/REPLACE}`

  For product-level privileges with role-level rules

Description

Use the PRIV command to enable or disable the rules for privileges and to assign privileges to or deassign privileges from roles, groups, and users.

There are two types of privilege: base database–level privileges (administrator privileges) and product-level privileges.

There are four types of rules: database-level rules, product-level rules (that is, global rules), user-level or group-level rules, and role-level rules.

Constraints

The PRIV command can be run only by users who have the appropriate privilege management capabilities.

QUIT – Quit

Description

Alias for EXIT.

RA – Remove Area

```
<area-name>
```

Example `RA <area-name>`

Parameters

- `<area-name>`

  Specifies the name of the area. Area names must be unique within the base database.

Description

The RA command deletes an area definition. This command does not delete the contents of the area (files or folders) on disk.

Constraints

To delete a work area, you must have the Delete Work Areas privilege. To delete a deployment area, you must have the Delete Deployment Areas privilege.

RAI – Remove Archived Item

NOTE This command is not available in the Issue Management–only licensed version of Dimensions.

```
<item-spec>
[/FILENAME=<file-name>]
[/[NO]CHECK]
```

Example `RAI "PRODX:DECODER.AAAA-SRC;1"`

See the *Administrator's Guide* for details.

RAMA – Remove Archived Material Selected by Archive

NOTE This command is not available in the Issue Management–only licensed version of Dimensions.

```
<archive-id>
[/[NO]CHECK]
```

Example `RAMA ARCH_BL5`

See the *Administrator's Guide* for details.

RAMP – Remove Archived Material Selected by Product

NOTE This command is not available in the Issue Management–only licensed version of Dimensions.

```
<product-id>
[/NOCHECK]
```

Example `RAMP PRODX`

See the *Administrator's Guide* for details.

RAT – Read Archive Tape

NOTE This command is not available in the Issue Management–only licensed version of Dimensions.

```
<archive-id>
/DEVICE=<device-id>
/TAPE=<tape no.>
/VOLUME=<volume-id>
```

Example

```
RAT AA12AB /DEVICE="/dev/rmt0h" -
   /TAPE="aa100"/VOLUME="bb100"
```

See the *Administrator's Guide* for details.

RAWS – Relate Area to Project

NOTE This command is not available in the Issue Management–only licensed version of Dimensions.

```
<area-name>
/WORKSET=<project-spec>
[/RELATIVE_LOCATION=<relative-path>]
[/[NO]POPULATE]
[/[NO]KEEP]
```

Examples

- `RAWS DM10-WIN32 /WORKSET="PVCS:DM10" /RELATIVE_LOCATION="win32"`

  Relates an area to a project. Does not update area *contents*.

- `RAWS DM10-WIN32 /WORKSET="PVCS:DM10" /RELATIVE_LOCATION="windows" /POPULATE`

  Changes the relative location and populates the area. Does not delete existing area files that originated from this project.

- `RAWS DM10-WIN32 /WORKSET="PVCS:DM10" /RELATIVE_LOCATION="windows" /POPULATE /NOKEEP`

  Changes the relative location, deletes existing area files originating from this project, and populates the area with new contents.

Parameters

- `<area-name>`

  The name of the area that you want to relate to the specified project.

- `<project-spec>`

  The project to which you want to relate the specified area.

- `<relative-path>`

  Specifies the relative path within the area's file system directory to which this project's files will be deployed. For example, if `<relative-path>` is `"component1"` and the area's directory is `stal-dev-lx1::/deployment_areas`, a recursive deployment operation would copy all project items into `stal-dev-lx1::/deployment_areas/component1`.

NOTES

– Specifying an empty string clears the relative location.

– If there is no relative location, the area's directory is used for deployment.

– The relative location cannot contain `"..."`.

- `/POPULATE or /NOPOPULATE`

  Specifies whether to populate the area with item revisions from this project (and its child collections) given the specified relative path. By default, the area is not populated (/NOPOPULATE is the default). If /POPULATE is specified, files corresponding to item revisions from the specified project (including item revisions in child collections) are transferred into the area one by one (after any deletions caused by /NOKEEP are performed).

- `/KEEP or /NOKEEP`

  Specifies whether to keep files corresponding to the previously transferred item revisions in the area or delete them if the relative path changes. By default, existing area files are note deleted (/KEEP is the default). If /NOKEEP is specified, files corresponding to item revisions from the specified project (including item revisions in child collections) previously transferred to this area are deleted one by one before any re-population caused by /POPULATE.

Description

Creates or modifies the relationship between a deployment or reference area and the specified project. Deployment areas related to a project will be updated whenever an item revision is deployed to a stage in the context of the specified project. Reference areas related to a project will be updated whenever an item revision is added to, renamed in, removed from, or deleted from the project.

If the relationship specified by an RAWS command already exists, the value of /RELATIVE_LOCATION is used to update the relationship. If both /POPULATE and /NOKEEP are specified, /NOKEEP is processed first.

Constraints

Only users with the "Assign Deployment Areas to Project" privilege can run this command.

RBA – Remove Build Area

NOTE This command is no longer available; use RA (Remove Area) instead.

See the RA command.

RBBL – Relate Baseline to Baseline

NOTE This command is not available in the Issue Management–only licensed version of Dimensions.

```
<child-baseline-spec>
/BASELINE=<parent-baseline-spec>
[/RELATIVE_LOCATION=<relative-path>]
[/DIRECTORY=<baseline-root-spec>]
[/USAGE or /INFO]
```

Example

```
RBBL <child-baseline-spec> /BASELINE=<parent-baseline-spec>
     /DIRECTORY=<baseline-root>
```

Parameters

- `<child-baseline-spec>`

  Specifies the child baseline in the parent-child relationship.

- `<parent-baseline-spec>`

  Specifies the parent baseline in the parent-child relationship.

- `<relative-path>`

  Specifies the relative path of the file system directory to which the child baseline's top-level directory is mapped with respect to the file system directory to which the parent baseline's top-level directory is mapped. For example, if `/RELATIVE_LOCATION` is `"../component1"` and the top-level directory of the parent baseline is mapped to `"/raid1/home/pjwr/work/project/main"`, a recursive `get` operation would map the top-level directory of the child baseline to `"/raid1/home/pjwr/work/project/component1"`.

  Specifying /RELATIVE_LOCATION implies /USAGE.

- `<baseline-root-spec>`

  Specifies the file system directory to which to map the top-level directory of the baseline when the current project is an ancestor of the parent baseline. This value is stored on a per-user basis as part of the relationship.

NOTES

– Specifying an empty string clears the relationship-level baseline root for the user.

– If there is no relationship-level baseline root, /RELATIVE_LOCATION is used.

- `/USAGE`

  Specifies that a usage relationship is to be created.

- `/INFO`

  Specifies that an information relationship is to be created. If /INFO is specified, /RELATIVE_LOCATION cannot be specified.

Description

Creates or modifies a parent-child relationship between the specified baselines. If there is no relative location, the user must specify a value for /DIRECTORY; otherwise, item operations involving the content of the baseline will fail with an error.

If a relationship already exists, the values of /USAGE, /INFO, /RELATIVE_LOCATION, and /DIRECTORY are used to update the relationship. If both /RELATIVE_LOCATION and /DIRECTORY are cleared, a warning is issued.

This command is intended primarily to allow component baselines within a larger release baseline to be updated. The parent baseline must be at the initial lifecycle state for the relationship to be created or for any attribute of the relationship to be changed other than /DIRECTORY.

Constraints

Only users with the appropriate management privileges can run this command.

RBCD – Relate Baselines to Requests

NOTE This command is not available in the Issue Management–only licensed version of Dimensions.

```
<baseline-spec>
/CHANGE_DOC_IDS=(<request1>,<request2>,…)
[/AFFECTED or /IN_RESPONSE_TO or INFO]
```

Example

```
RBCD "PAYROLL:ACME_2.1" -
/CHANGE_DOC=("PAYROLL_TDR_1","PAYROLL_TDR_2")
/INFO
```

- `<baseline-spec>` comprises:

  `<product-id>:<baseline-id>`

- `/CHANGE_DOC_IDS=(<request1>,<request2>,…)`

  `<requestN>` identifies a request to which the specified baseline is to be related.

- `/AFFECTED or /IN_RESPONSE_TO or /INFO`

  Specifies the type of relation to be set up between the given baseline and the associated requests. The qualifiers are mutually exclusive.

  The default is `/AFFECTED`.

Constraints

RBCD is restricted to merge, release, and revised baselines. If it is run with respect to an archive or design baseline, then an appropriate error will be returned.

RBCD will only work successfully if you have both the baseline and requests in your pending list. If you specify an `/INFO` relationship, however, then this pending list restriction is relaxed.

There is no support for phase rules or change management rule enhancements within the context of baseline to request relationships.

Only the three relationship types – Info, Affected and in-Response-to – are supported. There is no support for user-defined relationship types.

RBPROJ – Delete a Dimensions Build Project

NOTE This command is no longer available. Use Dimensions Build to delete a build project.

```
Command no longer available.
```

RBWS – Relate Baseline to Project

NOTE This command is not available in the Issue Management–only licensed version of Dimensions.

```
<baseline-spec>
/WORKSET=<project-spec>
[/RELATIVE_LOCATION=<relative-path]
```

Example `RBWS <child-baseline-spec> /WORKSET=<parent-project-spec>`

Parameters

- `<baseline-spec>`

  Specifies the child baseline in the parent-child relationship.

- `<project-spec>`

  Specifies the parent project in the parent-child relationship.

- `<relative-path>`

  Specifies the relative path of the file system directory to which the child baseline's top-level directory is mapped with respect to the file system directory to which the parent project's top-level directory is mapped.

NOTES

– Specifying an empty string clears the relationship-level baseline root for the user.

– If there is no relationship-level baseline root, /RELATIVE_LOCATION is used.

– If the path contains `::` (that is, it has the format `<node-or-area>::<path>`, and `<node-or-area>` matches the name of an existing area), the path is interpreted as an offset (relative path) with regard to the area root directory. (This offset may be empty, in which case the file system directory is set to equal the area directory.) Otherwise, standard Dimensions interpretation of the path applies.

Description

Creates or modifies a parent-child relationship between the specified child baseline and the specified parent project.

If a relationship already exists, the value of /RELATIVE_LOCATION is used to update the relationship.

This command is intended primarily to allow component baselines within a larger release baseline to be updated. The parent baseline must be at the initial lifecycle state for the relationship to be created or for any attribute of the relationship to be changed.

Constraints

Only users with the appropriate management privileges can run this command.

RCCD – Relate Requests to Request

```
<request-id>
/CHANGE_DOC_IDS=(<request1>,<request2>,…)
[/RELATIONSHIP=<relname> or /DEPENDENT or /INFO]
```

Example `RCCD PROD_CN_4 /CHANGE=PROD_DR_25`

- `<request-id>`

  This is the identity of the request to be the parent in the relationship to be created.

- `/CHANGE_DOC_IDS=(<request1>,<request2>,…)`

  Specifies the identities of one or more requests to be children in the relationship to be created.

- `/RELATIONSHIP=<relname>`

  Specifies the relationship class as `DEPENDENT` or `INFO`, or as the name of one of the subclasses of relationship defined as equivalent to either of these.

  The default is `/RELATIONSHIP=DEPENDENT`.

- `/DEPENDENT` or `/INFO`

  This is equivalent to specifying either of these as settings of the `/RELATIONSHIP` qualifier.

Constraints

This command can be run by users who have a role (any role will suffice) on the product or products owning the specific request concerned. Such users must have the parent request in their Pending List.

However, the user with the role of `PRODUCT-MANAGER` can setup the Process Model to specify that no request relationships can be created for certain request types.

RCDI – Return Request ID

```
<request-id>
[/[NO]CANCEL_TRAVERSE]
[/COMMENT=<comment>]
[DIRECTORY=<project-directory-filter>]
[/[NO]RECURSIVE]
[/[NO]FORCE]
[/[NO]KEEP]
[/LOGFILE=<log-file>]
[/USER_DIRECTORY=<target-directory>
[/WORKSET=<project>]
[/CONTENT_ENCODING=<file-encoding>]
[/NOMETADATA]
```

Example `RCDI PAYROLL_CR1`

Parameters

- `<request-id>`

  The name of a Dimensions request.

- `/CANCEL_TRAVERSE`

  By default, all requests related as dependent to the specified request are processed by this command. This qualifier forces the command to process only the request specified.

- `/COMMENT=<comment>`

  Specifies a comment associated with this return.

- `/DIRECTORY`

  Enables you to specify a project directory filter to restrict the number of items processed.

- `/RECURSIVE`

  Used with /DIRECTORY, this specifies that the filter is to be processed recursively; that is, subdirectories are to be processed.

- `/FORCE`

  Forces an item update.

- `/KEEP`

  Specifies that files are not to be deleted when they are checked in.

- `/LOGFILE`

  Specifies a local log file to which the command is to divert all messages.

- `/USER_DIRECTORY`

  Specifies the target directory to which files are to be returned.

- `/WORKSET`

  Specifies the project to be processed by this command.

- /CONTENT_ENCODING

  Specifies the content encoding for new item revisions to be created. Supported encodings are the Microsoft codepages, the ISO-8859 variants (1–10), UTF-8, UTF-16, UTF-16BE, UTF-16LE, UTF32, UTF32BE, and UTF32LE.

- /NOMETADATA

  This parameter disables creation and usage of metadata files in the local work area.

Description

This command returns all the items that are checked out against a specified request.

When multiple revisions of the same item are related to requests processed by this command, only the latest is processed.

RCDWS – Remove Request Items from Project

```
<request-id>
[/[NO]CANCEL_TRAVERSE]
[DIRECTORY=<project-directory-filter>]
[/[NO]RECURSIVE]
[/LOGFILE=<log-file>]
[/WORKSET=<project>]
```

Example `RCDWS PAYROLL_CR1`

Parameters

- `<request-id>`

  The name of a Dimensions request.

- `/CANCEL_TRAVERSE`

  By default, all requests related as dependent to the specified request are processed by this command. This qualifier forces the command to process only the request specified.

- `/DIRECTORY`

  Enables you to specify a project directory filter to restrict the number of items processed.

- `/RECURSIVE`

  Used with /DIRECTORY, this specifies that the filter is to be processed recursively; that is, subdirectories are to be processed.

- `/LOGFILE`

  Specifies a local log file to which the command is to divert all messages.

- `/WORKSET`

  Specifies the project to be processed by this command.

Description

This command removes from the current project (or a specified project) all the items that are related to a specified request.

When multiple revisions of the same item are related to requests processed by this command, only the latest is processed.

RCI – Report Current Items

NOTE This command is not available in the Issue Management–only licensed version of Dimensions.

```
<file-name>
[/NEW or /OLD]
[/PART=<part-spec> or /BASELINE=<baseline-spec>]
[/SORT=<sort>]
[/WORKSET=<project-spec>]
```

NOTE RCI cannot be run from Dimensions for z/OS.

Example

```
RCI sunrm3_01.export
/NEW -
/PART="PROD:RELEASE MANAGEMENT.AAAA"
```

If /NEW *is specified*

- <file-name>

  Specifies the name of the export file where the exported structure is to be stored. If specified as * (asterisk), then a temporary export file is created which is deleted at the end of the operation.

  If no export file is specified, the file name defaults to item_list.out.

Either <part-spec> or <baseline-spec> *must* be specified:

- /PART=<part-spec>

  Specifies the top design part of the current product structure which is to be included in the export file.

  It comprises: <product-id>:<part-id>.<variant>;<pcs>

  <variant> **must** be specified, even if only one variant exists.

  <pcs> is ignored; the current PCS is always used.

- /BASELINE=<baseline-spec>

  Specifies the baseline on which the structure to be included in the export file is to be based.

  It comprises: <product-id>:<baseline-id>

NOTE The above descriptions of <file-name>, **<part-spec>,** and <baseline-spec>**,** with the /NEW option, are identical for RCI, RCP and RDS commands.

- /SORT=<sort>

  Indicates the report sequence. It should be omitted, as currently the only valid option is IID to list current items in item-id order.

- /WORKSET=<project-spec> comprises:

  <product-id>:<project-id>

  This optionally specifies the project to be used for this command: failing this, the user's current project will be taken.

If /NEW *is omitted*

- <file-name>

  Specifies where an exported structure has previously been stored.

- /OLD

  may be specified, but is optional as this is the default when /NEW is omitted.

- /PART=<part-spec>

  This is normally omitted. If specified, the report is restricted to the items in <part-spec> and any design parts below it in the tree structure, which are also within the exported structure.

- /BASELINE=<baseline-spec>

  This is normally omitted. If specified, the report is restricted to those items which are both within the scope of <baseline-spec> and within the exported structure.

- /SORT=<sort>

  should be omitted (as above for the /NEW option).

- /WORKSET=<project-spec> comprises:

  <product-id>:<project-id>

  This qualifier is only meaningful if /NEW is specified.

Constraints

This command can be run only by the user initiating the report who must have a valid role for the top design part in the structure to be reported. In a structure report which includes items, if an item has two or more revisions currently in the same lifecycle state, only the latest (most recently created/updated) of these is shown.

RCP – Report Current Parts

```
<file-name>
[/NEW or /OLD]
[/PART=<part-spec> or /BASELINE=<baseline-spec>]
[/SORT=<sort>]
```

NOTE RCP cannot be run from Dimensions for z/OS.

Example `RCP sunrm3_01.export /OLD`

If `/NEW` *is specified*

- `<file-name>`

  Specifies the name of the file where the exported structure is to be stored. If specified as * (asterisk), then a temporary file is created which is deleted at the end of the operation.

  If no file name is specified, `part_list.out` is created by default.

Either `<part-spec>` or `<baseline-spec>` *must* be specified:

- `/PART=<part-spec>`

  Specifies the top design part of the current product structure which is to be included in the export file.

  It comprises: `<product-id>:<part-id>.<variant>;<pcs>`

  `<variant>` **must** be specified, even if only one variant exists.

  `<pcs>` is ignored; the current PCS is always used.

- `/BASELINE=<baseline-spec>`

  Specifies the baseline on which the structure to be included in the export file is to be based.

  It comprises: `<product-id>:<baseline-id>`

NOTE The above descriptions of `<file-name>`, **`<part-spec>`,** and `<baseline-spec>`**,** with the `/NEW` option, are identical for RCI, RCP and RDS commands.

- `/SORT=<sort>`

  Indicates the report sequence. Valid options are:

  - PNO to list current design parts in part number order; or
  - PID to list current design parts in part-id order.

If omitted, it defaults to PID.

If /NEW *is omitted*

- <file-name>

  Specifies where an exported structure has previously been stored.

- /OLD

  may be specified, but is optional as this is the default when /NEW is omitted.

- /PART=<part-spec>

  This is normally omitted. If specified, the report is restricted to those design part which are both within the tree structure specified by <part-spec> and within the exported structure.

- /BASELINE=<baseline-spec>

  This is normally omitted. If specified, the report is restricted to those design parts which are both within the scope of <baseline-spec> and within the exported structure.

- /SORT=<sort>

  Indicates the report sequence (as above for the /NEW option).

Constraints

This command can be run only by the user initiating the report who must have a valid role for the top design part in the structure to be reported.

RCU – Remove Customer

NOTE This command is not available in the Issue Management–only licensed version of Dimensions.

```
<name>
/LOCATION=<location>
/PROJECT=<project-spec>
```

Example

```
RCU "Brown Finances"
/LOCATION="Bristol" -
/PROJECT="PAYROLL"
```

- `<name>`

  Specifies the customer's name.

- `/LOCATION=<location>`

  Specifies the customer's physical location.

- `/PROJECT=<project-spec>`

  Specifies the project name.

Description

The Dimensions product allows you to maintain a list of customers and a record of which Dimensions releases have been sent to each customer.

The RCU command enables you to remove a customer's details.

Constraints

The combination of customer name, location, and project-spec must be unique in the Dimensions database.

You cannot remove a customer to whom releases have been forwarded.

RDEL – Delete a Job from the Job Queue

`<job-key>`

string (<id>-<job-uid>) that identifies the job to be deleted from the job queue.

Example `RDEL B-123456`

Description

RDEL enables you to delete a job from the job queue.

RDS – Report Design Structure

```
<file-name>
[/NEW or /OLD]
[/PART=<part-spec> or /BASELINE=<baseline-spec>]
[/SORT=<sort>]
[/LEVEL=<level>]
[/STRUCTURE=(<option>,…)]
[/WORKSET=<project-spec>]
```

NOTE RDS cannot be run from Dimensions for z/OS.

Example `RDS sunrm3_01.export /STRUCTURE=ALL`

If `/NEW` *is specified*

- `<file-name>`

  Specifies the name of the file where the exported structure is to be stored. If specified as ***** (asterisk), then a temporary file is created which is deleted at the end of the operation.

  If no file name is specified, `design_structure.out` is created by default.

Either `<part-spec>` or `<baseline-spec>` *must* be specified:

- `/PART=<part-spec>`

  Specifies the top design part of the current product structure which is to be included in the export file.

  It comprises: `<product-id>:<part-id>.<variant>;<pcs>`

  `<variant>` **must** be specified, even if only one variant exists.

  `<pcs>` is ignored; the current PCS is always used.

- `/BASELINE=<baseline-spec>`

  Specifies the baseline on which the structure to be included in the export file is to be based.

  It comprises: `<product-id>:<baseline-id>`

NOTE The above descriptions of `<file-name>`, **`<part-spec>`,** and `<baseline-spec>`**,** with the `/NEW` option, are identical for RCI, RCP and RDS commands.

- /SORT=<sort>

  Indicates the report sequence. Valid options are:

  - PNO to list current design parts in part number order at each structural level.
  - PID to list current design parts in part-id order at each structural level.

  If omitted, it defaults to PID.

- /LEVEL=<level>

  Specifies the number of levels of the structure which are to be reported, working downwards either from <part-spec> if specified, or otherwise from the highest-level design part in the tree structure which is being reported on.

  All levels of that structure are included if this parameter is omitted.

- /STRUCTURE=<option>

  Specifies what contents of the design part structure are to be included in the report:

  - USAGE to include USED design parts.
  - ITEM to include items.
  - include requests.
  - ROLE to include user roles.
  - ALL to include all options.

  By default, i.e. if /STRUCTURE is not specified, only the structure of BREAKDOWN design parts is reported.

- /WORKSET=<project-spec> comprises:

  <product-id>:<project-id>

  This optionally specifies the project to be used for this command: failing this, the user's current project will be taken.

If /NEW is omitted

- <file-name>

  Specifies where an exported structure has previously been stored.

- /OLD

  may be specified, but is optional as this is the default when /NEW is omitted.

- /PART=<part-spec>

  This is normally omitted. If specified, the report is restricted to those design parts (and items if specified - see <option>) which are both within the tree structure specified by <part-spec> and within the exported structure.

- /BASELINE=<baseline-spec>

  This is normally omitted. If specified, the report is restricted to those design parts (and items if specified - see <option>) which are both within the scope of <baseline-spec> and within the exported structure.

- /SORT=<sort>

  as above for the /NEW option.

- /LEVEL=<level>

  as above for the /NEW option.

- /STRUCTURE=<option>

  as above for the /NEW option.

- /WORKSET=<project-spec>

  This qualifier is only meaningful if /NEW is specified.

Constraints

This command can be run only by the user initiating the report who must have a valid role for the top design part in the structure to be reported.

REGDEPLOY – Register Serena Automated Deployment Details with Dimensions

NOTE This command is not available in the Issue Management–only licensed version of Dimensions.

```
/DEPLOY_URL=<deploy_url> | /DELETE | /SHOW
```

Examples

1 This command registers deployment details with the Dimensions product. (Typically, this command is directly invoked by the deployment engine to register its location.)
```
REGDEPLOY /DEPLOY_URL="http://mover20server:80/servlet"
```

2 This command deletes (uregisters) the details of a current deployment with the Dimensions product.
```
REGDEPLOY /DELETE
```

3 This command shows the details of a current deployment with the Dimensions product.
```
REGDEPLOY /SHOW
```

For details, refer to the associated user's guide that accompanies this optional software.

REL – Release

NOTE This command is not available in the Issue Management–only licensed version of Dimensions.

```
<release-spec>
/BASELINE=<baseline-spec>
[/TEMPLATE_ID=<template-id>]
[/DESCRIPTION=<description>]
[/DIRECTORY=<directory>]
[/FILENAME=<file-name>]
[/[NO]DELTA /PREV_RELEASE=<release-id>]
[/[NO]EXPAND]
[/[NO]TOUCH]
[/[NO]OVERWRITE]
[/NOMETADATA]
```

NOTE When a release is created, Dimensions records the target directory and the person who was responsible for creating the release. You can obtain this information through the published view 'PCMS_RELEASE_DATA' (for details of this view, see the *Reports Guide*).

Example

```
REL PROD:"R M 2.0 FOR HP" -
/BASE=PROD:"R M VERSION 2 FOR HP"
```

- `<release-spec>`

  Comprises:

  `<product-id>:<release-id>`

- `/BASELINE=<baseline-spec>`

  Comprises:

  `<product-id>:<baseline-id>`

  `<product-id>` *must be the same as that for the* `<release-spec>`*.*

- `/TEMPLATE=<template-id>`

  Specifies a release template to be used for this release.

- `/DESCRIPTION=<description>`

  This is a description of the release.

- `/DIRECTORY=<directory>`

  Specifies the top level directory where the release is to be stored.

  If omitted, it defaults to the current directory.

NOTE On a UNIX installation where the Dimensions System Administrator user-id is not the default `dmsys`, you will need to include the node name in the directory specification, for example, `/DIRECTORY="hp6::/tmp/project/rel1"`

- `/FILENAME=<file-name>`

Generates command script to <file-name> but does not run the command

- /DELTA

  Specifies a delta release is to be created; that is, items that are identical to the /PREV_RELEASE are not exported.

NOTE /PREV_RELEASE comprises <release-id> **not** <release-spec>. For example, if you wish to release PROD:REL_2 based on a previous release of PROD:REL_1, then the /DELTA part of the REL syntax would be:
REL "PROD:REL_2"… /DELTA /PREV_RELEASE="REL_1"
You **do not** specify PROD for /PREV_RELEASE.

- /[NO]EXPAND
  /[NO]TOUCH
  /[NO]OVERWRITE

  Adds the specified qualifier to each FI command in the script generated by REL.

- /NOMETADATA

  This parameter disables creation and usage of metadata files in the local work area.

Notes

REL processes the content of child baselines when the relationship specifies a relative location. Each child baseline for which no relative location exists receives a diagnostic indicating that the child baseline was not processed and why.

Constraints

This command can be run by users who can 'get the items' that compose the release. In practice this usually means being assigned one or more roles whose scope is at least either for the top design part in the baseline used, extending to all of the design structure segment below that design part, or for the project specified when that Baseline was created, or for both.

If you are going to be creating several releases of (varying aspects or segments of) a particular product, it will be simpler if you are assigned a role for the product-level design part, so that it extends to every design part, and therefore to every item, in the product. (Even so, you might sometimes need some permissions additional to this, whenever the baseline used, and thus the release made from it, contains some foreign Items – i.e. Items that do not belong to the baselined / released product.)

The baseline used must be of release mode (i.e. it cannot contain more than one revision of any one Item). This means that revised baselines and merged baselines also qualify as release mode, as well as those of release mode created using a baseline template; but that baselines of design and archive modes do not qualify.

If a release is created using a release template, that template then cannot be altered (until all releases that used that template have been deleted). This is to ensure that each new release operation is repeatable. (Another release template based on this one can be created and then altered, if such a template is needed for some later release.)

Although this operation will place files in the release directory, automatically creating sub-directories if and as required, it will do so only if the operating system where the release

directory is located would have permitted you to create such files and subdirectories yourself.

RENAME – Rename an Existing Product, Baseline, Design Part, Project, or Items

```
[/PRODUCT=<old-product-spec>] /NEW_ID=<new-product-id>
or
[/BASELINE=<old-baseline-spec>] /NEW_ID=<new-baseline-id>
or
[/PART=<old-part-spec>] /NEW_ID=<new-part-id>
or
[/WORKSET=<old-project-spec>] /NEW_ID=<new-project-id>
or
[/ITEM_ID=<old-item-spec>] /NEW_ID=<new-item-id>
```

Examples

```
RENAME
/BASELINE=MINAKO:RELEASE_1.0
/NEW=RELEASE_DUFF
```

```
RENAME
/WORKSET=MINAKO:DEVELOPMENT_WORK
/NEW=SOURCE_CODE
```

- `<old-product-spec>`

  Specifies the old product name

- `<new-product-id>`

  Specifies the new product name.

- `<old-baseline-spec>`

  Specifies the old, full baseline specification.

- `<new-baseline-id>`

  Specifies the new baseline-id. This is the name of the baseline only, not the full specification.

- `<old-part-spec>`

  Specifies the old, full part specification.

- `<new-part-id>`

  Specifies the new part-id. This is the name of the part only, not the full specification.

- `<old-project-spec>`

  Specifies the old, full project specification.

- `<new-project-id>`

  Specifies the new project-id. This is the name of the project only, not the full specification.

- `<old-item-spec>`

  Specifies the old, full item specification.

- `<new-project-id>`

  Specifies the new item-id. This is the name of the item only, not the full specification.

Description

The "product" version of this command enables an existing product to be renamed.

The "baseline" version of this command enables an existing baseline to be renamed within the context of the same product.

The "design part" version of this command enables an existing design part to be renamed within the context of the same product.

The "project" version of this command enables an existing project to be renamed within the context of the same product.

The "item" version of this command enables an existing item to be renamed within the context of the same product.

Constraints

The "product" version of this command can be run only by a user with the appropriate management privileges on the project.

The "baseline" version of this command can be run only by a user with the appropriate management privileges on the baseline.

The "design part" version of this command can be run only by a user with the appropriate management privileges on the design part.

The "project" version of this command can only be run by a user with the appropriate management privileges on the project or the owner of the project (the user who created the project).

REQC – Request Dimensions Request

```
<request-id>
[/CANCEL]
```

Examples

```
REQC "PAYROLL_TDR_1"
REQC "PAYROLL_TDR_1" /CANCEL
```

- `<request-id>`

  Identifies the request.

- `/CANCEL`

  cancel an existing issue replication "request" for a request. This is only valid if executed by the user who made the original request or by a user with the role of CHANGE-MANAGER or PRODUCT-MANAGER.

Description

Upon running the REQC command, the request for the request is logged locally until the next scheduled replication occurs. When the replication actually happens, all the requests for requests are processed by the site that currently owns them and they are reassigned to those sites that requested them. If multiple sites have requested the same request, then this reassignment is done on a first-come-first-served basis, with the other requests being returned with an appropriate error message.

When the request has been processed as a result of replication, the result of the request – whether it was accepted or rejected – will automatically be e-mailed to the request requester. This e-mail will notify the user whether or not they can now work on that request.

Constraints

1 To successfully run REQC you must either:

- have the requested request in your pending list and have valid privileges for the work that you intend to undertake, or
- be a user with the appropriate management privileges.

2 A request will only succeed if the request in question is not currently being worked on. This means that if any items are checked out against that request then the request will be denied. This will also be the case for any requests that have been locked using the /EXCLUSIVE_LOCK qualifier of the CC or UC commands, see pages 79and 353 respectively.

REXEC – Execute a Job on a Network Node

```
/NETWORK_NODE=<nodename>
/TEMPLATE_ID=<template-name>
[/[NO]BATCH]
[/[NO]CAPTURE]
[/USER=<userid>]
[/PASSWORD=<password>]
[/PARAMETERS=<name1=value1,name2=value2,…,nameN=valueN>]
[/DESCRIPTION=<text>]
[/EXECUTION_DIRECTORY=<directory>]
```

- `/NETWORK_NODE=<nodename>`

  Specifies the Dimensions network node on which to run the task.

NOTE If an AUTH command was previously issued against the node specified by /NETWORK_NODE, then REXEC will reuse credentials specified by that AUTH command.

- `/TEMPLATE_ID=<template-name>`

  Specifies the template name to be used. This maps to the template file located in the tertiary node directory specified by the value of the `DM_TEMPLATE_CATALOG<n>` symbols. Normally, these symbols are defined in the `dm.cfg` file on the network node; however, if they are not defined there, `%DM_ROOT%templates\<template-name>` is used on Windows, and `$DM_ROOT/templates/<template-name>` is used on UNIX and z/OS.

- `[/BATCH]`

  Specifies asynchronous execution.

- `[/CAPTURE]`

  if specified, a one-time certificate will be provided in the symbol table.

- `[/PARAMETERS=<name1=value1,name2=value2,…, nameN=valueN>]`

  list of keyword and values (possibly arrays of values) to be passed into the template being executed. These keyword/values have to be comma separated.

- `[/DESCRIPTION]`

  text that is used to update the job description text.

- `[/USER]`

  Specifies the user-id to be used for execution at the network node.

- `[/PASSWORD]`

  Specifies the password to be used for execution at the network node.

- `[/EXECUTION_DIRECTORY=<directory>]`

  Specifies the directory in which the remote command is to be started.

Description

REXEC enables you to execute a job on a tertiary node and records the job in the job queue.

On output, REXEC will print out the job key of the following form:

B-<job-uid> for build jobs

R-<job-uid> for other jobs

If /BATCH is specified, then the job will be executed asynchronously. Such jobs will have their status set to SUBMITTED in the job queue. The RSTAT command can be used to update the status of a batch job.

If /BATCH is not specified, then the job will be executed synchronously. After execution, the job status will be set to FAILED or SUCCEEDED depending on whether or the not the job executed successfully.

NOTE Synchronous builds are not supported on a mainframe node.

For addition information concerning remote job entry, see the *Administrator's Guide*.

RI – Return (Check In) Item

NOTE This command is not available in the Issue Management–only licensed version of Dimensions.

```
<item-spec>
[/ROOT_PROJECT=<project-spec>]
[/FILENAME=<file-name>]
[/USER_FILENAME=<user-filename>]
[/[NO]KEEP]
[/STATUS=<status>]
[/COMMENT=<comment text>]
[/ATTRIBUTES=(<attr1>=<value1>,<attr2>=<value2>,…)]
[/WORKSET=<project-spec>]
[/[NO]FORCE_UPDATE]
[/[NO]CANCEL_UNCHANGED]
[/CODEPAGE=<code-page>|DEFAULT]
[/CONTENT_ENCODING=<file-encoding>]
```

Example

```
RI PROD:"QUERY RELEASE".AAAA-SRC;1 -
/KEEP /STATUS="UNDER TEST"" -
/COMMENT="checked in for CRB 91"
```

- `<item-spec>`

  Comprises:

  ```
  <product-id>:<item-id>.<variant>–<item-type>;<revision>
  ```

  `<item-id>` may be omitted if `<file-name>` is specified.

  `<variant>` may be omitted if only one exists.

  `<revision>` may be omitted if you have checked out only one revision of this item.

- `/ROOT_PROJECT=<project-spec>`

  Comprises:

  `<product-id>:<project-id>`

  This optionally specifies the root project. Use this when the current project set via SCWS (or the project specified by the /WORKSET qualifier) occurs in more than one project tree.

- `/FILENAME=<file-name>`

  Specifies the name of the project file name. If /ROOT_PROJECT is used to specify a the root project, /FILENAME is interpreted in the scope of that project.

  The project file name identifies the relative path (directory plus file name) from the working location to the item to be used from the current project. The project file name for the same item may *differ* between projects; for example, `src/hello.c`, `hello.c`, or `src/build/hello.c`.

  It may be omitted if `<item-id>` is specified.

- /USER_FILENME=<user-filename>

  Specifies the file in the user-area from which the item will be copied.

  It may be omitted if the file has the same name as had been given to the user-area file when this item was checked out (EI command).

- /KEEP

  Specifies that the user area file, which is normally deleted after its data has been placed under Dimensions control, is to be left intact.

  If the command is successful and /KEEP is specified, local metadata is updated; the new revision number is recorded and marked as no longer checked out. /NOKEEP causes the local metadata to be deleted as well as the file.

- /STATUS=<status>

  Specifies a valid changed status (lifecycle state) for the checked in revision.

NOTE The only equivalent to this parameter in GUI mode is RI followed by AI. The status, if specified, must be one which would be valid if AI had been used separately. If omitted, the revision remains at the initial state (in the lifecycle defined for <item-type>).

- /COMMENT=<comment text>

  comment text to explain the reason for the check in of this item revision. The comment text can be up to 1978 characters long, and can be made available within the item header.

- /ATTRIBUTES=(<attr1>=<value1>,<attr2>=<value2>,…)

  <attrN> is the Variable Name defined for one of the user-defined attributes for items, which has also been declared usable for the <product-id> and <item-type> specified in <item-spec>.

  <valueN> is the substitution value to be given to this attribute.

- /WORKSET=<project-spec> comprises:

  <product-id>:<project-id>

  and is optional (subject to the following). A checked out item can **only** be checked in to the specific project from which it was originally checked out (an error will be generated if you try to check it in to another project). Therefore, if your current project is not that project, either this qualifier must be used to specify the correct check in <project-spec> or your current project must be set to that <project-spec> using the Set Current Project (SCWS) command.

- /FORCE_UPDATE

  If the checksum is enabled for the item type and the file checked in has not been modified, the check in will succeed only if this qualifier is used, otherwise it will fail.

- /CANCEL_UNCHANGED or /NOCANCEL_UNCHANGED

  /CANCEL_UNCHANGED performs a CIU (Cancel Item Update) command if the user file does not differ from the base revision and Dimensions is configured to allow updates only if a real change is made.

- /CODEPAGE=<code-page>|DEFAULT

Specifies the *code page* to be associated with the item. The code page defines the method of encoding characters. It encompasses both the different ways characters are encoded on different platforms (EBCDIC on z/OS and ASCII on Windows and UNIX) and differences between human languages. Every item in Dimensions has a code page associated with it, this being defined or derived for the connection setting or an individual item.

The /CODEPAGE parameter defaults to the code page specified when the connection between the database server and the logical node on which the user file resides was created. Whenever the item moves between platforms, for example, on a check-out from the mainframe to a PC, if the code page for the target platform is different to the item's code page, Dimensions automatically converts the item.
/CODEPAGE is relevant only for text files, because whenever a text file is checked out or gotten, it must be in the right code page for the target platform, so that it displays correctly. Binary files are moved between platforms with no conversion.

For further details concerning code pages and logical nodes see the *Distributed Development Guide*.

You are advised to let the parameter default to the code page for the item type or the platform.

The /CODEPAGE options available are:

<code-page>	Specify one of the code page values listed in the text file codepage.txt, located on your Dimensions server in the codepage subdirectory of the Dimensions installation directory. This file also provides more information about translation between code pages.
DEFAULT	Use the code page specified for the target node connection.

- <file-encoding>

  Specifies the content encoding for new item revisions to be created. Supported encodings are the Microsoft codepages, the ISO-8859 variants (1–10), UTF-8, UTF-16, UTF-16BE, UTF-16LE, UTF32, UTF32BE, and UTF32LE.

NOTE RI results in the creation of a new revision in the project. If DEVELOPMENT deployment areas are in use, this command automatically updates the corresponding areas.

Constraints

This command can be run only by the user who previously checked out the item revision or by a user with the appropriate management privileges.

IMPORTANT! The maximum size of any item in the database is 2GB.

RICD – Relate Item to Requests

NOTE This command is not available in the Issue Management–only licensed version of Dimensions.

```
<item-spec>
[/FILENAME=<file-name>]
/CHANGE_DOC_IDS=(<request1>,<request2>,…)
[  /AFFECTED or /IN_RESPONSE_TO or /INFO  ]
[/WORKSET=<project-spec>]
```

Example `RICD PROD:"QUERY RELEASE".AAAA-SRC;1 /CHANGE_DOC=PROD_DR_25`

- `<item-spec>`

  Comprises:

  `<product-id>:<item-id>.<variant>–<item-type>;<revision>`

`<item-id>`	may be omitted if `<file-name>` is specified.
`<variant>`	may be omitted if only one exists.
`<revision>`	defaults to the latest revision (see Introduction on page 16).

- `/FILENAME=<file-name>`

  Specifies the name of the project file name.

  The project file name identifies the relative path (directory plus file name) from the working location to the item to be used from the current project. The project file name for the same item may *differ* between projects; for example, `src/hello.c`, `hello.c`, or `src/build/hello.c`.

  It may be omitted if `<item-id>` is specified.

- `/CHANGE_DOC_IDS=(<request1>,<request2>,…)`

  `<requestN>` identifies a request to which the specified item is to be related.

- `/AFFECTED or /IN_RESPONSE_TO or /INFO`

  Specifies the type of relation to be set up between the given item and the requests. The qualifiers are mutually exclusive.

  The default is `/AFFECTED`.

- `/WORKSET=<project-spec>` comprises:

  `<product-id>:<project-id>`

  This optionally specifies the project to be used for this command: failing this, the user's current project will be taken.

  Item revisions to be affected by the command may be specified explicitly, or they will be selected from the project.

Constraints

This command can be run only by a user who has the request in his/her pending list or who has the appropriate management privileges.

RII – Relate Item to Item

NOTE This command is not available in the Issue Management–only licensed version of Dimensions.

```
<src-item-spec>
<dst-item-spec>
/RELATIONSHIP=<relationship-id>
[/FILENAME=<src-filename>]
[/FILENAME=<dst-filename>]
[/COMMENT=<comment-text>]
[/WORKSET=<project-spec>]
```

Example

```
RII "PROD_X:INTERFACE_C.AAAA-C;1" -
"PROD_X:INTERFACE_H.AAAA-H;1" /RELATIONSHIP="INCLUDES" –
/COMMENT="INCLUDED HEADER"
```

- `<src-item-spec>`

  Specifies the source item from which the relationship instance will start.

- `<dst-item-spec>`

  Specifies the destination item to which the relationship will be made.

- `/RELATIONSHIP=<relationship-id>`

  Specifies the relationship type as defined by the DIR command.

- `/FILENAME=<src-filename>`
  `/FILENAME=<dst-filename>`

  Specify the names of the project file names.

  The project file name identifies the relative path (directory plus file name) from the working location to the item to be used from the current project. The project file name for the same item may *differ* between projects; for example, `src/hello.c`, `hello.c`, or `src/build/hello.c`.

  It may be omitted if `<item-id>` is specified.

- `/COMMENT=<comment-text>`

  This is an optional qualifier and specifies a comment to be attached to the relationship instance.

- `/WORKSET=<project-spec>` comprises:

  `<product-id>:<project-id>`

  this optionally specifies the project to be used for this command: failing this, the user's current project will be taken.

  Item revisions to be affected by the command may be specified explicitly, or they will be selected from the project.

Description

The RII command is used to create instances of relationships defined by the DIR command. The source and destination item types must be consistent with the relationship definition.

(The XII command (on page 403) is used to remove such relationships.)

Constraints

This command can be run only by a user who has the items in their Pending list or by a user with the appropriate management privileges.

RIP – Relate Item to Part

NOTE This command is not available in the Issue Management–only licensed version of Dimensions.

```
<item-spec>
[/FILENAME=<file-name>]
/PART=<part-spec>
[/WORKSET=<project-spec>]
```

Example `RIP PROD:"QUERY RELEASE".AAAA-SRC /PART=PROD:"RELEASE MANAGEMENT".IBM`

- `<item-spec>` comprises:

  `<product-id>:<item-id>.<variant>-<item-type>;<revision>`

  `<item-id>`

  `<variant>` may be omitted if only one exists.

  `<revision>` is ignored; all revisions are related to the specified design part.

- `/FILENAME=<file-name>`

  Optionally specifies the name of the project file name.

  The project file name identifies the relative path (directory plus file name) from the working location to the item to be used from the current project. The project file name for the same item may *differ* between projects; for example, `src/hello.c`, `hello.c`, or `src/build/hello.c`.

- `/PART=<part-spec>` comprises:

  `<product-id>:<part-id>.<variant>;<pcs>`

  `<variant>` may be omitted if only one exists.

- `/WORKSET=<project-spec>` comprises:

  `<product-id>:<project-id>`

  This optionally specifies the project to be used for this command: failing this, the user's current project will be taken.

  Item revisions to be affected by the command may be specified explicitly, or they will be selected from the project.

Constraints

This command can be run only by a user who has the item revision in their pending list or by a user with the appropriate management privileges.

RIR – Remove Item Relation Definition

NOTE This command is not available in the Issue Management–only licensed version of Dimensions.

```
<relationship-id>
/SRC_TYPE=<product-id>:<item-type-name>
/DST_TYPE=<product-id>:<item-type-name>
```

Example

```
RIR "INCLUDES" -
/SRC_TYPE="PROD_X:C" -
/DST_TYPE="PROD_X:H"
```

- `<relationship-id>`

  Specifies the identifier of the relationship to be removed

- `/SRC_TYPE=<prod-id>:<item-type-name>`

  Specifies the source product and item type from which the link will start

- `/DST_TYPE=<prod-id>:<item-type-name>`

  Specifies the destination product and item type at which the link will end or point.

Description

The RIR command is used to remove an item-to-item relationship definition that was defined by the DIR command.

Constraints

Only users with the appropriate management privileges can run this command.

RIWS – Remove Item Revision from Project

NOTE This command is not available in the Issue Management–only licensed version of Dimensions.

```
<item-spec>
/WORKSET=<project-spec>
[/FILENAME=<file-name>]
```

Example `RIWS PROD_X:"HELLO WORLD".AAAA-SRC;2.6 /WORKSET=PROD_X:"WS DVLP"`

- `<item-spec>`

  Comprises:

  `<product-id>:<item-id>.<variant>–<item-type>;<revision>`

  `<item-id>` may be omitted if `<file-name>` is specified.

  `<variant>` may be omitted if only one exists.

  `<revision>` may be omitted if `<file-name>` is specified.

- `/WORKSET=<project-spec>` comprises:

  `<product-id>:<project-id>`

  This specifies the project from which the item revision is to be removed.

- `/FILENAME=<file-name>`

  Specifies the name of the project file name.

  The project file name identifies the relative path (directory plus file name) from the working location to the item to be used from the current project. The project file name for the same item may *differ* between projects; for example, `src/hello.c`, `hello.c`, or `src/build/hello.c`.

  It may be omitted if `<item-id>` is specified.

Description

This command will remove the item specified from the given project. The specified item and project must exist. If the specified item is not in the project a warning will be given.

NOTE Whenever a new revision is added to a project, its stage is reset to DEVELOPMENT, and associated deployment areas and reference areas are updated.

Constraints

Normally this command can be run only by a user with the appropriate management privileges for the project concerned.

This constraint can, however, be relaxed using the Set Project Permissions (SWSP) command, as described on page 342.

RLCA – Remove Library Cache Area

NOTE To delete a library cache area definition, you must have a Delete Library Cache Areas definition.

```
<area-name>
```

Example

```
RA <area-name>
```

Parameters

- `<area-name>`

  Specifies the name of the library cache area. Area names must be unique within the base database.

Description

The RLCA command deletes a library cache area definition. This command does not delete the contents of the area (files or folders) on disk.

Constraints

To delete an area definition, you must have the DELETE_AREA_PRIV privilege. It must be associated with the `object_owned_by_user` rule; that is, in order to delete an area definition, the user must be the owner of the area or a member of the group that is the owner of the area.

RLIST – Lists Jobs in the Job Queue

```
<job-key>
[/USER=<userid>]
[/NETWORK_NODE=<nodename>
[/DATE<date>]
[/TEMPLATE_ID=<template-name>
[/STATUS=<status>]
[/RC=xx]
[/[NO]DETAIL]
```

NOTE Any of the qualifier values described below may contain the % character to enable pattern matching in SQL.

- `<job-key>`

  string with format <id>-<job-uid> that identifies the job to list in the job queue. To list all jobs in the queue, execute RLIST with no parameters.

- `[/USER]`

  further restrict the search for jobs to list to those that match this user name.

- `[/NETWORK_NODE=<nodename>]`

  further restrict the search for jobs to list to those that match this network node name.

- `[/DATE=<date>]`

  further restrict the search for jobs to list to those that match this date. The date has to be in DD-MTH-YY format, for example, 03-FEB-04.

- `/TEMPLATE_ID=<template-name>`

  further restrict the search for jobs to list to those that match this template name. The template name maps to the template file located in the tertiary node directory specified by the value of the `DM_TEMPLATE_CATALOG<n>` symbols. Normally, these symbols are defined in the `dm.cfg` file on the network node; however, if they are not defined there, `%DM_ROOT%templates\<template-name>` is used on Windows, and `$DM_ROOT/templates/<template-name>` is used on UNIX and z/OS.

- `[/STATUS=<status>]`

  further restrict the search for jobs to list to those that match this status. A job's status can have the values FAILED, SUCCEEDED, or SUBMITTED.

- `[/RC=xx]`

  further restrict the search for jobs to list to those that match this return code.

- `[/DETAIL]`

  provide detailed job information in the list.

Description

RLIST enables you to list jobs in the job queue, either as summary or in detail.

For addition information concerning remote job entry, see the *Administrator's Guide*.

RMDF – Remove Data Formats

```
<format>
```

Example `RMDF TXT`

- `<format>`

  the format definition being removed.

Description

The RMDF command removes existing defined item or request type data formats. For a discussion on item or request type data formats refer to the DDF command (see page 125).

Constraints

Only users with the appropriate management privileges can run this command.

RMVB – Remove Version Branch

NOTE This command is not available in the Issue Management–only licensed version of Dimensions.

```
<branch-id>
[/[NO]FORCE]
```

Example `RMVB MAINT`

- `<branch-id>`

  Identifies a branch that was defined by the DVB command. If the branch is used to label any item pedigree trees it may **not** be removed unless the `/FORCE` qualifier is used.

- `/FORCE`

  removes branch-id even if it is being used. Warning messages will **always** be issued to the standard output following the forced removal of a used branch-id. Items with removed branch-ids will continue to exist, but no further revisioning operations will be possible with respect to them e.g. EI or UI will not be possible. You will continue to be able to get these items, and it is always possible to re-define the branch-ids again (using DVB) and carry on revisioning the items.

Description

The RMVB command removes existing defined version branch-ids. The `/FORCE` qualifier allows removal of branch-ids which are being used.

Constraints

Only users with the appropriate management privileges can run this command.

ROA – Retrieve Offline Archive

NOTE This command is not available in the Issue Management–only licensed version of Dimensions.

```
<archive-id>
```

Example `ROA AA12AB`

See the *Administrator's Guide* for details.

RP – Relate Design Part

```
<participle>
/PARENT_PART=<parent-part-spec>
```

Example `RP PROD:"LIBRARY CONTROL".AAAA /PARENT=PROD:"RELEASE MANAGEMENT".AABB`

- `<part-spec>`

  (both for the child design part and its new usage parent part[1]) comprises:

  ```
  <prod-id>:<part-id>.<variant>;<pcs>
  ```

  `<variant>` may be omitted if only one variant of that design part exists.

  `<pcs>` is ignored, the current PCS is always used.

Constraints

Only users with the appropriate management privileges can run this command.

RPCD – Relate Part to Requests

```
<part-spec>
/CHANGE_DOC_IDS=(<request1>,<request2>,…)
[/PENDING]
```

Example

```
RPCD PROD:"RELEASE MANAGEMENT".AAAA -
/CHANGE_DOC=(PROD_DR_25,PROD_DR_26)
```

- `<part-spec>` comprises:

  `<product-id>:<part-id>.<variant>;<pcs>`

  `<variant>` may be omitted if only one exists.

  `<pcs>` is ignored; the current PCS is always used.

- `/CHANGE_DOC_IDS=(<request1>,<request2>,…)`

  `<requestN>` is the identity of a request to which the specified design part is to be related.

- `/PENDING`

  Base the operation on the requests pending to the user.

Constraints

- This command can be run only by the user who has the request in their Pending list or by a user with the appropriate management privileges.
- This command cannot be used with the secondary catalog list.
- This command cannot be run if the request is at its end (closed) lifecycle state – even by a change-manager; however, a change-manager can action it back to an earlier lifecycle state perform the relate operation, and then action the request back to its closed state.

RPCP – Report Product Control Plan (Process Model)

```
<product-id>
[/[NO]DETAIL]
[/SECTION=(<section>,…)]
```

NOTE RPCP cannot be run from Dimensions for z/OS.

Example `RPCP PROD /DETAIL /SECTION=(ITEM,CDOC)`

- `<product-id>`

  Specifies the product which is to be reported.

- `/DETAIL`

  means print full details of the process model (formerly control plan) sections requested.

  If omitted, the report defaults to summary lists, except for RULE which is given in full.

- `/SECTION=(<section>,…)`

  Specifies which sections of the process model are to be reported. The following sections can be requested:

  - PCAT: reports on design part categories.
  - ITEM: reports on item types.
  - CDOC: reports on request types.
  - ROLE: reports on roles, users, privileges, and groups.
  - RULE: reports on change management rules.
  - ALL: (or if `/SECTION` is not specified) reports on all sections of the process model.

NOTE RPCP creates the report, with file name `cntl_plan.out`, in the current working directory. If you run the RPCP from the Windows Serena | Dimensions | Command Client, the current working directory will be `%DM_ROOT%\prog`.

Constraints

Only users with the appropriate management privileges can run this command.

RPNO – Report Part Numbers

```
<product-id>
```

Example `RPNO PROD`

- `<product-id>`

  Specifies for which product a part numbers report is to be produced. If specified as ***** (asterisk), then the report covers part numbers for all products in the database.

NOTE RPNO creates the report, with file name `pcms_partno.out`, in the current working directory. If you run the RPNO from the Windows Serena | Dimensions | Command Client, the current working directory will be `%DM_ROOT%\prog`.

Constraints

Only users with the appropriate management privileges can run this command.

RPROJ – Remove a Dimensions Project

NOTE This command is no longer available. Use RWS instead.

See the RWS command.

```
<product-id>
<request-category>
/NAME=<report-type>
/CATALOGUE or /CATALOGUE/SECONDARY or /PENDING
/FILENAME=<outfile>
[/ATTRIBUTES=(<attr1>=<string1>,<attr2>=<string2>,…)]
[/CHANGE_DOC_ID=<ch-doc-id>]
[/CH_DOC_TYPE=<ch-doc-type>]
[/PART=<part-spec>]
[/PHASE=<phase>]
[/STATUS=<request-status>]
[/USER=<user-name>]
[/FROM=<date-from>]
[/TO=<date-to>]
[/[NO]HEADING_WRAP]
[/[NO]WRAP]
[/[NO]INDENT]
[/[NO]DETAIL]
[/[NO]HOLD]
[/[NO]PRINT]
```

Example

```
RPT PROD 4 /NAME=CH_DOC_LIST /CATALOGUE /DETAIL -
/FILENAME=workpack_all.list /PRINT
```

- `<product-id>`

  This is the product for which a report is required.

- `<request-category>`

  This is the request category for all requests to appear in the report. Valid values for this indicator are 1, 2, 3, or 4. Category 1 is for bugs or problems, category 2 is for change or enhancement requests, category 3 is for "other" (miscellaneous), and category 4 is reserved for work packages.

  The value may be specified as **%** (percent sign) to permit requests from all categories to be included.

- `/NAME=<report-type>`

  Specifies which particular report is required (e.g. `CH_DOC_LIST`). For a full list of possible report types, refer to the "Change Management Reports" section of the *Reports Guide*.

  For a `BASELINE_DETAIL REPORT`, refer to the separate entry on page 313

- /CATALOGUE or /CATALOGUE/SECONDARY or /PENDING

  Determines the basis for the report. Only one of these qualifiers must be specified.

/CATALOGUE	report based on the primary (main) catalog of requests.
/CATALOGUE /SECONDARY	report based on the secondary catalog of requests.
/PENDING	report based on the requests pending to the user.

- /FILENAME=<outfile>

  Specifies the name of a file to receive the generated report. If /DETAIL is present, the file will include that output.

Additional Criteria to Ensure Inclusion in Report

All the remaining parameter values are used to specify additional criteria which requests must satisfy in order to be included in the report. **Except** for the parameters <date-from> and <date-to>, these parameters may include **%** (percent) characters as **wild cards** (i.e. each **%** character is considered to match zero or more characters in the corresponding request attribute).

The **default** in each case is to specify **no** additional criteria: i.e. all requests, which are otherwise valid for inclusion, may have any value for the corresponding attribute.

- /ATTRIBUTES=(<attr1>=<string1>,…)

  Specifies strings to be matched by the corresponding user-defined attributes in each of the requests to be reported on. Each <attrN> is the Variable Name defined for one of the attributes. Each <stringN> is a string for that attribute's value to match.

  Wild card **%** may be used in each <stringN> (see above).

- /CHANGE_DOC_ID=<ch-doc-id>

  This is a string to be matched by the identifier of each of the requests to be reported on.

  Wild card **%** may be used (see above).

- /CH_DOC_TYPE=<ch-doc-type>

  This is a string to be matched by the type of each of the requests to be reported on.

  Wild card **%** may be used (see above).

- /PART=<part-spec> comprises:

  <product-id>:<part-id>.<variant>;<pcs>

<pcs>	is ignored in the matching criteria.
<part-spec>	must match a design part which is related to a request, in order for that request to be included in the report.

  Wild card **%** may be used (see above).

- /PHASE=<phase>

  This is a string to be matched by the current phase of each of the requests to be reported on.

  Wild card **%** may be used (see above).

- /STATUS=<request-status>

  This is a string to be matched by the current status of each of the requests to be reported on.

  Wild card **%** may be used (see above).

- /USER=<user-name>

  This is a string to be matched by a Dimensions user associated with each of the requests to be reported on.

  Wild card **%** may be used (see above).

  A /CATALOGUE or /SECONDARY report includes all requests which (meet all other specified criteria and) have been created or actioned by a Dimensions user matching <user-name> at any time between <date-from> and <date-to>.

  A /PENDING report includes all requests which (meet all other specified criteria and) are awaiting actioning by a Dimensions user matching <user-name> at any time between <date-from> and <date-to>. (A request can be awaiting actioning by any of several Dimensions users. For valid inclusion of the request in the report, at least one of these users must match <user-name> as specified here, but they need not all do so.)

- /FROM=<date-from>

  Specifies the initial date associated with each of the requests to be reported on. (For usage, see details immediately above.) It must be in the format 25–DEC–1996.

  The default is 01–JAN–1900.

- /TO=<date-to>

  Specifies the final date associated with each of the requests to be reported on. (For usage, see details immediately above.) It must be in the format 25–DEC–1996.

  The default is the current system date.

- /NOHEADING_WRAP

  Specifies that any columns in the report headings where the data exceeds the column width are to be truncated.

  The default is that these columns are word wrapped.

- /NOWRAP

  Specifies that any columns in the body of the report where the data exceeds the column width are to be truncated.

  The default is that these columns are word wrapped.

- /INDENT

  Specifies that any lines in the body of the report that start with a number are to be indented by four times that number of columns.

  By default there is no indentation.

- /DETAIL

  Specifies that the full text of each request listed in the report is to be printed following the report.

  By default just the report itself is printed.

- /HOLD

  Indicates that the report request is to be processed so as to produce a command script file, but that this is not to be executed.

  By default a Dimensions batch job to generate the report is submitted immediately.

- /PRINT

  Indicates that the report is also to be sent to a printer.

  By default the completed report is merely placed in <outfile>.

Constraints

Only users with the appropriate management privileges can run this command.

RPT – Baseline Detail Report

NOTE This command is not available in the Issue Management–only licensed version of Dimensions.

```
<product-id>
<request-category>
/NAME=BASELINE_DETAIL
/USER_DEFINED
[/BASELINE=<baseline-spec>]
[/FROM=<date-from>]
[/TO=<date-to>]
/FILENAME=<outfile>
[/[NO]HEADING_WRAP]
[/[NO]WRAP]
[/[NO]INDENT]
[/[NO]HOLD]
[/[NO]PRINT]
```

Example

```
RPT PROD 1 /NAME=BASELINE_DETAIL /USER_DEFINED -
/FILENAME=hp_rm_chdocs96.rep -
/BASELINE=PROD:"R M VERSION 2 FOR HP" -
/FROM=01-JAN-1989 /TO=31-DEC-1996
```

- `<product-id>`

  This is the product for which the report is required.

- `<request-category>`

  This is any valid request category. (Its value is not used in this report.)

- `/NAME=BASELINE_DETAIL`
  `/USER_DEFINED`

  are specified as shown.

  For other types of report, refer to the **RPT** entry for **Report Requests**, which precedes this one on page 309.

- `/BASELINE=<baseline-spec>`

Specifies the baseline(s) to be reported on. It comprises:

`<product-id>:<baseline-id>`

Wild card **%** may be used in `<baseline-spec>` to determine which baselines are to be included (i.e. each **%** character is considered to match zero or more characters in the corresponding baseline).

The default is to report on all baselines in the product.

- `/FROM=<date-from>`
  `/TO=<date-to>`

  specify the first and final dates on which an item, included in a reported baseline, may have been related to a request, in order for the request to be listed in the report. Each must be in the format 25–DEC–1996.

The defaults are 01–JAN–1900 and the current system date.

NOTE Any other RPT qualifiers from the Report Requests command on page 309 used here (including /DETAIL) will be ignored.

Constraints

Only users with the appropriate management privileges can run this command.

RRCD – Relate Requirement to Request

NOTE This command is not available in the Issue Management–only licensed version of Dimensions.

```
RRCD /CHANGE_DOC_ID=<request_id> /REQUIREMENT_ID=<requirement>
```

Description

`<request-id>` is the request to be related to the requirement.

`<requirement>` is the requirement to be related to the request.

Constraints

Only users with the appropriate management privileges can run this command.

RREG – Reassign User Registration

```
<existing-user-id>
/USER=<new-user-id>
```

Description

This command is documented in the *Serena Dimensions CM Administrator's Guide*.

RSTAT – Update Job Status

```
<job-key>
[/STATUS=<status>]
[/RESPONSE_FILE=<response-file>]
[/DESCRIPTION=text
[/RC=xx]
[/USER_FILENAME=<report-file>]
```

- `<job-key>`

  string with format <id>-<job-uid> that identifies the job whose attributes are to be updated.

- `[/STATUS=<status>]`

  updates the status of the job in the job queue table. Valid values are FAILED or SUCCEEDED.

NOTE The status of an asynchronously submitted job cannot be reset to SUBMITTED.

- `[/RESPONSE_FILE=<response-file>]`

  name of a file that contains a summary of the results of executing the remote job. In the case of ChangeMan Builder, the response file contain an XML build summary detailing which targets were built.

- `[/DESCRIPTION=text]`

  text that is used to update the job description.

- `[/RC=xx]`

  a numeric return code indicating whether or not the remote job successfully completed.

- `[/USER_FILENAME=<report-file]`

  Specifies a file where the RSTAT execution report is to be generated.

Description

RSTAT enables you to update job attributes such as return code, status, and description.

RSTAT is also used by the default ChangeMan Builder templates to report the status of asynchronously submitted build jobs and implement collection of build outputs.

For addition information concerning remote job entry, see the *Administrator's Guide*.

RUR – Run User-Defined Report

```
<report name>
/PRODUCT=<product-range>
[/PARAMETER=(<param2>,<param3>,…,<param8>)]
[/FILENAME=<output file>]
[/[NO]PRINT]
[/[NO]HOLD]
```

Example

```
RUR "VARIANTS TYPES FORMATS" /PROD="CAR%" -
/PARAM=(AAA_,"%",C) /PRINT
```

- `<report name>`

  Specifies the kind of report to be produced. It must be one of the report names containing report definitions that has been set up via one of the following means:

  - The User Reports Administration cluster of the web-based Administration Console.
  - A JavaScript, utilizing the Dimensions `dmpmcli` scripting interface. Refer to the *Process Modeling User's Guide* for instructions on how to run such scripts. Example reports scripts for UNIX and Windows are available, online, in the directories `$DM_ROOT/AdminConsole/examples/reportsDemo.js` and `%DM_ROOT%\AdminConsole\examples\reportsDemo.js` respectively

- `/PRODUCT=<product-range>`

  This is a string to be matched by the product-id of one or more of the products in the database. "Wild card" characters may be used in the specified string — this is explained further in the *Reports Guide*.

  The Dimensions user executing the RUR command must hold a role in *every* product matched by the `<product-range>` string; otherwise execution of the command is terminated with an error message.

- `/PARAMETER=(<param2>,<param3>,…,<param8>)`

  This is a set of parameter values, separated by commas and enclosed in parentheses, which are the parameter values to be supplied to this report following the `<product-range>` string (which is always the value of parameter number 1).

  The last parameter, for which a value is supplied, may well be less than number 8 – it depends on the particular requirements of the command script file for the specified report name. However, note the following:

  There must be the exact number of non-blank values, in the correct order, as required by the `<report name>` specified.

  Lower case letters may be included in parameter values, in addition to the standard character set. **%** (percent) characters are also accepted. Any parameter value which contains embedded blanks must be enclosed in a set of double-quotation characters (" ").

NOTE The default of **%** for an omitted parameter value **is not applicable** in command mode. If a single **%** is wanted as a parameter value, then that must be coded. Two consecutive commas in the set of parameter values will be flagged as a syntax error.

There is no default. The parameters **must be supplied exactly** as required by the `<report name>`. The purpose of the square brackets is to indicate that the /

PARAMETER qualifier **must be omitted**, if the report name is one which takes **no** input parameter values, apart from the single <product-range> string.

- /FILENAME=<output file>

  Specifies the name of a file which RUR will create in the user area, and into which it will write the output from the report. If omitted, the file name will be user_report.out

- /PRINT

  Specifies that the report output is also to be spooled for printing. If omitted, the report is merely placed in <output file> (specified or default).

- /HOLD

  Specifies that the batch job to produce the report is to be created, but that it is to be left in the user area for the user to start its execution later. If omitted, execution of the batch job is initiated automatically as soon as it has been created.

Constraints

This command can be run by users who hold a role, either for the single product-id specified or for every one of the product-ids designated by a string using wildcard characters. However, if a user with the role of PRODUCT-MANAGER assigns a special role PCMS-CM-USER to the wild card user *, then any user will able to run a report on the product even if they do not have such a role explicitly assigned.

RWCD – Relate Project to Request

NOTE This command is not available in the Issue Management–only licensed version of Dimensions.

```
<project-spec>
/CHANGE_DOC_IDS=(<request1,<request2>,request3>,…)
```

Example `RWCD PAYROLL:PRJ_INITIAL /CHANGE_DOC_IDS=(PAYROLL_CR_21)`

Description

The example above relates the PAYROLL_CR_21 request to the PAYROLL:PRJ_INITIAL project.

Constraints

Only users with the appropriate management privileges can run this command.

RWS – Remove Project

NOTE This command is not available in the Issue Management–only licensed version of Dimensions.

```
<project-spec>
```

Example

```
RWS "PROD_X:TEST_WS"
```

- `<project-spec>` comprises:

  `<product-id>:<project-id>`

Constraints

This command can be run only by a user with the appropriate management privileges for the project concerned.

A project used as a child collection cannot be deleted.

RWWS – Relate Project to Project

NOTE This command is not available in the Issue Management–only licensed version of Dimensions.

```
<child-project-spec>
/WORKSET=<parent-project-spec>
[/RELATIVE_LOCATION=<relative-path]
```

Example `RWWS <child-project-spec> /WORKSET=<parent-project-spec>`

Parameters

- `<child-project-spec>`

  Specifies the child project in the parent-child relationship.

- `<parent-project-spec>`

  Specifies the parent project in the parent-child relationship.

- `<relative-path>`

  Specifies the relative path of the file system directory to which the child project's top-level directory is mapped with respect to the file system directory to which the parent project's top-level directory is mapped.

NOTES
– Specifying an empty string clears the relationship-level project root for the user.
– If there is no relationship-level project root, /RELATIVE_LOCATION is used.
– If the path contains `::` (that is, it has the format `<node-or-area>::<path>`, and `<node-or-area>` matches the name of an existing area), the path is interpreted as an offset (relative path) with regard to the area root directory. (This offset may be empty, in which case the file system directory is set to equal the area directory.) Otherwise, standard Dimensions interpretation of the path applies.

Description

Creates or modifies a parent-child relationship between the specified child project and the specified parent project.

If a relationship already exists, the value of /RELATIVE_LOCATION is used to update the relationship.

Constraints

Only users with the appropriate management privileges can run this command.

SCWS – Set Current Project

```
[<project-spec>]
[/ROOT_PROJECT=<project-spec>]
[/DIRECTORY=<directory>]
[/LIBRARY_CACHE_AREA=<area-name>]
[/CHANGE_DOC_ID=<request-id>]
[/[NO]DEFAULT]
[/[NO]CHECK]
```

Example `SCWS "PROD:WS CUSTOM" /DIR="/product9" /NODEFAULT`

- No parameters or qualifiers

  SCWS without any parameters or qualifiers will display:

  - Login user name
  - The current project specification.
  - The working location.
  - Whether the project is locked or not.
  - Whether trunk or branch revisioning is enabled.
  - Whether or not revisioning is enforced.
  - Branches assigned to the project.
  - The project associated with the specified project.

NOTE SCWS without parameters or qualifiers displays the current project of the current Dimensions session. Additionally, it is only refreshed from the database at the start of the session.

- `<project-spec>`

  Comprises:

  `<product-id>:<project-id>`

`<project-spec>`	Must be stated when `/DIRECTORY` and/or `/(NO)DEFAULT` are included in the command.

- `/ROOT_PROJECT=<project-spec>`

  Comprises:

  `<product-id>:<project-id>`

  This optionally specifies the root project. Use this when the current project occurs in more than one project tree. If you set this here, you do not need to set it on subsequent commands.

- `/DIRECTORY=<directory>`

  Specifies the top level directory specification for the project. This "working location" defines the point in the directory hierarchy structure below which (or relative to which) the project file name is placed e.g. in UNIX `<dir>/<ws_filename>`. This overrides what would have been the default working location.

Operating system permissions permitting, you can specify a Windows network path when running SCWS from a UNIX host as follows:

```
/DIRECTORY=/"nodename::<Drive>\<path>"
```

For example:

```
/DIRECTORY=/"earth::F:\Optimus"
```

Checked out or fetched items will then be directed to the project file whose path is identified by this new path name (e.g. in UNIX, <dir>/<ws_filename>). You can also use a work area.

NOTE If the path contains :: (that is, it has the format <node-or-area>::<path>, and <node-or-area> matches the name of an existing work area), the path is interpreted as an offset (relative path) with regard to the work area root directory. (This offset may be empty, in which case the file system directory is set to equal the area directory.) Otherwise, standard Dimensions interpretation of the path applies.

- /LIBRARY_CACHE_AREA=<area-name>

  Specifies a library cache area defined with the CLCA (Create Library Cache Area) command. During fetch operations (FI, FWI, FBI, FCDI, EI, EWI, EBI, ECDI, DOWNLOAD), Dimensions checks whether the library cache area associated with the current project already contains a copy of of the requested file. If so, Dimensions copies the file from the library cache to the user file area instead of from the library itself, which improves performance when the connection between the item library node and the user's network is slow.

- /CHANGE_DOC_ID=<request-id>

  Sets a default per-user request on the project. To unset a default request, set /CHANGE_DOC_ID to ".".

- /DEFAULT or /NODEFAULT

/DEFAULT	Specifies that the current project specified by this command will remain the default for all future Dimensions sessions until respecified. /DEFAULT is the default.
/NODEFAULT	Specifies that the current project specified by this command is for the duration of the present Dimensions session only (this should not be confused with your operating-system session). The current project will revert to its former setting once the session is exited. This project will also be used for batch jobs started during the Dimensions session.

  The /NODEFAULT qualifier should, however, be treated with caution when submitting command files containing SCWS and commands that spawn separate operating-system processes e.g. Create Baseline (CBL). The scope of the /NODEFAULT qualifier does not get extended to such commands. To guarantee correct behavior, it is recommended that SCWS /DEFAULT is used to set the current project before such commands as CBL, and then used again afterward to reset it to its earlier specification (if so desired).

- /CHECK or /NOCHECK

  Specifies whether SCWS is merely to check and report on the feasibility of performing the requested action, or is actually to implement it. /CHECK is the default.

Example Output

```
Dimensions>scws
The current project is PVCS:DOCS_DIM_10
The working location is D:\Dimensions
The current user is JDOE
        Description         : Work Set DOCS_DIM_10.AAAA
        Status              : OPEN
        Trunk               : FALSE
        Branch              : TRUE
        Enforce Revisioning : TRUE
        Parallel Extraction : TRUE
        Deployment          : AUTOMATIC
There are no valid Named Branches for this project
The project library cache area is not defined
Operation completed
```

Description

This command sets the user's current project, and optionally the user's default project.

To set the current project to the "Global Workset", the `<project-spec>` must be set to `$GENERIC:$GLOBAL` (see Introduction on page 16).

Constraints

This command can be run by users who have a role on the project being set.

SDF – Set Data Format Flags

```
<format>
[/DESCRIPTION=<format -description>]
[/CLASS=<class-no>]
[/MIME_TYPE=<mime-type>]
[/COMPRESSION_LEVEL=<level>]
```

Example

```
SDF TXT /DESCRIPTION="Plain text" -
/CLASS=1 /MIME_TYPE="TEXT/PLAIN"
```

- `<format>`

  the format definition being updated.

- `/DESCRIPTION=<format-description>`

  the new descriptive name for the format.

- `/CLASS=<class-no>`

  Specifies the new file types, where:

  1=TEXT
  2=BINARY
  3=OpenVMS
  4=Macintosh
  5=NT

- `/MIME_TYPE=<mime-type>`

  Specifies the new Multipurpose Internet Mail Extension (MIME) type. MIME types comprise seven broad categories, with each category also having subcategories defined by using a forward slash (/) separator. The broad categories are: Application, Audio, Image, Message, Multipart, Text and Video. An example of a subcategory is APPLICATION/WORD.

- `COMPRESSION_LEVEL=<level>`

  Specifies the compression level to be used when getting item revisions assigned this data format. Use a digit from 0 to 9, where 0 indicates no compression, 1 means fastest compression method (but less compression) and 9 indicates slowest compression method (but best compression). If this qualifier is omitted, text file formats will use fastest compression method (level 1) while all other file formats will use no compression.

Description

This command enables a user with the role of TOOL-MANAGER to update the file format definition. These defined file formats can then, where appropriate, be subsequently assigned:

- By a user with the role of TOOL-MANAGER to particular item types using the Assign Data Formats to Item Types (ADF) command (page 45).
- By a user with the role of PRODUCT-MANAGER or CHANGE-MANAGER to a particular request type using the Assign Data Formats to Request Types (ACF) command (page 44).

This function is also available from the Administration Console, Data Formats and MIME Types option.

See the DDF command (page 125) for a description of the uses for file formats.

Constraints

This command can be run only by a user with the appropriate management privileges for the product.

SET – Set DIR, PRINTER, OVERWRITE, or CMD_TRACE Environment

```
PRINTER <printer-cmd>
or
DIRECTORY <directory-spec>
or
OVERWRITE ON|OFF
or
CMD_TRACE ON|OFF [/USER_FILENAME=<file-name>]
or
INFO ON|OFF
```

Example `SET PRINTER "lpr -Pdev"`

- `PRINTER <printer-cmd>`

  Set a valid UNIX or Windows printer command. This is used in preference to the value assigned to the *DM_PRINT* symbol.

- `DIRECTORY <directory-spec>`

  Set the current Dimensions default directory is set to the directory specified. This must be an appropriate format for the host machine operating system, and it can be absolute or relative to the current directory.

- `OVERWRITE [ON]|OFF`

  When checking out or getting an item revision, you can specify whether or not Dimensions is allowed to perform such an operation depending on:

  - The existence of a local file of the same name.
  - The status (read-only or writeable) of an existing local file of the same name.

  No overwriting is the normal default and results in a file only being successfully checked out or gotten by Dimensions if the local (target) file does not already exist or is marked read-only. This is the traditional Dimensions behavior (with respect to the file being read-only, the assumption is that if it is writable then the file could potentially be a more recently modified revision of the item that the user does not want to lose). This default can be overridden on a per command basis using `/OVERWRITE` qualifier of EI and FI as explained on page 178 and page 186 respectively.

  The `SET OVERWRITE ON | OFF` command allow you to control – on a per Dimensions session basis – the default behavior globally without needing to specify the `/OVERWRITE` or `/NOOVERWRITE` qualifier on every FI or EI command. At the beginning of a Dimensions session, by default `SET OVERWRITE OFF` would be in effect; however, you could change this environment as illustrated in the following examples:

  ```
  SET OVERWRITE OFF
  FI "FS:CBEVENT C.A-SRC;b1#4" -
  /USER_FILENAME="e:\test\cbevent.c" /NOEXPAND
  ```

  would not allow `cbevent.c` to be overwritten if it existed and was not marked read only.

  ```
  SET OVERWRITE ON
  FI "FS:CBEVENT C.A-SRC;b1#4" -
  /USER_FILENAME="e:\test\cbevent.c" /NOEXPAND
  ```

would overwrite `cbevent.c` if it existed, irrespective as to whether it was marked read only or not.

- `CMD_TRACE [ON]|OFF [/USER_FILENAME=<file-name>]`

  Executing
  `SET CMD_TRACE ON`

  creates a log file[1] in the DM_TMP directory of the format

  Windows:
  `%DM_TMP%dmappsrvcmd_<user-name>.log`

  UNIX:
  `$DM_TMP/dmappsrvcmd_<user-name>.log`

  which records all the commands and session details run through the user's session.

  You can explicitly specify the name of the log file created on the server by the use of the `/USER_FILENAME` qualifier.

  Executing
  `SET CMD_TRACE OFF`

  will switch off the command logging.

  For a sample of the information contained in this log file, refer to "Logging All Commands Run by All Users" on page 33.

Constraints

This command sets the above Dimensions parameters only for the duration of the current Dimensions session.

1. The log file is created on the Dimensions server, *not* the client.

SI – Suspend Item

NOTE This command is not available in the Issue Management–only licensed version of Dimensions.

```
<item-spec>
[/FILENAME=<file-name>]
[/WORKSET=<project-spec>]
```

Example `SI PROD:"QUERY RELEASE".AAAA-SRC;1`

or, to suspend all revisions

SI PROD:"QUERY RELEASE".AAAA-SRC;*

- `<item-spec>` comprises:

  `<product-id>:<item-id>.<variant>-<item-type>;<revision>`

`<item-id>`	may be omitted if `<file-name>` is specified.
`<variant>`	may be omitted if only one exists.
`<revision>`	may be specified as * (asterisk) to suspend all revisions of the item that are in the project. If omitted, the latest revision is suspended (see note in Introduction on page 16).

- `/FILENAME=<file-name>`

  Specifies the name of the project file name.

  The project file name identifies the relative path (directory plus file name) from the working location to the item to be used from the current project. The project file name for the same item may *differ* between projects; for example, `src/hello.c`, `hello.c`, or `src/build/hello.c`.

  It may be omitted if `<item-id>` is specified.

- `/WORKSET=<project-spec>` comprises:

  `<product-id>:<project-id>`

  This optionally specifies the project to be used for this command: failing this, the user's current project will be taken.

  Item revisions to be affected by the command may be specified explicitly, or they will be selected from the project.

NOTE Suspended items will be included in a revised baseline if they are identified as requiring updates based on a related request used in the revised baseline specification.

Constraints

This command can be run at any lifecycle state either by a user with the appropriate management privileges or by a user for whom the item is pending.

A suspended item may be 'unsuspended' by actioning it to a valid state in the lifecycle.

SPSP – Set Per-Stage Preservation Policy

NOTE This command is not available in the Issue Management–only licensed version of Dimensions.

```
/PROJECT=<project-spec> /STAGE=<deployment-stage>
[/POLICY=<policy-id> | /RESET_POLICY]
```

Example

The command

```
SPSP /PROJECT="PAYROLL:EXEDLL 2.0" /STAGE="UNIT TEST" -
/POLICY="PAYROLL:DEFAULT_POLICY"
```

assigns a preservation policy to UNIT TEST builds of the PAYROLL:EXEDLL 2.0 project.

- `/PROJECT=<project-spec>`

  Comprises `<product-id>:<project-id>` and specifies the project.

- `/STAGE=<deployment-stage>`

  Specifies the deployment stage.

- `/POLICY=<policy-spec>`

  `<policy-spec>` comprises `<product--id>:<policy-id>` and specifies the preservation rules policy to be applied at this stage for the specified project.

- `/RESET_POLICY`

  Specifies that the preservation policy is to be reset.

Description

The SPSP command specifies per-stage project build properties that apply to all the build areas defined for the specified build stage within a project.

SPV – Suspend Design Part Variant

<part-spec>

Example SPV PROD:"RELEASE MANAGEMENT".IBM

- <part-spec> comprises:

  <product-id>:<part- id>.<variant>;<pcs>

  <variant> may be omitted if only one exists.

  <pcs> is ignored; the current PCS is always used.

Description

This command enables a design part variant that is no longer required to be suspended at its current PCS. In the SUSPENDED state a design part variant can serve no useful purpose in the design process. Also, once set to this state it cannot be restored to active use at its current PCS. Only by creating a new PCS via the UP command can a design part variant be restored to active use.

Constraints

This command can be run only by a user with the appropriate management privileges for the selected design part. This design part must be in an OPEN state but not be referenced in an open request.

SUB – Subscribe to Notification Rule

```
<notification-id>
[/[NO]DIGEST]
[/ROLES=(role1,role2,…)]
[/USER_LIST=(user1,user2,…)]
```

Example `SUB <rule-id> /USER_LIST=Smith`

- `<notification-id>`

  Name of the notification rule.

- `/DIGEST`

  Enables this subscription for digests (notification summaries).

- `/ROLES`

  Specifies roles to subscribe to this notification rule.

- `/USER_LIST`

  Specifies users to subscribe to this notification rule.

Description

Subscribe users to a notification rule.

SVBF – Set Version Branch Flags

NOTE This command is not available in the Issue Management–only licensed version of Dimensions.

```
<branch-id>
[/DESCRIPTION=<description>]
[/[NO]LOCK]
[/OWNER]
```

Example `SVBF MAINT/LOCK /OWNER=LOCAL`

- `<branch-id>`

  unique branch identifier.

- `/DESCRIPTION=<description>`

  brief description of the purpose for the branch.

  If omitted, the description last entered (using DVB or SVBF) remains unchanged.

- `/LOCK`

  Optional flag to specify that the branch is locked and further development along it cannot take place.

- `/NOLOCK`

  Optional flag, negation of LOCK and is the default.

- `/OWNER=<site_id>`

  where `<site_id>` is either:

  - LOCAL, a keyword which can be used to set the ownership to the local base database, or
  - `<node_name>:<dbname>@@<sid>`

  `<node_name>` = the node name

  `<dbname>` = the base database

  `<sid>` = the # sid

NOTE @@ is used because @ is the default Dimensions Escape character for the command line. The parameter OWNER enables change in branch ownership created by the Replicator product.

Description

This command modifies (sets) branch-id definitions that were defined using the DVB command i.e. the description or lock status.

Constraints

Only users with the appropriate management privileges can run this command.

SWF – Set Project File Name

NOTE This command is not available in the Issue Management–only licensed version of Dimensions.

```
<item-spec>
[/FILENAME=<file-name>]
/WS_FILENAME=<ws-filename>
[/WORKSET=<project-spec>]
```

Example

```
SWF PROD:"QUERY RELEASE"-SRC /FILENAME=qr.c -
/WS_FILENAME=maintenance/src/system.c
```

This command is used to change the name of the file associated with an item in a project. The new file name may include a directory path relative to the project (for example, `src/foo.c`.)

- `<item-spec>` comprises:

  ```
  <product-id>:<item-id>.<variant>–<item-type>;<revision>
  ```

`<item-id>`	identifies the item within the product.
`<variant>`	if omitted, the default (specified when the product was defined) is used.
`<revision>`	is ignored, all revisions within the project will be affected.

- `/FILENAME=<file-name>`

  Specifies the name of the project file name.

  The project file name identifies the relative path (directory plus file name) from the working location to the item to be used from the current project. The project file name for the same item may *differ* between projects; for example, `src/hello.c`, `hello.c`, or `src/build/hello.c`.

  It may be omitted if `<item-id>` is specified.

- `/WS_FILENAME=<ws_filename>`

  This is the new project file name for the item in the project.

- `/WORKSET=<project-spec>` comprises:

  ```
  <product-id>:<project-id>
  ```

  This optionally specifies the project to be used for this command. If unspecified, the user's current project will be taken.

NOTE A project file name (`<ws_filename>`) can also be assigned to an item when it is created (see Dimensions command CI on page 87). A particular Dimensions item can have different project file names in different projects.

Note on Areas

Whenever a new revision is added to a project, its stage is reset to DEVELOPMENT, and associated deployment areas and reference areas are updated.

Constraints

This command can be run only on pending item revisions by a user with the appropriate management privileges for the project concerned, or by users whose privileges have been extended by a user with the appropriate management privileges because the users have a role on the product.

This constraint can, however, be relaxed using the Set Project Permissions (SWSP) command, as described on page 342.

SWS – Set Project Attributes

NOTE This command is not available in the Issue Management–only licensed version of Dimensions.

```
<project-spec>
[/BRANCH|/TRUNK]
[/[NO]AUTO_REV]
[/DESCRIPTION=<description>]
[/VALID_BRANCHES=(<branch-id1>,<branch-id2>,…)]
[/DEFAULT_BRANCH=<branch-id>]
[/FILENAME=<report-filename>]
[/[NO]POPULATE]
[/[NO]PARALLEL_EXTRACT]
[/DEPLOYMENT_MODEL=MANUAL or /DEPLOYMENT_MODEL=AUTOMATIC]
[/COPY_ON_DEPLOY]
```

Examples

```
SWS PROD:"WS MAIN DVL" /BRANCH

SWS PROD:"WS MAINT DVL" /VALID_BRANCHES=(maint,upgrade)
```

- `<project-spec>` comprises:

  `<product-id>:<project-id>`

- `/BRANCH`

  Optional qualifier to adopt "branching" for the item revision scheme. This means that if an item-revision is at revision `maint#5`, and the users decide to stay on this `maint` branch, then subsequent revisions will be `maint#5.1`, `maint#5.2`, `maint#5.3` etc.

- `/TRUNK`

  Optional qualifier to adopt "trunking" for the item revision scheme. This means that if an item-revision is at revision `maint#5`, and the users decide to stay on this `maint` branch, then subsequent revisions will be `maint#6`, `maint#7`, `maint#8` etc.

- `/AUTO_REV`

  Optional qualifier to tell Dimensions to automatically generate a new revision each time an item-spec is edited/updated.

- `/NOAUTO_REV`

  Optional qualifier to tell Dimensions **not** to automatically generate a new revision each time an item-spec is edited/updated, and instead request the user to supply a revision.

- `/DESCRIPTION=<description>`

  Optionally specify a new description to be attached to the project definition, thus replacing the one which was assigned when the project was created (with the DWS command).

- `/VALID_BRANCHES=(<branch-id1>,<branch-id2>,…)`

  Identifies one or more branches–each previously defined in a Define (Item) Version Branches (DVB) command–that are to be valid for new item revisions created in this existing project. The list of valid branch-ids replaces the list (if any) specified previously for this project using DWS or SWS.

This list specifies the branches on which newly created item revisions can be placed.

If the project attributes are set to have *one or more* valid branches, every *new* item revision in the project must use one of these branch-ids.

If the project attributes are set to have *no* valid branches, new revisions with no branch-ids in them can continue to be used.

- `/DEFAULT_BRANCH=<branch-id>`

  selects, from the valid-list of branch-ids, the branch-id to be the default branch for the whole project. If a default branch-id is not defined, the first branch-id in the valid-list of branch-ids is taken as the default.

- `/FILENAME=<report-filename>`

  Specifies the output file name for a report.

- `/[NO]POPULATE`

  Populates the associated build areas.

- `[/[NO]PARALLEL_EXTRACT]`

  Stops you checking out (extracting) an item if a revision of that item is already checked out. This behaves in the same manner as "Allow Parallel Checkout" for item type options, but with respect to all item types on a per project basis. See "About Item Type Options" in the "Object Type Definitions" chapter of the *Process Modeling User's Guide* for a discussion of parallel check out, and other places in that guide for a discussion of parallel development in general.

- `/DEPLOYMENT_MODEL=MANUAL or /DEPLOYMENT_MODEL=AUTOMATIC`

  Specifies whether the project uses a manual or automatic deployment model.

- `/COPY_ON_DEPLOY`

  Optional qualifier that specifies whether files that are deployed from an earlier stage to a newer stage will be copied between areas rather than moved (they are moved by default).

Description

This command is deprecated. Use the UWA command instead.

The SWS command is used to set (or reset) the attributes of an existing project created using the DWS command.

NOTE The `/BRANCH`, `/TRUNK`, `/AUTO_REV` and `/NOAUTO_REV` qualifiers may further be used to alter the options associated with the project. The permitted combinations of these qualifiers are:

```
SWS <project-spec> /BRANCH
SWS <project-spec> /TRUNK
SWS <project-spec> /AUTO_REV
SWS <project-spec> /NOAUTO_REV
SWS <project-spec> /BRANCH /AUTO_REV
SWS <project-spec> /BRANCH /NOAUTO_REV
SWS <project-spec> /TRUNK /AUTO_REV
SWS <project-spec> /TRUNK /NOAUTO_REV
```

Constraints

This command can be run only by a user with the appropriate management privileges for the specified project.

SWSP – Set Project Permissions

NOTE This command has been superseded by privileges for project operations.

```
Command no longer available.
```

TBI – Transfer Baseline In

NOTE This command is not available in the Issue Management–only licensed version of Dimensions.

```
<tbo-id>
/PART=<part-spec>
[/DEVICE=<device-id> /TAPE=<tape no.> /VOLUME=<volume-id> or
    /DEVICE=NONE]
/CATEGORY=<replacement-category>
[/DIRECTORY=<directory>]
[/REPORT or /TOKEN]
[/CHANGE_DOC_IDS=(<request-type>,…) or /CHANGE_DOC_IDS=*]
[/WORKSET=<project-id>]
[/SOURCE_OS=WINDOWS or UNIX]
```

Example

```
TBI TB12AB /PART="PRODY:P123" -
/CATEGORY=SUBPRODUCT /CHANGE_DOC_IDS=* /DEVICE="/dev/rst0" -
/TAPE="ta100" /VOLUME="tb100" -
/DIRECTORY="/usr/jones/work"
```

Example

```
TBI TB12AB /PART="PRODY:P123" /CATEGORY=SUBPRODUCT -
/CHANGE_DOC_IDS=* /DIRECTORY="c:\usr\smith\work"
```

See the *Administrator's Guide* for details.

TBO – Transfer Baseline Out

NOTE This command is not available in the Issue Management–only licensed version of Dimensions.

```
<tbo-id>
/BASELINE=<baseline-spec>
[/DEVICE=<device-id> /TAPE=<tape no.> /VOLUME=<volume-id>
    or /DEVICE=NONE]
[/DESCRIPTION=<description>]
[/DIRECTORY=<directory>]
[/REPORT or /TOKEN]
[/CHANGE_DOC_IDS=(<request-type>,…) or /CHANGE_DOC_IDS=*]
```

Example

```
TBO TB12AB /BASELINE="PRODX:BL12AB" -
/DEVICE="/dev/rmt0h" /TAPE="ta100" /VOLUME="tb100" -
/DIRECTORY="/usr/smith/work" -
/CHANGE_DOC_IDS=(PR,CR) -
/DESC="12AB transfer - sources & requests"
```

Example

```
TBO TB12AB /BASELINE="PRODX:BL12AB" -
/DIRECTORY="c:\usr\smith\work" -
/CHANGE_DOC_IDS=(PR,CR) -
/DESC="12AB transfer - sources & requests"
```

See the *Administrator's Guide* for details.

UA – Update Area

NOTE This command is not available in the Issue Management–only licensed version of Dimensions.

```
<area-name>
/NEW_NAME=<new-name>
[/DESCRIPTION=<area-description>]
[/NETWORK_NODE=<node-name>]
[/DIRECTORY=<HLQ/directory>]
[/TYPE=<area-type> [/STAGE=<stage-name>]]
[/USER_LIST=(<user-or-group>,<another-user-or-group>,...)]
[/USER=<user-name> [/PASSWORD=<password>]]
[/TRANSFER_SCRIPTS=<script-set>]
[/OWNER=<user-name> or <group-name>]
[/ADD] or [/DELETE]
[/STATUS=ONLINE or OFFLINE]
[/FILTER=<area-filter>]
```

Example

```
UA <area-name> /NETWORK_NODE=<host-machine> /DIRECTORY=<area-directory>
    /TYPE=WORK USER_LIST=(<user1>,<user2>,<user3>)
```

Parameters

- `<area-name>`

  Specifies the name of the area. Area names must be unique within the base database.

- `<new-name>`

  Specifies the new name for the area.

- `/DESCRIPTION=<description>`

  Specifies a description for the new area.

- `/NETWORK_NODE=<node-name>`

  Specifies the machine hosting the area.

- `/DIRECTORY=<HLQ/directory>`

  Specifies the directory, or PDS (partitioned data set), where the area is located.

  HLQ is a high-level qualifier; for example, MERVK.WORK. It is a common prefix for all data sets in the area such as MERVK.WORK.C, MERVK.WORK.CBL, and so forth.

- `/TYPE=<area-type>`

  Specifies the type of the area: WORK or DEPLOYMENT.

  Below is a mapping between Dimensions 9 and Dimensions 10 area types:

Dimensions 9	Dimensions 10
Development Area (working location)	Work Area
Managed Development Area	Deployment Area associated with stage DEVELOPMENT
Build Area associated with stage <XXX>	Deployment Area associated with stage <XXX>

- /STAGE=<stage-name>

  Applicable only to deployment areas. If the area type is DEPLOYMENT, this qualifier specifies the stage with which the new deployment area is associated.

- /USER_LIST=(<user-or-group>,<another-user-or-group>,...)

  Specifies the list of users and groups that are granted the permission to work with this area. Applies only to areas of type WORK. If /ADD is specified, the specified users are appended to the area's user list. If /DELETE is specified, the specified users are deleted from the area's user list. If neither /ADD nor /DELETE is specified, the specified list of users replaces the area's user list.

- /USER=<user-name> [/PASSWORD=<password>]

  Login information for the operating system user account that will own files transferred into the area.

- /TRANSFER_SCRIPTS=<script-set>

  Applicable only to the DEPLOYMENT area type. Specifies the transfer script set. The script set contains a comma-separated list of the names of pre/post/fail transfer scripts in the following format:

  (<pre-script>,<post-script>,<fail-script>)

  If one of the scripts is undefined, CA uses $NONE as a placeholder. The pre-script is executed before items are transferred into an area, the post-script is executed after successful transfer of all items into an area, and the fail script is executed after a failed transfer of all items into an area.

- /OWNER=<user-name> or <group-name>

  Optional. Specifies the user or group that is to become the owner of the new area. If /OWNER is not specified, the user who created the area is set as its owner.

- /STATUS=ONLINE or OFFLINE

  Applicable only to the DEPLOYMENT area type. Specifies the status of the area. If the area's status is ONLINE, the area may participate in file transfer operations. If the area's status is OFFLINE, the area is automatically excluded from any file transfer operations. If this qualifier is not specified, an area with status ONLINE is created.

- /FILTER=<area-filter>

  Applicable only to the DEPLOYMENT area type. Specifies the name of the area filter to be used when deploying files into this area.

Description

The UA command updates an area definition.

If an area is in use (that is, associated with a project), /NETWORK_NODE, /DIRECTORY, /TYPE, and /STAGE may not be updated. If an area is not in use, changing /NETWORK_NODE or /DIRECTORY *does not* physically transfer files from the old location to the new location.

Constraints

To update a work area, you must have the Update Work Area Properties privilege. To update a deployment area, you must have the Update Deployment Area Properties privilege. These privileges are automatically granted to the owner of the area.

UBA – Update Build Area

NOTE This command is no longer available; use UA (Update Area) instead.

See the UA command.

UBDB – Update an Existing Base Database Entry

```
/BDB_NAME=<base_db_name>
/DB_SERVICE=<base_db_instance>
/NN_NAME=<network_node_name>
[/SITE_NO=<site_no>]
```

This command enables you to edit registered base database entries in an installation's network administration tables. See the *Administrator's Guide* for details.

UBLA – Update Baseline Attributes

NOTE This command is not available in the Issue Management–only licensed version of Dimensions.

```
<baseline-spec>
[/ATTRIBUTES=(<attr1>=<value1>, <attr2>=<value2>,…)]
[/CHILD_ORDER=(<collection-list>)]
```

Example

```
UBLA PROD:"R M VERSION 2 FOR HP" -
/ATTRIBUTES=(TESTED_BY=GROUP5, AUTH_CODE=542)
```

- `<baseline-spec>` comprises:

  `<product-id>:<baseline-id>`

- `/ATTRIBUTES=(<attr1>=<value1>,<attr2>=<value2>,…)`

  `<attrN>` is the Variable Name defined for one of the user-defined attributes for the baseline's type, which has also been declared as usable for this `<product-id>` and baseline's type.

  `<valueN>` is the substitution value to be given to this attribute.

  To add a new value to an existing multivalue attribute, use the following syntax:

  `/ATTRIBUTES=(<attr1>+=["<value1>"])`

  For example, to add "Charlie" to the multivalue attribute "NAME":

  `/ATTRIBUTES=(NAME+=["Charlie"])`

NOTE For full details about how to use the /ATTRIBUTES qualifier to append or prepend values to an existing multivalue attribute, see the UIA command on page 366.

- `/CHILD_ORDER=(<collection-list>)`

  This qualifier is used to establish a search order among child collections. This order is used to resolve what should be fetched when the paths in the child collections specify conflicting physical locations. The first collection in the list where a match occurs takes precedence.

Description

Subject to user authorization, each of the specified attributes is updated to the value given; or, if any of these attributes had no value previously set for this baseline, it is now set with the value given.

NOTE In the other commands (CBL, CMB and CRB) that assign values to user-defined baseline attributes, the `/ATTRIBUTES` qualifier is shown as optional. But it cannot be omitted if there exist any user-defined attributes, applicable to this baseline type, **whose Mandatory flag is Y, and for which there is no Default Value**.

Constraints

This command can be run in accordance with the attribute update rules defined by a user with the appropriate management privileges.

UBPROJ – Update a Dimensions Build Project

NOTE This command is no longer available. Use Dimensions Build to manage build projects.

```
Command no longer available.
```

UC – Update Request

```
<request-id>
[/ATTRIBUTES=(<attr1>=<value1>,<attr2>=<value2>,…)]
[/DESCRIPTION=<desc-file>] and/or [/ADD_DESCRIPTION or
    /EDIT_ACTION_DESCRIPTIONS or /CANCEL_EDIT]
[/ATTACHMENTS=
([FILENAME=<file-id>, DESCRIPTION=<description-text>], …)]
[/ADD_ATTACHMENTS=
([FILENAME=<file-id>, USER_FILE=<user-file>, DESCRIPTION=<description-
    text>], …)]
[/DELETE_ATTACHMENTS=([FILENAME=<file-id>], …)]
[/DETAILED_DESCRIPTION=<desc-file>]
[/EDIT_DETAILED_DESCRIPTION]
[/[NO]EXCLUSIVE_LOCK]
```

NOTE This command only functions for pending or held requests. Additionally, it cannot be run from Dimensions for z/OS.

Example

```
UC PROD_DR_28 -
/ATTRIB=(TITLE="QREL Subdir format problem")
/ATTACHMENTS=([/FILENAME=Figure3.jpg, DESCRIPTION="updated description
    text"])
/ADD_ATTACHMENTS=([/USER_FILE=c:\temp\newfile.jpg, DESCRIPTION="new
    page image"]
/FILENAME=Figure7.jpg])
/DELETE_ATTACHMENTS=([FILENAME=Figure1.jpg])
```

- `<request-id>`

  This is the identity of the request to be modified.

NOTE All the following qualifiers and parameters are optional, but **only** one of them must be specified.

The qualifiers /ATTRIBUTES, /DESCRIPTION, /ADD_DESCRIPTION, /EDIT_ACTION_DESCRIPTIONS and /CANCEL_EDIT are mutually exclusive i.e. you cannot submit a UC command with more than one.

- `/ATTRIBUTES=(<attr1>=<value1>,<attr2>=<value2>,…)`

  `<attrN>` is the Variable Name defined for one of the user-defined attributes for requests, which has also been declared as usable for the product and type specified in `<request-id>`.

  `<valueN>` is the new **substitution** value to be given to this attribute.

- `/ATTRIBUTES=(<attr1>+=<value1>,)`
- `/ATTRIBUTES=(<attr1>++=<value1>,)`

NOTE For details about how to use the above /ATTRIBUTES qualifiers to append or prepend values to an existing multivalue attribute, see the UIA command on page 366.

- /DESCRIPTION=<desc-file>

  Specifies a file containing the text body to be used as:

  - the detailed description of the request, if it is currently held; or
  - an action description if it has been saved (entered into system).

- /ADD_DESCRIPTION

  calls an editor for the user to edit (or enter, if <desc-file> is omitted) the detailed description of the request (if it is held) or an action description (if it is saved).

NOTE Must *not* be specified if you wish to run UC from Dimensions for z/OS.

- /EDIT_ACTION_DESCRIPTIONS

  calls an editor to allow a user with a leader role to edit all the action descriptions entered since the request was last actioned.

NOTE Must *not* be specified if you wish to run UC from Dimensions for z/OS.

- /CANCEL_EDIT

  undoes the effects of a failed edit of a request.

- /ATTACHMENTS=([FILENAME=<file-id>, DESCRIPTION=<descriptiontext>],…)

  changes the description of an existing attachment.

FILENAME=<file-id>	Specifies the name of the attachment.
DESCRIPTION=<descriptiontext>	is the description of the attachment.

- [/ADD_ATTACHMENTS=([FILENAME=<file-id>, USER_FILE=<user-file>, DESCRIPTION=<description-text>], …)]

  Adds a new attached file.

FILENAME=<file-id>	Specifies the file to be attached.
USER_FILE=<user-file>	Specifies the user file from where the attachment is to be loaded.
DESCRIPTION=<descriptiontext>	is the description of the attachment.

NOTE The FILENAME parameter must be unique on the request. The DESCRIPTION parameter is generated automatically if you omit it.

- /DELETE_ATTACHMENTS=([FILENAME=<file-id>],…)]

  deletes an existing attached file. The FILENAME parameter must identify an existing attachment.

- /DETAILED_DESCRIPTION=<desc-file>

allows a user (subject to constraints below) to replace a request's detailed description with the contents of file <desc-file>. You cannot chose this option together with /EDIT_DETAILED_DESCRIPTION i.e. they are mutually exclusive.

- /EDIT_DETAILED_DESCRIPTION

  calls an editor to allow a user (subject to constraints below) to edit the request's detailed description. In UNIX the interactive editor is specified by the setting of the symbol DM_CHD_EDT or DM_CHD_EDT_SCRIPT. You cannot chose this option together with /DETAILED_DESCRIPTION=<desc-file> i.e. they are mutually exclusive.

- /EXCLUSIVE_LOCK

  Specifies that the new request will be "locked" against any request (issue) replication "requests" from users located on other replication sites. A locked request is still available for users to work on normally if they are located on the "owning" replication site.

  There are primarily two conceptual working models that are used to provide issue replication—the delegation model and the request model. The delegation model works on the assumption that requests are created on one site and then "delegated" to another site to work on; while the request model follows the principle that a user on any site who sees a request that they want to work on can "request" that the responsibility for that request is handed over to them. See the *Administrator's Guide* for details of issue replication.

  The default is /NOEXCLUSIVE_LOCK meaning that the new request can be "requested" from any authorized replication site.

Constraints

This command can be run only by a user with the appropriate management privileges or by users who have the minimum role to action the request for the current state to the next state. Your edit is, however, also subject to any update rules set by a user with the appropriate management privileges.

Requests that were created in a held state are not considered to have been "created" as far as Process Models where optional sensitive states or attributes have been set up ("electronic signatures") are concerned. The act of entering them into the system by actioning them out of the held state is considered the "authorization point" for such process models. This also applies to held requests that are updated at the held state (using the command UC) before being actioned on.

UCO – Update an Existing Contact

```
/CO_NAME=<contact_name>
```

This command enables you to update an installation codeset. See the *Administrator's Guide* for details.

UCST – Update an Existing Codeset

```
/CDST_NUMBER=<codeset_number>
[DESCRIPTION=<description>]
```

This command enables you to update an installation's codeset. See the *Administrator's Guide* for details.

UCU – Update Customer

NOTE This command is not available in the Issue Management–only licensed version of Dimensions.

```
<name>
/LOCATION=<location>
/PROJECT=<project-spec>
[/COMMENT=<comment>]
[/CONTACT=<contact-details]
[/NEW_LOCATION=<location>]
[/NEW_NAME=<name]
[/NEW_PROJECT=<project-spec>]
```

Example

```
UCU "Brown Finances" /LOCATION="Manchester" -
/PROJECT="PAYROLL" /CONTACT="Mrs E Green" -
/NEW_LOCATION="Bristol"
```

- `<name>`

  Specifies the **present** name for the customer details you wish to update.

- `/LOCATION=<location>`

  Specifies the **present** physical location for the customer details you wish to update.

- `/PROJECT=<project-spec>`

  Specifies the **present** project name for the customer details you wish to update.

- `/COMMENT=<comment>`

  for optionally customer contact details.

- `/CONTACT=<contact-details>`

  for optionally adding the new customer contact details.

- `/NEW_LOCATION=<location>`

  Specifies the customer's **updated** physical location.

- `/NEW_NAME=<name>`

  Specifies a **updated** name for the customer.

- `/NEW_PROJECT=<project-spec>`

  Specifies the **updated** project name for the customer details you wish to update.

Description

The Dimensions product allows you to maintain a list of customers and a record of which Dimensions releases have been sent to each customer.

The UCU command enables you to update a customer's details.

Constraints

The combination of customer name, location, and project-spec must be unique in the Dimensions database.

If any releases are related to a customer, you can only edit the contact and location information.

UFS – Edit an Existing File System

```
/FS_NAME=<file_system_name>
[DESCRIPTION=<description>]
```

This command enables you to edit specific file systems definitions for each registered installation operating system. See the *Administrator's Guide* for details.

UGRP – Update Group

```
<group-name>
/DESCRIPTION="<description>"
```

Description

This command updates a group's properties. `<group-name>` is the name of the group, and `<description>` is the group's description to be updated.

Constraints

Only users with the appropriate management privileges can run this command.

UI – Update Item

NOTE This command is not available in the Issue Management–only licensed version of Dimensions.

```
<item-spec>
[/ROOT_PROJECT=<project-spec>]
[/FILENAME=<file-name>]
[/USER_FILENAME=<user-filename>]
[/[NO]KEEP]
[/REVISION=<new-revision>]
[/ATTRIBUTES=(<attr1>=<value1>,<attr2>=<value2>,…)]
[/CHANGE_DOC_IDS=(<request1>,<request2>,…)]
[/STATUS=<status>]
[/COMMENT=<comment text>]
[/WORKSET=<project-spec>]
[/[NO]FORCE_UPDATE]
[/CODEPAGE=<code-page>|DEFAULT]
[/CONTENT_ENCODING=<file-encoding>]
[/NOMETADATA]
```

Example

```
UI PROD:"QUERY RELEASE".AAAA-SRC;1 /KEEP -
/CHANGE=(PROD_DC_16, PROD_DR_8) /STAT="UNDER TEST" -
/COMMENT="updated for ERB 58"
```

- `<item-spec>` comprises:

  `<product-id>:<item-id>.<variant>-<item-type>;<revision>`

`<item-id>`	may be omitted if `<file-name>` is specified.
`<variant>`	may be omitted if only one exists.
`<revision>`	defaults to the latest revision in the project specified by /WORKSET. If /WORKSET is unspecified, the user's default project will be assumed.

- `/ROOT_PROJECT=<project-spec>`

  Comprises:

  `<product-id>:<project-id>`

  This optionally specifies the root project. Use this when the current project set via SCWS (or the project specified by the /WORKSET qualifier) occurs in more than one project tree.

- `/FILENAME=<file-name>`

  Specifies the name of the project file name. If /ROOT_PROJECT is used to specify a the root project, /FILENAME is interpreted in the scope of that project.

  The project file name identifies the relative path (directory plus file name) from the working location to the item to be used from the current project. The project file name for the same item may *differ* between projects; for example, `src/hello.c`, `hello.c`, or `src/build/hello.c`.

It may be omitted if <item-id> is specified.

- /USER_FILENAME=<user-filename>

  Specifies the name of the file holding the item in the user area.

  If omitted, it defaults to <file-name> – i.e. the file in the user area (the current directory) has the same name as that of the item's file in the item library.

- /KEEP

  Specifies that the user area file, which is normally deleted once its data has been placed under Dimensions control, is to be left intact.

- /REVISION=<new-revision>

  Specifies a new revision for the item. If /WORKSET is specified, the new revision will be placed in that project; otherwise, the new revision will be placed in the user's current default project.

  If new revision is omitted, Dimensions increments the current revision (the rightmost sub-field only), unless the item revision in <item-spec> is at its initial lifecycle state. In this case, the revision is unchanged.

- /ATTRIBUTES=(<attr1>=<value1>,<attr2>=<value2>,…)

<attrN>	is the Variable Name defined for one of the user-defined attributes for items, which has also been declared usable for the <product-id> and <item-type> specified in <item-spec>.
<valueN>	is the value to be given to this attribute.

- </CHANGE_DOC_IDS=(<request1>,<request2>,…)

<requestN>	identifies a request to which the new item-revision is to be related **in-Response-to**.

- /STATUS=<status>

  Specifies the status of the new item-revision.

NOTE The status, if specified, must be one which would be valid if AI had been used separately. If omitted, the initial state (in the lifecycle defined for <item-type>) is assigned.

- /COMMENT=<comment text>

  comment text to explain the reason for the update of this item revision. The comment text can be up to 1978 characters long, and can be made available within the item header.

- /WORKSET=<project-spec> comprises:

  <product-id>:<project-id>

  and is optional.

  If specified, the new item revision will be placed in that project.

  If unspecified, Dimensions will place the new item revision in the user's current project.

- /FORCE_UPDATE
- If the checksum is enabled for the item type and the file checked in has not been modified, the check in will succeed only if this qualifier is used, otherwise it will fail.CODEPAGE=<code-page>|DEFAULT

  Specifies the *code page* to be associated with the item. The code page defines the method of encoding characters. It encompasses both the different ways characters are encoded on different platforms (EBCDIC on z/OS and ASCII on Windows and UNIX) and differences between human languages. Every item in Dimensions has a code page associated with it, this being defined or derived for the connection setting or an individual item.

  The /CODEPAGE parameter defaults to the code page specified when the connection between the database server and the logical node on which the user file resides was created. Whenever the item moves between platforms, for example, on a check-out from the mainframe to a PC, if the code page for the target platform is different to the item's code page, Dimensions automatically converts the item.
  /CODEPAGE is relevant only for text files, because whenever a text file is checked out or gotten, it must be in the right code page for the target platform, so that it displays correctly. Binary files are moved between platforms with no conversion.

  For further details concerning code pages and logical nodes see the *Distributed Development Guide*.

  You are advised to let the parameter default to the code page for the item type or the platform.

  The /CODEPAGE options available are:

<code-page>	Specify one of the code page values listed in the text file codepage.txt, located on your Dimensions server in the codepage subdirectory of the Dimensions installation directory. This file also provides more information about translation between code pages.
DEFAULT	Use the code page specified for the target node connection.

- <file-encoding>

  Specifies the content encoding for new item revisions to be created. Supported encodings are the Microsoft codepages, the ISO-8859 variants (1–10), UTF-8, UTF-16, UTF-16BE, UTF-16LE, UTF32, UTF32BE, and UTF32LE.

- /NOMETADATA

  This parameter disables creation and usage of metadata files in the local work area.

Description

You use the UI command when you want to create a new item revision using the contents of a file in your working directory. Updating an item is similar to using the Edit command in that you do not need to check out the item first. When you update an item, the revision ID is changed according to your process model rules.

Your process model may require you to relate a request to the updated item.

If local metadata is present, it is used to update the correct revision (the revision originally fetched by the user). If the user specifies an explicit revision to update that differs from the revision originally fetched by the user, a warning is generated, but the operation succeeds. After a successful update in which /KEEP is specified, the local metadata is updated.

/NOKEEP causes the local metadata to be deleted as well as the real file.

If the original file was fetched with item header substitution turned on (and some substitutions performed), the UI command produces a warning message and fails.

If the original revision fetched does not exist in the current project, the UI operation fails with an error message.

NOTE UI results in the creation of a new revision in the project. If DEVELOPMENT deployment areas are in use, this command automatically updates the corresponding areas.

Constraints

This command can be run only by users who have one of the roles required to action the item from the initial lifecycle state to a new state.

IMPORTANT! The maximum size of any item in the database is 2GB.

UIA – Update Item Attributes

NOTE This command is not available in the Issue Management–only licensed version of Dimensions.

```
<item-spec>
[/FILENAME=<file-name>]

[/ATTRIBUTES=(<attr1>=<value1>, <attr2>=<value2>, …)]
–or–
[/ATTRIBUTES=(<attr1>+=<value1>,)]
–or–
[/ATTRIBUTES=(<attr1>++=<value1>, …)]

[/COMMENT=<comment text>]
[/WORKSET=<project-spec>]
[/FORMAT=<format>]
[/DESCRIPTION=<item-description>]
[/ORIGINATOR=<Dimensions-user>]
```

Examples

```
UIA PROD:"QUERY RELEASE".AAAA-SRC;1 -
    /ATTRIB=(DELIVERY_DATE=10-JUN-1998, AUTH_CODE=542) -
    /COMMENT="updated for ERB 58a"
UIA PAYROLL:"FORM1 FRM".AAAA-SRC;win2000#1 -
    /ATTRIB=(DEPLOY_ID+=["1.0"],)
UIA PAYROLL:"FORM1 FRM".AAAA-SRC;win2000#1 -
    /ATTRIB=(DEPLOY_ID+=["1.1"],["1.2"])
UIA PAYROLL:"FORM1 FRM".AAAA-SRC;win2000#1 -
    /ATTRIB=(DEPLOY_ID++=["1.0"],)
UIA PAYROLL:"FORM1 FRM".AAAA-SRC;win2000#1 -
    /ATTRIB=(DEPLOY_ID++=["1.2"],["1.1"])
```

- `<item-spec>` comprises:

  `<product-id>:<item-id>.<variant>– <item-type>;<revision>`

  `<item-id>` may be omitted if `<file-name>` is specified.

  `<variant>` may be omitted if only one exists.

  `<revision>` defaults to the latest revision (see Introduction on page 16).

- `/FILENAME=<file-name>`

  Specifies the name of the file holding the item in the item-library.

  It may be omitted if `<item-id>` is specified.

- /ATTRIBUTES=(<attr1>=<value1>,<attr2>=<value2>,…)

  <attrN> is the Variable Name defined for one of the user-defined single-valued attributes for items, which has also been declared usable for the <product-id> and <item-type> specified in **<item-spec>**. There are a total of 220 (Oracle) or 56 (DB2 or SQL Server) attributes that can be declared for all single-valued and multivalued attributes.

  <valueN> is the substitution value to be given to this attribute.

- /ATTRIBUTES=(<attr1>+=<value1>,)

  <attrN> is the Variable Name defined user-defined multivalue attributes for items, which has also been declared usable for the <product-id> and <item-type> specified in **<item-spec>**. There are a total of 220 (Oracle) or 56 (DB2 or SQL Server) attributes that can be declared for all single-valued and multivalued attributes.

  The '+' syntax enables you to append multivalue attributes. In the example of a pair of such UIA commands given earlier, DEPLOY_ID will have the values "1.1" and "1.2" appended to "1.0". Therefore, after executing the pair of commands, the item DEPLOY_ID will have values in the following order

  1.0
  1.1
  1.2

  If you attempt to use the '+' syntax on a single value attribute, you will receive the error message "Error: Attribute <ID> cannot be appended / prepended as it is not a Multi Value attribute".

  <valueN> is the substitution value to be given to this attribute.

- /ATTRIBUTES=(<attr1>++=<value1>,)

  <attrN> is the Variable Name defined user-defined multivalue attributes for items, which has also been declared usable for the <product-id> and <item-type> specified in **<item-spec>**. There are a total of 220 (Oracle) or 56 (DB2 or SQL Server) attributes that can be declared for all single valued and multi valued attributes.

The '++' syntax enables you to prepend multi value attributes. In the example of a pair of such UIA commands given earlier, DEPLOY_ID will have the values "1.2" and "1.1" prepended to "1.0". Therefore, after executing the pair of commands, the item DEPLOY_ID will have values in the following order

1.2

1.1

1.0

If you attempt to use the '++' syntax on a single value attribute, you will receive the error message "Error: Attribute <ID> cannot be appended / prepended as it is not a Multi Value attribute".

`<valueN>` is the substitution value to be given to this attribute.

- `/COMMENT=<comment text>`

  comment text to explain the reason for the update of this item attributes. The comment text can be up to 1978 characters long.

- `/WORKSET=<project-spec> comprises:`

  `<product-id>:<project-id>`

  This optionally specifies the project to be used for this command: failing this, the user's current project will be taken.

  Item revisions to be affected by the command may be specified explicitly, or they will be selected from the project.

- `/FORMAT=<format>`

  Specifies the item data format. You use this qualifier to modify an item's format from, for example, that with which it was created. If item data formats have been assigned with the ADF command, the format specified must be one of those in the valid list of formats.

NOTE Only the PRODUCT-MANAGER can modify a FORMAT.

- `/DESCRIPTION=<item-description>`

NOTE descriptive name for itemOnly the PRODUCT-MANAGER can modify a DESCRIPTION.

- `/ORIGINATOR=<Dimensions-user>`

  Specifies a new Dimensions-user to be treated as the "originator" of the item. This qualifier is for use in scenarios where the historical originator (whose name will remain in the item's history log) is no longer a Dimensions user or is no longer actively involved in the project concerned. From now on, whenever the original originator would have had the item appear in their pending list or have had received

e-mail, the "new" originator will become the originator as far as Dimensions is concerned.

NOTE Only the PRODUCT-MANAGER can modify an ORIGINATOR.

Description

Subject to the Constraints below, each of the specified attributes is updated to the value given; or, if any of these attributes had no value previously set for this item revision, it is now set with the value given. (It is not possible to unset an attribute, once it has been set for that revision.)

Attribute values for other revisions of the same item remain unaffected, except for any attribute(s) specified here whose All Revisions flag is **Y**. Such attributes are updated, or defined, with the value given here for all those other revisions as well.

NOTE In the other commands (CI, EI and UI) that assign values to user-defined item attributes, the `/ATTRIBUTES` qualifier is shown as optional. However, it cannot be omitted if the command is creating a new item or revision, and there exist any user-defined attributes, applicable to the item-type of the new revision, **whose Mandatory flag is Y, and for which there is no Default Value**. (Attributes with **Y** also for All Revisions do, in effect, always have a default value for all revisions after the first.)

Constraints

- UIA can—in all circumstances—be run only in accordance with the attribute update rules set by a user with the appropriate management privileges. With the above proviso, UIA can then be run by users who have, for each attribute specified, a role that is compatible with the value of the Role Check parameter. If the attribute update rules are defined for the item revision, UIA can be run by users who have, for each attribute specified, the role required by the update rule for that attribute. If there are no attribute update rules for the item revision, the item must be in the user's pending list or the user must have the appropriate management privileges.
- The value specified for each attribute must be consistent with its `Data_type` parameter.
- Each attribute must be one which has **I** for items as the `Scope` flag, and if a specific item-type is set for the attribute, the `<item-type>` specified in `<item-spec>` must match it.
- Any attribute whose Updateable flag is **N** cannot be specified in this UIA command. Values can be assigned to such attributes for a revision, only by the command which creates it (and only by the CI command if in addition the All Revisions flag is **Y**).
- If an attribute's Visible flag is **N**, no value can be assigned to it by this or any other standard Dimensions function.

UINS – Update an Existing Database Instance Entry

```
/DB_SERVICE=<base_db_instance>
/NN_NAME=<network_node_name>
```

This command enables you to edit registered database instance entries in an installation's network administration tables. See the *Administrator's Guide* for details.

ULCA – Update Library Cache Area

NOTE This command is not available in the Issue Management–only licensed version of Dimensions.

```
<area-name>
/NEW_NAME=<new-name>
[/DESCRIPTION=<area-description>]
[/NETWORK_NODE=<node-name>]
[/DIRECTORY=<HLQ/directory>]
[/USER=<user-name> [/PASSWORD=<password>]]
[/OWNER=<user-name> or <group-name>]
[/STATUS=ONLINE or OFFLINE]
```

Example `ULCA <area-name> /NETWORK_NODE=<host-machine> /DIRECTORY=<area-directory>`

Parameters

- `<area-name>`

  Specifies the name of the area. Area names must be unique within the base database.

- `<new-name>`

  Specifies the new name for the area.

- `/DESCRIPTION=<description>`

  Specifies a description for the new area.

- `/NETWORK_NODE=<node-name>`

  Specifies the machine hosting the area.

- `/DIRECTORY=<HLQ/directory>`

  Specifies the directory, or PDS (partitioned data set), where the area is located.

  HLQ is a high-level qualifier; for example, MERVK.WORK. It is a common prefix for all data sets in the area such as MERVK.WORK.C, MERVK.WORK.CBL, and so forth.

- `/USER=<user-name> [/PASSWORD=<password>]`

  Login information for the operating system user account that will own files transferred into the area.

- `/OWNER=<user-name> or <group-name>`

  Optional. Specifies the user or group that is to become the owner of the new area. If /OWNER is not specified, the user who created the area is set as its owner.

- `/STATUS=ONLINE or OFFLINE`

  Specifies the status of the area. If the area's status is ONLINE, the area may participate in file transfer operations. If the area's status is OFFLINE, the area is automatically excluded from any file transfer operations. If this qualifier is not specified, an area with status ONLINE is created.

Description

The ULCA command updates a library cache area definition.

If an area is in use (that is, associated with a project), /NETWORK_NODE and /DIRECTORY cannot be updated. If an area is not in use, changing /NETWORK_NODE or /DIRECTORY *does not* physically transfer files from the old location to the new location.

Constraints

To update a library cache area, you must have the Update Library Cache Area Properties privilege. This privilege is automatically granted to the owner of the library cache area.

ULCK – Unlock Project

NOTE This command is not available in the Issue Management–only licensed version of Dimensions.

```
workset <project-spec>
```

Example

- `ULCK WORKSET PROD_X:TEST_WSworkset-spec> comprises:`

  `<product-id>:<project-id>`

  The specified project must exist.

Description

This command unlocks the project with project as a fixed parameter and `<project-spec>` a user-defined parameter. The unlocked state allows new Dimensions items to be added to the project.

Constraints

This command can be run only by a user with the appropriate management privileges for the project concerned.

UMNR – Update Mail Notification Rule

```
<rule-id>
/DISPLAY_TEXT=<display-text>
[/DESCRIPTION=<description>]
[/[NO]PRIVATE]
[/PRE_STATUS=<pre-status>]
[/POST_STATUS=<post-status>]
[/PRE_ATTRIBUTES=(<pre-attrib1>,<pre-attrrib2>,…)]
[/POST_ATTRIBUTES=(<post-attrib1>,<post-attrib2>,…)]
```

Example `UMNR <rule-id> /DISPLAY_TEXT="display text"`

Parameters

- `<rule-id>`

  is the name of the rule to be updated.

- `<display-text>`

  is the text that will be displayed to the user updating the rule.

- `<description>`

  is the description of the rule.

- `/PRIVATE or /NOPRIVATE`

  If a notification rule is private, only the user who created the rule can see it and subscribe to it. If the rule is public, any user can subscribe to the rule and, if they have the correct privilege, the user can also update the rule.

- `<pre-status>`

  is the pre-status to be used to filter the rule.

- `<post-status>`

  is the post-status to be used to filter the rule.

- `(<pre-attrib1>,<pre-attrrib2>,…)`

  is the list of pre-attributes to be used to filter the rule.

- `(<post-attrib1>,<post-attrib2>,…)`

  is the list of post-attributes to be used to filter the rule.

Description

The UMNR command updates a mail notification rule.

Constraints

Only users with the appropriate management privileges can run this command.

UNC – Update an Existing Network Node Connection

```
/SERVER_NAME=<server_node_name>
/CLIENT_NAME=<client_node_name>
/CDST_NUMBER=<codeset_number>
/NWO_NAME=<network_object_name>
[/DIRECT_FILE_COPY]
[/FILE_COMPRESSION]
```

This command enables you to edit an existing installation network node connection. For details, see the *Administrator's Guide*.

UNN – Update an Existing Network Node

```
/NN_NAME=<network_node_name>
```

This command enables you to edit an existing installation network node. See the *Administrator's Guide* for details.

UNWO – Update an Existing Network Object

`/NWO_NAME=<network_object_name>`

This command enables you to edit an existing installation network object. See the *Administrator's Guide* for details.

UOS – Update an Existing Operating System

```
/OS_NAME=<operating_system_name>
```

This command enables you to edit an existing installation operating system. See the *Administrator's Guide* for details.

UP – Update Design Part PCS

```
<part-spec>
/NEW_PCS=<new-pcs>
[/DESCRIPTION=<description>]
[/ATTRIBUTES=(<attr1>=<value1>,<attr2>=<value2>,…)]
```

Example

```
UP PROD:"RELEASE MANAGEMENT".AAAA /NEW_PCS=1A -
/DESC="Release Support - Sun Test Version"
```

- `<part-spec>`

  Comprises:

  `<product-id>:<part-id>.<variant>;<pcs>`

  `<variant>` may be omitted if only one exists.

  `<pcs>` is ignored. On completion of this command, what is now the current PCS will become CLOSED.

- `/NEW_PCS=<new-pcs>`

  Specifies the new PCS of the design part, to be OPENed and become the current PCS.

- `/DESCRIPTION=<description>`

  This is a text description that applies to every PCS in the design part or design part variant. You can update the description for every PCS in the design part or design part variant; however you cannot define a unique description for each PCS.

- `/ATTRIBUTES=(<attr1>=<value1>,<attr2>=<value2>,…)`

  `<attr1>` is the Variable Name defined for one of the user-defined attributes for design parts, which has also been declared as usable for this `<product-id>` and design part's category.

  `<valueN>` is the substitution value to be given to this attribute for the new PCS only.

Constraints

This command can be run only by the user who has the appropriate management privileges for the design part to which the variant being updated belongs.

UPA – Update Part Attributes

```
<part-spec>
[/ATTRIBUTES=(<attr1>=<value1>, <attr2>=<value2>,…)] or
    [/DESCRIPTION=<part-description>]
[/ORIGINATOR=<part-creator>]
```

Examples

```
UPA PROD:"ITEM OPS".AAAA;5
    /ATTRIBUTES=(TESTED_BY=GROUP5, AUTH_CODE=542)"

UPA PROD:"ITEM OPS".AAAA;5
    /DESCRIPTION="LIMITED TO GROUP 5 WITH CODE 542"
```

- `<part-spec>` comprises:

  `<product-id>:<part-id>.<variant>;<pcs>`

`<variant>`	may be omitted if only one exists.
`<pcs>`	is ignored. Only the current PCS may be updated.

- `/ATTRIBUTES=(<attr1>=<value1>,<attr2>=<value2>,…)`

`<attrN>`	is the Variable Name defined for one of the user-defined attributes for design parts, which has also been declared as usable for this `<product-id>` and design part's category.
`<valueN>`	is the substitution value to be given to this attribute.

- `/DESCRIPTION=<part-description>`

  If omitted, the original description is retained.

  /ATTRIBUTES and /DESCRIPTION are mutually exclusive.

- `/ORIGINATOR=<part-creator>`

  Changes who created the part.

Description

Subject to user authorization, each of the specified attributes is updated to the value given; or, if any of these attributes had no value previously set for this design part PCS, it is now set with the value given. (It is not possible to unset an attribute, once it has been set for that PCS.)

Attribute values for the earlier, closed PCSs of the same design part variant are never altered.

NOTE In the other commands (CP, CPV and UP) that assign values to user-defined design part attributes, the `/ATTRIBUTES` qualifier is shown as optional. However, it cannot be omitted if there exist any user-defined attributes, applicable to this design part's category, **whose Mandatory flag is Y, and for which there is no Default Value**.

Constraints

Only users with the appropriate management privileges can run this command.

UPLOAD – Upload Local File or Directory

NOTE This command is not available in the Issue Management–only licensed version of Dimensions.

```
[<file-spec> or /DIRECTORY=<directory-spec>] or
    /USER_FILELIST=<filelist-file>]
[/BRANCH or /FORCE_TIP]
[/FORCE_CHECKIN]
[/[NO]KEEP]
[/[NO]RECURSIVE]
[/[NO]RESTRICTED]
[/LOGFILE=<file-spec> or /SCRIPTFILE=<file-spec>]
[/ATTRIBUTES=(<name>=<value>, ...)]
[/CHANGE_DOC_IDS=(<id-spec>, ...)]
[/CODEPAGE=<code-page> or DEFAULT]
[/COMMENT=<text>]
[/DESCRIPTION=<description>]
[/PART=<part-spec>]
[/WORKSET=<project-spec>]
[/[NO]CONFLICT_CHECK]
[/FILTER=<filter-name>]
[/CONTENT_ENCODING=<file-encoding>]
```

Examples

```
UPLOAD "C:\temp\work\FooBar.java" /COMMENT="Fixed a bug"
    /ATTRIBUTES=(Complexity="High") /NOKEEP
```

This command will upload the locally modified file `FooBar.java`. If any conflicting revisions are found in the repository, they will be reported and the upload will fail. The newly created revision will have the comment "Fixed a bug", and the revision's Complexity attribute will be set to "High".

```
UPLOAD /DIRECTORY="C:\temp\work" /COMMENT="Finished refactoring"
    /BRANCH /CHANGE_DOC_IDS=(PAYROLL_TDR_2)
```

All files found in the directory `C:\temp\work` and in any directories below it will be considered for upload. If the user has locally modified any file or explicitly checked out any file in that directory, it will be checked in. Checked-in revisions will be placed on a branch (no merge will occur) and be related to PAYROLL_TDR_2.

Parameters

- `<file-spec>`

  This parameter specifies the name of the file to be uploaded. (The Dimensions `node::` syntax is also valid.)

- `<directory-spec>`

  This parameter specifies a directory path. The files in the directory are enumerated, and each one that has been been modified is uploaded.

  You can specify the directory using the Dimensions `node::` syntax.

- `<filelist-file>`

This optional qualifier must specify a file containing a list of file names to be downloaded form the project or baseline. Each file name must be on a separate line.

File names may be specified as either relative or absolute paths. If the path is absolute, it is interpreted as a full project or baseline file path; otherwise, Dimensions obtains the project or baseline path by mapping the file name to the operation root directory, which is the current working location as specified by the last SCWS command. If such a mapping is not possible, the file name is ignored.

- /BRANCH

  This qualifier specifies that if any conflicts occur, modified files will be uploaded to form a branch. No merge will take place.

- /FORCE_TIP

  This qualifier specifies that if any conflicts occur, modified files will be uploaded to form the "tip" (latest) version of the file. No merge will take place and there is the potential for other conflicting changes to be "hidden".

- /FORCE_CHECKIN

  This qualifier specifies that if there are any local files without Dimensions metadata that correspond to existing items in the repository, then these files will be upload to the latest version of the items. No merge will take place and there is potential for conflicting changes. By default, /FORCE_CHECKIN is not specified and these files are skipped. The UPLOAD command issues a warning for each such file, in this case.

- /NOKEEP

  If this qualifier is specified, the modified files will be removed from the local workspace after the upload completes. If /NOKEEP is not specified (or if /KEEP is specified), the local files will remain after the upload.

- /NORECURSIVE

  If /DIRECTORY is specified and this qualifier is not present, all files that have been modified in all directories beneath the one specified are uploaded. /NORECURSIVE specifies that only files at the specified directory level are uploaded.

- /RESTRICTED

  Specifies that UPLOAD will run in "restricted" mode and will not create new project items or project directories. Only checked out or locally modified item revisions will be updated. By default, UPLOAD runs in unrestricted mode, creating new directories and items as needed.

- /LOGFILE=<file-spec>

  This qualifier specifies that a log file be generated at the given file location containing the results of all the individual Dimensions operations executed through this command.

- /SCRIPTFILE=<file-spec>

  This qualifier specifies that a script file be generated at the given file location containing the individual Dimensions operations that would otherwise be executed through this command. The operations are not executed.

- <name>=<value>

  The user defined attributes to set on the newly created revisions. All attributes specified must be valid for the item types created.

- `<id-spec>`

  The requests for the items to be related to. The originally fetched versions will be related as "Affected", and the newly created versions will be "In-Response To".

- `<code-page>`

  The code page to be associated with the item.

- `<text>`

  The comment to apply to all of the newly created item revisions.

- `<description>`

  The description to apply to all the newly created items.

- `<part-spec>`

  Design part specification in the form:

  ```
  <product-id>:<part-id>.<variant>;<pcs>
  ```

- `<project-spec>`

  The project to which to upload the files. If this parameter is not specified, files are uploaded to the current session project.

- `/CONFLICT_CHECK`

  By default, if neither /BRANCH nor /FORCE_TIP is specified, UPLOAD assumes that /CONFLICT_CHECK was specified an searches for unresolved merge conflicts that correspond to each item revision to be updated. If any unresolved merge conflicts are found, Dimensions issues a warning and does not update the corresponding item revision. This gives you a chance to resolve conflicts before the update.

  You can turn off the check for unresolved conflicts by specifying /NOCONFLICT_CHECK when you invoke the UPLOAD command.

- `/FILTER=<filter-name>`

  Specifies that UPLOAD will create or update only files that satisfy the criteria specified in the <filter-name> area filter.

  An area filter is a regular expression following the same syntax as that used by the Dimensions GREP command.

- `<file-encoding>`

  Specifies the content encoding for new item revisions to be created. Supported encodings are the Microsoft codepages, the ISO-8859 variants (1–10), UTF-8, UTF-16, UTF-16BE, UTF-16LE, UTF32, UTF32BE, and UTF32LE.

Description

Allows the upload of a specified file or directory. If the specified files have been locally modified (by the use of optimistic locking) or are checked out by the user who is invoking the command, the files are checked in.

If the original file was fetched with item header substitution turned on (and one or more substitutions performed), the UPLOAD command produces a warning message and fails.

UPNO – Update Part Numbers

```
<part-spec>
[/GENERIC_NO=<standard-no> [/NOCHECK]]
[/LOCAL_NO=<local-no>]
[/DESCRIPTION=<description>]
```

Example

```
UPNO PROD:"RELEASE MANAGEMENT".AAAB -
/LOCAL="HP 44" /DESC="Release Support HP Version"
```

- `<part-spec>`

  Specifies the design part to be renumbered.

  It comprises: `<product-id>:<part-id>.<variant>;<pcs>`

  `<variant>` may be omitted if only one exists.

  `<pcs>` is ignored. A part number always applies to all PCSs.

- `/GENERIC_NO=<standard-no>`

  Specifies the modified standard part number to be allocated.

  It may be omitted, provided a `<local-no>` is specified.

- `/NOCHECK`

  Specifies the modified standard part number need not be in a range of numbers allocated to the product.

- `/LOCAL_NO=<local-no>`

  Specifies the modified local part-number to be allocated.

  It may be omitted, provided a `<standard-no>` is specified.

- `/DESCRIPTION=<description>`

  Specifies a new description to be given to the design part.

Constraints

Only users with the appropriate management privileges can run this command.

Each part category that is to use part numbers has to be enabled by the Administration Console.

UPROJ – Update a Dimensions Project

NOTE This command is no longer available. Please use UWA to update project attributes or Dimensions Build to manage build projects.

See the UWA command and the *Dimensions Build User's and Administrator's Guide*.

UREG – Register User

```
<user-id>
[/WORKSET=<project-spec>]
[/[NO]PASSWORD_SAVE]
[/ATTRIBUTES=(site=<site>,
    group_id=<group-id>,Dept=<dept>,
    full_name=<full-name>,phone=<phone>,
    <attribute-id>=<value>,email_addr=<email-addr>)]
```

Description

This command is the same as CUSR.

For details, see the *Serena Dimensions CM Administrator's Guide*.

URP – Unrelate Design Part

```
<part-spec>
/PARENT_PART=<parent-part-spec>
```

Example

```
URP PROD:"LIBRARY CONTROL".AAAA -
/PARENT=PROD:"RELEASE MANAGEMENT".AABB
```

- `/PARENT_PART=<parent-part-spec>`[1]

  (both for the child design part and its obsolete USAGE parent part) comprises:

  ```
  <prod-id>:<part-id>.<variant>;<pcs>
  ```

  `<variant>` may be omitted if only one variant of that design part exists.

  `<pcs>` is ignored; the current PCS is always used.

Constraints

Only users with the appropriate management privileges can run this command.

URSD – Update an Existing Resident Software Definition

NOTE This command is not available in the Issue Management–only licensed version of Dimensions.

```
/RSD_NAME=<name_RSD>
```

This command enables you to edit an existing installation Resident Software Definition (RSD). See the *Administrator's Guide* for details.

USUB – Unsubscribe from Notification Rule

```
<notification-id>
[/ROLES=(role1,role2,…)]
[/USER_LIST=(user1,user2,…)]
```

Example `SUB <rule-id> /USER_LIST=Smith`

- `<notification-id>`

  Name of the notification rule.

- `/ROLES`

  Specifies roles to unsubscribe from this notification rule.

- `/USER_LIST`

  Specifies users to unsubscribe from this notification rule.

Description

Unsubscribe users from a notification rule.

UUA – Update User Attributes

```
UUA
<user_name>
[/ATTRIBUTES=(<attribute-id>=<value>, <attribute-id>=<value>,…)]
[/[NO]PASSWORD_SAVE]
```

Example `UUA newuser /ATTRIBUTES=(site=Michigan, passport_id=1243)`

- `<user_name>`

  Specifies the identity of a user whose attribute(s) you want to update.

- `/ATTRIBUTES=<attribute-id>=<value>)`

  Specifies an attribute whose value is to be updated.

`<attribute -id>`	Specifies an attribute ID.
`<variant>`	is the attribute value.

- `[/[NO]PASSWORD_SAVE]`

  Certain Dimensions client tools have a "remote automatic login" facility (see, for example, the *User's Guide*). To be able to utilize this facility, the user's operating system login password on the remote Dimensions server must be saved locally on the client. When you update a Dimensions user's attributes:

  - This facility is permitted by specifying the `/PASSWORD_SAVE` qualifier. Subsequently, the first time a client tool is invoked the user will have the option in the Login dialog of storing their remote login password in an encrypted format on the client (under the existing connection specific key with an Access Control List (ACL) applied to it for added security). Further remote logins will than be automatic unless the CTRL or `Shift` key is kept depressed during client invocation.
  - This facility is deactivated by specifying the `/NOPASSWORD_SAVE` qualifier. If neither the `/PASSWORD_SAVE` nor the `/NOPASSWORD_SAVE` is specified, then the latter is the assumed default.

  If the user subsequently changes their remote login password:

  - The next time they try an automatic remote login from one of the specified client tools, they will receive a message informing them that their stored remote login password is incorrect.
  - The Login dialog will be displayed prompting them for their (new) remote login password.
  - Once connection has been established, the new remote login password will get stored as described above.
  - From then on, subsequent remote logins will again be automatic.

  The length of each item of information about the new user is limited, and any excess will be truncated. In particular, the database names, passwords and login-id are limited to 25 characters, and the department field is limited to 10 characters.

  An advisory mail message is sent to the user giving details on changes to the appropriate environment for running Dimensions. The Dimensions Batch Log is mailed to the DBA.

Description

This command enables you to update user attributes.

Constraints

Only users with the appropriate management privileges can run this command.

UWA – Update Project Attributes

NOTE This command is not available in the Issue Management–only licensed version of Dimensions.

```
<project-spec>
[/BRANCH or /TRUNK]
[/[NO]AUTO_REV]
[/[NO]PARALLEL_EXTRACT]
[/DESCRIPTION=<description>]
[/VALID_BRANCHES=(<branch-id1>,<branch-id2>,…)]
[/DEFAULT_BRANCH=<branch-id>]
[/ATTRIBUTES=(<attribute-value-list>)]
[/CHILD_ORDER=(<collection-list>)]
[/DEPLOYMENT_MODEL=MANUAL or /DEPLOYMENT_MODEL=AUTOMATIC]
[/COPY_ON_DEPLOY]
```

Example `UWA PROD:"WS MAINT DVL" /VALID_BRANCHES=(maint,upgrade)`

Parameters

- `<project-spec>`

  comprises `<product-id>:<project-id>`.

- `/BRANCH`

  Optional qualifier to adopt "branching" for the item revision scheme. This means that if an item revision is at revision `maint#5`, and the users decide to stay on this `maint` branch, subsequent revisions will be `maint#5.1`, `maint#5.2`, `maint#5.3`, and so forth.

- `/TRUNK`

  Optional qualifier to adopt "trunking" for the item revision scheme. This means that if an item revision is at revision `maint#5`, and the users decide to stay on this `maint` branch, subsequent revisions will be `maint#6`, `maint#7`, `maint#8`, and so forth.

- `/AUTO_REV or /NOAUTO_REV`

  Optional qualifier to tell Dimensions whether to automatically generate a new revision each time an item spec is edited/updated. If not, Dimensions requests the user to supply a revision.

- `/PARALLEL_EXTRACT or /NOPARALLEL_EXTRACT`

  Determines whether you can check out (extract) an item if a revision of that item is already checked out. This behaves in the same manner as "Allow Parallel Checkout" for item type options, but with respect to all item types on a per project basis. See "About Item Type Options" in the "Object Type Definitions" chapter of the *Process Modeling User's Guide* for a discussion of parallel check out, and other places in that guide for a discussion of parallel development in general.

- `<description>`

  An optional new description to be attached to the project definition, thus replacing the one which was assigned when the project was created (with the DWS command).

- /VALID_BRANCHES=(<branch-id1>,<branch-id2>,…)

  Identifies one or more branches—each previously defined in a Define (Item) Version Branches (DVB) command—that are to be valid for new item revisions created in this existing project. The list of valid branch ids replaces the list (if any) specified previously for this project using DWS or UWA.

  This list specifies the branches on which newly created item revisions can be placed.

  If the project attributes are set to have one or more valid branches, every new item revision in the project must use one of these branch ids.

  If the project attributes are set to have no valid branches, new revisions with no branch ids in them can continue to be used.

- /DEFAULT_BRANCH=<branch-id>

  Selects, from the valid list of branch ids, the branch id to specify the default branch for the whole project. If a default branch id is not defined, the first branch id in the valid list of branch ids is used as the default.

- /ATTRIBUTES=(<attribute-value-list>)

  Standard Dimensions user-defined attributes qualifier.

- /CHILD_ORDER=(<collection-list>)

  This qualifier is used to establish a search order among child collections. This order is used to resolve which item should be fetched when the paths in the child collections specify conflicting physical locations. The first collection in the list where a match occurs takes precedence.

- /DEPLOYMENT_MODEL=MANUAL or /DEPLOYMENT_MODEL=AUTOMATIC

  Specifies whether the project uses a manual or automatic deployment model.

- /COPY_ON_DEPLOY

  Optional qualifier that specifies whether files that are deployed from an earlier stage to a newer stage will be copied between areas rather than moved (they are moved by default).

Description

This command replaces SWS. It includes support for the /CHILD_ORDER qualifier.

Constraints

Only users with the appropriate management privileges can run this command.

WRC – Withdraw a Release from a Customer

NOTE This command is not available in the Issue Management–only licensed version of Dimensions.

```
<release-spec>
/CUSTOMER=<name>
/LOCATION=<location>
/PROJECT=<project-spec>
```

Example

```
WRC PROD:"R M 2.0 FOR HP" /CUSTOMER="Brown Finances -
/LOCATION="Bristol" /PROJECT="PAYROLL"
```

- `<release-spec>`

  Specifies the releases-spec, which comprises:
  `<product-id.:<release-id>`

- `/CUSTOMER=<name>`

  Specifies the customer's name.

- `/LOCATION=<location>`

  Specifies the customer's physical location.

- `/PROJECT=<project-spec>`

  Specifies the project name.

Description

The Dimensions product allows you to maintain a list of customers and a record of which Dimensions releases have been sent to each customer.

The WRC command enables you to remove the record of the fact that a release has been supplied to a specific customer.

Constraints

The combination of customer name, location, and project-spec must be unique in the Dimensions database.

You cannot forward the same release to a customer twice.

XAWS – Unrelate Area from Project

NOTE This command is not available in the Issue Management–only licensed version of Dimensions.

```
<area-name>
/WORKSET=<project-spec>
[/[NO]KEEP]
```

Example

- `XAWS DM10-WIN32 /WORKSET="PVCS:DM10"`

  Unrelates the `DM10-WIN32` area from the `PVCS:DM10` project. Does not delete any files.

Parameters

- `<area-name>`

  The name of the area that you want to unrelate from the specified project.

- `<project-spec>`

  The project from which you want to unrelate the specified area.

- `/KEEP or /NOKEEP`

  Specifies whether to keep files corresponding to the previously deployed item revisions in the area or delete them. By default, existing area files will not be deleted (/KEEP is default). If /NOKEEP is specified, files corresponding to item revisions from the specified project (including item revisions in child collections) previously transferred into this area will be deleted.

Description

This command breaks the relationship between an area and a project.

Constraints

Only users with the "Assign Deployment Areas to Project" privilege can run this command.

XBBL – Unrelate Baseline from Baseline

NOTE This command is not available in the Issue Management–only licensed version of Dimensions.

```
<child-baseline-spec>
/BASELINE=<parent-baseline-spec>
```

Example `XBBL <child-baseline-spec> /BASELINE=<parent-baseline-spec>`

Parameters

- `<child-baseline-spec>`

  Specifies the child baseline in the parent-child relationship.

- `<parent-baseline-spec>`

  Specifies the parent baseline in the parent-child relationship.

Description

This command breaks the relationship between a baseline and a parent baseline. The parent baseline must be at the initial lifecycle state.

Constraints

Only users with the appropriate management privileges can run this command.

XBCD – Unrelate Baselines from Requests

NOTE This command is not available in the Issue Management–only licensed version of Dimensions.

```
<baseline-spec>
/CHANGE_DOC_ID=(<request1>,<request2>,…)
[/AFFECTED or /IN_RESPONSE_TO or /INFO  ]
```

Example

```
XBCD "PAYROLL:ACME_2.1" -
/CHANGE_DOC=("PAYROLL_TDR_1","PAYROLL_TDR_2") /INFO
```

- `<baseline-spec>` comprises:

  `<product-id>:<baseline-id>`

- `/CHANGE_DOC_IDS=(<request1>,<request2>,…)`

  `<requestN>` identifies a request from which the specified baseline is to be unrelated.

- `/AFFECTED or /IN_RESPONSE_TO or /INFO`

  Specifies the type of relation to be deleted between the given baseline and the associated requests. The qualifiers are mutually exclusive.

  The default is `/AFFECTED`.

Constraints

XBCD is restricted to merge, release, and revised baselines. If it is run with respect to an archive or design baseline, then an appropriate error will be returned.

XBCD will only work successfully if you have both the baseline and requests in your pending list. If you specify an `/INFO` relationship, however, then this pending list restriction is relaxed.

There is no support for phase rules or change management rule enhancements within the context of baseline to request relationships.

Only the three relationship types – Info, Affected and in-Response-to – are supported. There is no support for user-defined relationship types.

XBWS – Unrelate Baseline from Project

NOTE This command is not available in the Issue Management–only licensed version of Dimensions.

```
<baseline-spec>
/WORKSET=<project-spec>
```

Example `XBWS <child-baseline-spec> /WORKSET=<parent-project-spec>`

Parameters

- `<baseline-spec>`

  Specifies the child baseline in the parent-child relationship.

- `<project-spec>`

  Specifies the parent project in the parent-child relationship.

Description

This command breaks the relationship between a baseline and a project.

Constraints

Only users with the appropriate management privileges can run this command.

XCCD – Unrelate Requests from Request

```
<request-id>
/CHANGE_DOC_IDS=(<request1>,<request2>,…)
[/DEPENDENT or /INFO ]
```

Example `XCCD PROD_CN_4 /CHANGE=PROD_DR_25`

- `<request-id>`

  This is the identity of the request which is the parent in the relationship to be removed.

- `/CHANGE_DOC_IDS=(<request1>,<request2>,…)`

  `<requestN>` is the identity of a request which is a child in the relationship to be removed.

- `/DEPENDENT or /INFO`

  Specifies the type of relation to be removed from between the given requests. The two qualifiers are mutually exclusive.

  The default is `/DEPENDENT`.

Constraints

This command can be run by users appropriate management privileges on the product or products owning the specific request concerned. Such users must have the parent request in their Pending List.

However, a user with the appropriate management privileges can set up the Process Model to specify that no request relationships can be modified for certain request types.

XICD – Unrelate Item from Requests

NOTE This command is not available in the Issue Management–only licensed version of Dimensions.

```
<item-spec>
[/FILENAME=<file-name>]
/CHANGE_DOC_IDS=(<request1>,<request2>,…)
[/AFFECTED or /IN_RESPONSE_TO or /INFO]
[/WORKSET=<project-spec>]
```

Example

```
XICD PROD:"QUERY RELEASE".AAAA-SRC;1 -
/CHANGE_DOC=(PROD_DR_25, PROD_DC_16)
```

- `<item-spec>` comprises:

  `<product-id>:<item-id>.<variant>–<item-type>;<revision>`

  `<item-id>` may be omitted if `<file-name>` is specified.

  `<variant>` may be omitted if only one exists.

  `<revision>` defaults to the latest revision (see Introduction on page 16).

- `/FILENAME=<file-name>`

  Specifies the name of the project file name.

  The project file name identifies the relative path (directory plus file name) from the working location to the item to be used from the current project. The project file name for the same item may *differ* between projects; for example, `src/hello.c`, `hello.c`, or `src/build/hello.c`.

  It may be omitted if `<item-id>` is specified.

- `/CHANGE_DOC_IDS=(<request1>,<request2>,…)`

  `<requestN>` identifies a request from which the specified item is to be unrelated.

- `/AFFECTED or /IN_RESPONSE_TO or /INFO`

  Specifies the type of relation to be deleted between the given item and the requests. The qualifiers are mutually exclusive.

  The default is `/AFFECTED`.

- `/WORKSET=<project-spec>` comprises:

  `<product-id>:<project-id>`

  This optionally specifies the project to be used for this command: failing this, the user's current project will be taken.

  Item revisions to be affected by the command may be specified explicitly, or they will be selected from the project.

Constraints

This command can be run only by users who have the request in their pending list or by a user with the appropriate management privileges.

XII – Unrelate Item from Item

NOTE This command is not available in the Issue Management–only licensed version of Dimensions.

```
<src-item-spec>
<dst-item-spec>
/RELATIONSHIP=<relationship-id>
[/FILENAME=<src-filename>]
[/FILENAME=<dst-filename>]
[/WORKSET=<project-spec>]
```

Example

```
XII "PROD_X:INTERFACE_C.AAAA-C;1" -
"PROD_X:INTERFACE_H.AAAA-H;1" -
/RELATIONSHIP="INCLUDES"
```

- `<src-item-spec>`

  Specifies the item from which the relationship instance will be removed.

- `<dst-item-spec>`

  Specifies the item at the other end of the relationship to be removed.

- `/RELATIONSHIP=<relationship-id>`

  Specifies the relationship type as defined by the DIR command

- `/FILENAME=<src-filename>`
  `/FILENAME=<dst-filename>`

  Optional qualifiers that further specify the source item and the destination item by giving their file names.

- `/WORKSET=<project-spec>` comprises:

  `<product-id>:<project-id>`

  This optionally specifies the project to be used for this command: failing this, the user's current project will be taken.

  Item revisions to be affected by the command may be specified explicitly, or they will be selected from the project.

Description

The XII command is used to remove instances of relationships created by the RII command (on page 293). The source and destination item types must be consistent with the relationship definition.

XIP – Unrelate Item from Part

NOTE This command is not available in the Issue Management–only licensed version of Dimensions.

```
<item-spec>
[/FILENAME=<file-name>]
/PART=<part-spec>
[/WORKSET=<project-spec>]
```

Example

```
XIP PROD:"QUERY RELEASE".AAAA-SRC -
/PART=PROD:"RELEASE MANAGEMENT".IBM
```

- `<item-spec>` comprises:

  `<product-id>:<item-id>.<variant>-<item-type>;<revision>`

  `<item-id>` may be omitted if `<file-name>` is specified.

  `<variant>` may be omitted if only one exists.

  `<revision>` is ignored; all revisions are unrelated from the specified design part.

- `/FILENAME=<file-name>`

  Specifies the name of the project file name.

  The project file name identifies the relative path (directory plus file name) from the working location to the item to be used from the current project. The project file name for the same item may *differ* between projects; for example, `src/hello.c`, `hello.c`, or `src/build/hello.c`.

  It may be omitted if `<item-id>` is specified.

- `/PART=<part-spec>` comprises:

  `<product-id>:<part-id>.<variant>;<pcs>`

  `<variant>` may be omitted if only one exists.

  `<pcs>` is ignored; the current PCS is always used.

- `/WORKSET=<project-spec>` comprises:

  `<product-id>:<project-id>`

  this optionally specifies the project to be used for this command: failing this, the user's current project will be taken.

  All Item revisions, regardless of whether they are in the project, will be affected.

Constraints

This command can be run only by users who have the item revision in their pending list or have the appropriate management privileges.

XPCD – Unrelate Part from Requests

```
<part-spec>
/CHANGE_DOC_IDS=(<request1>,<request2>,…)
```

Example

```
XPCD PROD:"RELEASE MANAGEMENT".AAAA -
/CHANGE_DOC=PROD_DR_26
```

- `<part-spec>` comprises:

  `<product-id>:<part-id>.<variant>;<pcs>`

  `<variant>` may be omitted if only one exists.

  `<pcs>` is ignored; the current PCS is always used.

- `/CHANGE_DOC_IDS=(<request1>,<request2>,…`

  `<requestN>` is the identity of a request from which the specified design part is to be unrelated.

Constraints

- This command can be run only by the user who has the request in their Pending list or by a user with the appropriate management privileges.
- This command cannot be used with the secondary catalog list.
- This command cannot be run if the request is at its end (closed) lifecycle state – even by a change-manager; however, a change-manager can action it back to an earlier lifecycle state perform the unrelate operation, and then action the request back to its closed state.

XRCD – Unrelate Requirement from Request

NOTE This command is not available in the Issue Management–only licensed version of Dimensions.

```
XRCD /CHANGE_DOC_ID=<request_id> /REQUIREMENT_ID=<requirement>
```

Description

This command breaks the link between a requirement and a request.

`<request-id>` is the request to be related to the requirement.

`<requirement>` is the requirement to be related to the request.

Constraints

Only users with the appropriate management privileges can run this command.

XREG – Unregister User

```
<user-id>
[\[NO]KEEP]
```

Description

This command is the same as DUSR.

For details, see the *Serena Dimensions CM Administrator's Guide*.

XWCD – Unrelate Project from Request

NOTE This command is not available in the Issue Management–only licensed version of Dimensions.

```
<project-spec>
/CHANGE_DOC_IDS=(<request1,<request2>,request3>,…)
```

Example `RWCD PAYROLL:PRJ_INITIAL /CHANGE_DOC_IDS=(PAYROLL_CR_21)`

Description

The example above unrelates the PAYROLL_CR_21 request from the PAYROLL:PRJ_INITIAL project.

Constraints

Only users with the appropriate management privileges can run this command.

XWWS – Unrelate Project from Project

NOTE This command is not available in the Issue Management–only licensed version of Dimensions.

```
<child-project-spec>
/WORKSET=<parent-project-spec>
```

Example `XWWS <child-project-spec> /WORKSET=<parent-project-spec>`

Parameters

- `<child-project-spec>`

  Specifies the child project in the parent-child relationship.

- `<parent-project-spec>`

  Specifies the parent project in the parent-child relationship.

Description

This command breaks the relationship between two projects

Constraints

Only users with the appropriate management privileges can run this command.

Chapter 3

Standalone Dimensions Utilities

Introduction

This chapter identifies various miscellaneous Serena Dimensions standalone utilities, some of which are described here and some of which are described in referenced related documents. Certain of these utilities are intended for use by users with the role of CHANGE-MANAGER, PRODUCT-MANAGER or PART-MANAGER only – these will be identified as such when discussed and for ease of reference will be referred to as change-managers, product-managers and part-managers respectively. Also, certain of the utilities are only available for specific operating systems – these too will be identified where appropriate.

The utilities discussed in this chapter are:

- **dm_auto_action** This utility is used to action requests once a date (attribute value) has passed.
- **dm_full_mail** and **dm_incremental_mail** These utilities are used to send users periodic reminders of requests in their pending lists.
- **dmpasswd** This utility ensures that Dimensions base database names, connection strings, and passwords are encrypted before being written to disk (they are stored in the Dimensions file `registery.dat` located in the `dfs` sub-directory).
- **prcs** [UNIX only] This utility provides a RCS-like front end to the version control commands of Dimensions.
- **psccs** [UNIX only] This utility provides an SCCS-like front end to the version control commands of Dimensions.

The auto-action and the two mail utilities have been designed to be executable as time-triggered automatic jobs, and some details on the use of UNIX's `crontab` are included.

The following standalone utilities are discussed in the indicated related documents.

- **download** This utility performs a download of a list of specified files under Dimensions into a target directory. See the *Administrator's Guide*.
- **dmdba** This is the Dimensions interactive DBA Tool. This tool is used to create and administer base databases and Dimensions published views. See the related document *Administrator's Guide*.
- **dm_mtu** This utility is used by Dimensions ART to read and write Archive and Transfer Baseline Out volumes. See the *Administrator's Guide*.
- **pdiff** (only for use by change managers and product-managers). This utility is used to import/export data via the Dimensions Data Interchange File Format into/from a specified Dimensions product. See the *Administrator's Guide*.
- **pm_label_branch** (only for use by tool-managers for the base databases concerned). This Dimensions Replicator utility enables tool managers to move items that are currently on a nameless branch to be placed on a named branch. See the *Administrator's Guide*.
- **replicator** (only for use by tool managers for the base databases concerned). This utility enables the tool manager to initiate the Dimensions replication process at the command prompt. It can be executed manually or automatically via a scheduler such as UNIX `cron`. See the *Administrator's Guide*.
- **upload** This utility performs an upload of a list of specified files in a user directory into Dimensions. See the *Administrator's Guide*.

General Information

All these utilities are run as independent programs from the operating system prompt (or from a command script file). They require the user to have performed a standard Dimensions user's login, which sets up the environment required for all Dimensions processing. The utilities all reside in the directory specified by the environment variable DM_PROG – which the standard login will include in the directory search path.

The syntax of each utility is explained under separate headings below, but the following general points are best explained in detail now:

Case Translation

Lower-case letters can be included in the values of parameters, and will automatically be interpreted as the equivalent in upper-case. This applies to all parameters, except those for which it is specifically stated that they are case-sensitive.

Wild Card Characters

In several parameters a range of possible values can be indicated by including a % (percent) character, which is interpreted as matching any zero or more characters. *This applies only to parameters for which it is stated that wild card % may be used*.

In other parameters a *null string* can be used to imply all possible values; a null string is specified as "" (two consecutive double-quote characters). This applies only to parameters for which it is stated that a null string may be used.

Execution Authority: Change-Manager or Tool-Manager

If the parameters are specified so that a utility is required to process the requests of several different products in the same execution, then the user must either have the CHANGE-MANAGER role for every product concerned, or else have the TOOL_MANAGER role for the database.

Actioning Requests by Date or Attribute Value

IMPORTANT! If the Dimensions CM "electronic signatures" facility is enabled (authentication required for sensitive changes to a request's or an item's lifecycle state and attributes), attempting automatic actioning to sensitive lifecycle states will always fail.

The dm_auto_action utility is available to all users and supports two different syntaxes:

Syntax (1st form)

```
dm_auto_action <product-id> <request-type> <holding-state>
   <preferred-state> <fallback-state> <date-attribute-name>
```

Example (1st form)

```
dm_auto_action PCMS CR HELD "CCB PENDING" \
   OUTSTANDING HOLD_UNTIL
```

This form of the utility actions each request in <product-id> of type <request-type>, whose current status is <holding-state>, to either <preferred-state> or <fallback-state>, provided that the value of <date-attribute-name> is less than the current system date (i.e. provided the value lies in the past).

<date-attribute-name> is the Variable Name parameter of a user-defined request attribute whose Data-Type is Date for date format (for details, refer to the Administration Console description (Configuration Object Management | Object Type Definitions | Requests | <request-type> | Attributes) in the *Process Modeling User's Guide* or the associated online help).

The action will be to <holding-state> provided that this will allow the request to be placed in at least one user's pending list. This means that: <holding-state> must not be a final state, and of the role(s) to handle the lifecycle transition(s) onwards from <holding-state>, at least one must be currently assigned to at least one user.

If this criterion cannot be met, the action will be to <pending-state> (regardless of what the pending-list position may be at this state: no requests that meet the specified criteria are left at <holding-state>).

Syntax (2nd form)

```
dm_auto_action <product-id> <request-type> <request-status>
   <preferred-state> <fallback-state> <attribute-name>=<value>
```

Example (2nd form)

```
dm_auto_action PCMS PR IN_PROGRESS CLOSED \
   COMPLETED WORK_STATUS=COMPLETED
```

In this form of the utility, if any requests of type <change-type> with status <request-status> have the value <value> in the attribute <attribute-name>, then an action check is performed for the preferred state <preferred-state>. If the action-check succeeds, the requests are actioned to the preferred state <preferred-state>, otherwise they are actioned to the fallback state <fallback-state>.

Sending Reminders of Pending Lists

Two utilities, available to all users, are provided to mail users with details on their pending requests:

- **dm_full mail**, which lists all relevant requests pending for each user; and
- **dm_incremental mail**, which lists only relevant requests last actioned (or, optionally, last modified) within the preceding few days.

Syntax

```
dm_full_mail [-v] <product-id> <request-type>
dm_incremental_mail [-u] [-v] <product-id> <request-type> <days>
```

`-v` specifies a verbose format for each request in the mail message (see below for details). The default is a brief format.

`-u` is valid only for `dm_incremental_mail`, and it specifies that the time of last modification is to be used rather than the time of last action to determine whether a request falls within the designated period.

<product-id> is the identity of the product whose requests are to be reported. A null string may be used to specify all products in the database.

<request-type> is the request type for the requests to be reported. A null string may be used to specify all request types.

<days> is the number of days to be covered by the messages generated by dm_incremental_mail. This period ends at midnight immediately before dm_incremental_mail is started.

For example, if three days are specified here, and dm_incremental_mail is started at 2 a.m. on Monday, requests last actioned at 1 a.m. the previous Friday or at 11 p.m. on Sunday will be reported, while requests last actioned at 11 p.m. on Thursday or at 1 a.m. today (Monday) will not be reported.

Separate mail messages are sent to each user listing all requests that meet the given selection criteria and that are pending for the user in a Leader, Primary, or Secondary role. The following are given for each request in both brief and verbose formats:

Request Identity

Current Status

Title (i.e. Attr-no 1): the first 70 characters

The following are given for each request in verbose format only:

The relevant user roles

Related Design Parts

Related Requests and their Current Status

Originator

These utilities are intended to provide regular reports to users on requests needing to be dealt with. They will normally be run as automatic jobs, as described below.

Automatic Job Triggering: Using crontab

If the following entries are placed in the crontab of the change-manager for product PRODX:

```
0 2 * * 2-5 /usr/local/bin/do_incmail PRODX PR 1
0 2 * * 1   /usr/local/bin/do_incmail PRODX PR 3
0 3 1 * *   /usr/local/bin/do_fullmail PRODX PR
```

then dm_full_mail will run at 3 a.m. on the first day of every month, while dm_incremental_mail will run at 2 a.m. on Mondays to Fridays (to cover just the preceding day, except on Monday when 3 days are included to cover the weekend). All reports are specified to cover requests of type PR in the PRODX product.

The entries refer to simple scripts that must be set up to invoke the utilities. For example, a C-shell script for do_incmail could be:

```
#!/bin/csh
source .login
dm_incremental_mail $1 $2 $3
exit
```

Similar crontab entries can be made to run dm_auto_action.

Encryption

In a Dimensions installation the following data is encrypted before being written to disk:

- User names
- Base database names
- Connection strings

  The above are collectively referred to as *User Ids* (<UserId>).

- Passwords

The encrypted information is stored in the Dimensions file registry.dat located in the dfs sub-directory of the installation home directory. The header in registry.dat contains the encryption version in use. For additional security, Serena recommends that only the Dimensions System Administrator user account (by default, dmsys) can read, and write to, registry.dat.

NOTE The User Ids are mapped to upper case before being encrypted. The passwords remain case sensitive.

The dmpasswd program enables <UserId> entries to be added and deleted, and for passwords to be changed for existing entries.

Constraints

Only the Dimensions System Administrator user account and the installation owner have permissions to run dmpasswd and access the registry.dat file. No Dimensions roles are required.

This feature is supported on UNIX and Windows. It is not supported on z/OS, since the feature is not required on that platform.

Syntax

```
dmpasswd <UserId> -add [ -pwd <Password> ] | -mod | -del | -help
```

where:

- <UserId>

  Specifies the user id entry for a particular user. This argument is mandatory and is always the first argument specified. Examples of <UserId> are:

  - Oracle: <base_db_name>@<connect_string>
  - IBM DB2: <connect_string>
  - Dimensions installation owner: dmsys

- `-add`

  Interactively adds a new `<UserID>` entry for a particular user. The program prompts for a password followed by confirmation of that password. The `-pwd <password>` argument can be used to avoid `dmpasswd` prompting for the password.

  Appropriate message texts are logged, for example, `user added`, `user already exists`, and `passwords don't match`. The `-add`, `-mod`, and `-del` arguments are mutually exclusive.

- `-pwd <Password>`

  Optional argument (can only be used with the `-add` option) that specifies the case sensitive password to be used.

- `-mod`

  Interactively modifies the password of an existing `<UserId>`. The program prompts for the old password followed by the new password, and then prompts for confirmation of the new password.

  Appropriate message texts are logged, for example, `password changed`, `user does not exist`, `invalid password`, and `new passwords don't match`. The `-add`, `-mod`, and `-del` arguments are mutually exclusive.

- `-del`

  Deletes a `<UserId>`. The program prompts for the password, and then prompts for confirmation of the deletion.

  Appropriate message texts are logged, for example, `user deleted`, `user does not exist`, and `invalid password`. The `-add`, `-mod`, and `-del` arguments are mutually exclusive.

- `-help`

  Displays program usage.

Examples

Command	Description
`dmpasswd intermediate@dim9 -add`	Registers the `intermediate` base database for an Oracle instance `dim9`.
`dmpasswd dmsys -add -pwd Not-Telling`	Registers the password of the pool owner user account on Windows platforms.
`dmpasswd pcms_sys -mod`	Modifies the password associated with the `pcms_sys` user-id.
`dmpasswd intermediate@dim9 -del`	Deletes the `intermediate@dim9` user-id.
`dmpasswd -help`	Lists a usage summary.
`dmpasswd dim9 -add -pwd dmsys/foobar`	Registers the user-name/password credential required to access a remote IBM UDB database pointed to by the ODBC alias `dim9`.

Connecting to DB2 on a Remote Host

To connect a Dimensions server to a DB2 database running on a remote host, you must execute the following command as the Dimensions System Administrator:

```
dmpasswd <DSN> -add -pwd <USER-NAME>/<PASSWORD>
```

where:

- <DSN>

  Specifies the data source name that is defined on the Dimensions CM server host, which points at the remote DB2 database to be used.

- <USER-NAME>

  Specifies the OS user name of the Dimensions database owner on the remote host running DB2.

- <PASSWORD>

  Specifies the OS password of the Dimensions database owner on the remote host running DB2.

NOTE On Windows hosts check that `registry.dat` contains the credentials of the pool owner user. For example, `dmpasswd <pool_owner> -add -pwd <password`

PRCS–RCS-like Front End to Dimensions

```
prcs subcommand [ option…] [ filename…]
   [ Dimensions option…]
```

NOTE prcs cannot be run from Windows or Dimensions for z/OS.

Examples

1 To create a Dimensions item for file `program.c`, ensure that the desired project is set and that you are in a sub-directory within the scope of the working location, then use *'prcs ci':*

```
example% prcs ci program.c +P fs:fs.aaaa +T src
PCMS/program.c <-- program.c
enter description, terminated with single '.' or end of file:
NOTE: This is NOT the log message!
>> cryptic description
>> .
initial revision: 1.1
done
```

2 To check out a copy of `program.c` for editing. Edit it, and then check it back in:

```
example% prcs co -l program.c
PCMS/program.c --> program.c
revision 1.1 (locked)
```

```
done
example% vi program.c
your editing session
example% prcs ci program.c
PCMS/program.c <-- program.c
new revision: 1.2; previous revision: 1.1
enter log message, terminated with single '.' or end of file:
>> clarified cryptic diagnostic
>> .
done
```

3 To check out with lock all files under Dimensions in the current directory:

```
example% prcs co -l PCMS
```

4 To check in all files currently checked-out to you:

```
example% prcs ci 'prcs rlog -R -L PCMS'
```

5 To search for `printf` in all files in the current project directory held in Dimensions.

```
example% prcs grep printf PCMS
```

6 To run `wc` on all files in the current project directory held in Dimensions.

```
example% prcs eval wc -- PCMS
```

(See page 425 for **Description**.)

Subcommands

The following `prcs` subcommands invoke programs that provide functionality with similar semantics to the standard RCS commands. See Constraints for non-supported RCS options on page 427.

- `co [-l[rev] -u[rev] -p[rev] -q[rev] -r[rev] -sstate -w[login] -k -ddate-time] filename….`

  Gets a revision from Dimensions. By default, this is a read-only working copy of the most recent revision on the trunk. The specified revision can be locked so that other users of `prcs` cannot deposit new revisions with this as a direct predecessor. Dimensions Item header substitutions are performed if enabled unless the revision is being locked or the `-k` option is used. Refer to `co(1)`.

`-r<rev>` Retrieves the latest revision whose number is less than or equal to `rev`. If `rev` indicates a branch rather than a revision, the latest revision on that branch is retrieved. If `rev` is omitted, the latest revision on the trunk will be retrieved.

`-l<rev>` Retrieves a revision for editing. The checked out revision will be writable and `prcs` will not allow other users to lock the specified revision. A repeated lock on the same revision will be allowed. The `-l` option overrides the `-u` option.

`-p<rev>` Prints the retrieved revision on the standard output.

`-q<rev>` Quiet mode: diagnostics are not printed. Check out will be aborted if there is a writable copy of the file in the working directory.

-u<rev> Same as -r except it unlocks the retrieved revision if it was locked by the caller. If rev is omitted, -u retrieves the revision locked by the caller if there was one, indicates errors if more than one is locked or else it retrieves the tip trunk revision.

-f<rev> Forces overwriting of the working file. This is useful with -q.

-ddate Retrieves the latest revision on the selected branch whose check in time is less than or equal to present date/time. Date may take the form: *mm/dd/yyyy [hhmm]* or dd-mon-yyyy [hh:mm]. Omitted units that default to zero values e.g. 08/23/1996 are equivalent to **any** setting of the locale that affects *date/time* formats, will be honored.

-k Does not perform Dimensions item header substitution. This is the same as the /NOEXPAND option to the Dimensions FI command. This is provided for compatibility with the psccs command.

-sstate Retrieves the latest revision on the selected branch whose state is set to state.

-w<login> Retrieves the latest revision on the selected branch which was checked in by the user with login name login. If the user name is omitted, the caller's login is assumed.

- ci [-l[rev] -u[rev] -q[rev] -r[rev] -sstate -i -j -mmsg -tdesc] filename…

  Checks in new revisions. The effective user ID must be the same as the ID of the person who has the predecessor revision locked or must have a role to update the item. Refer to **ci(1).**

  Unless the -f option is given, ci will check if the working file differs from the preceding one. If there are no differences, ci will not perform the check in. Check-summing must be turned on for the item type if this mode of operation is to be enabled. If check-summing is off or the files differ, a new revision will be created. For each revision deposited a log message is requested. If multiple files are to be operated on, ci will ask if the log message is to be reused. If a log message is given with -m, this is used for all check ins.

  If there is no equivalent Dimensions item, a new item is created with default revision 1.1. The +P, +T and optionally +F, together with any desired +C or +A options must be provided. The following steps will be performed:

  - Creates a Dimensions item with the following parameters:

    Item-id derived from the file name by replacing all "**_ .$ # ~ ; : -**" characters with a space character and truncating to 25 characters. If this is not unique, the user must use Dimensions to create the item.

    Item type given by the +T Dimensions Option.

    Revision defaults to 1.1 unless a revision is given using the -r option. If just the release component is given, the initial revision will then be <release>.1.

    Owned by the design part given by the +P Dimensions specific option.

    The project file name will consist of a project directory that matches the difference between the current working directory and working location and the file name specified to the create command.

    The +F Dimensions specific option must be used to specify the format if the file name has no extension.

- Performs a `prcs co` or `prcs co -l` on the item to retrieve a read-only or editable copy of the initial revision if the `-u` or `-l` options were given.

The revision given to any new Dimensions revision will be determined in the following way.

- If `rev` is a full revision number, it must be higher than the latest one on the branch to which the revision is being appended.
- If `rev` is a branch number, the revision is appended to an existing branch by incrementing the sequence component.
- If the branch given is non-existent and the branch point exists, then a new branch is created with the initial revision being `Rev.1`.
- If `rev` is omitted, the new revision will be derived from existing locks on the file.
- If a tip revision was locked, then a new revision will be appended to the relevant trunk or branch. The new revision number will be given by incrementing the branch or sequence component.
- If a non-tip revision was locked, then a new distinct branch will be started at that point by incrementing the branch component and starting with a sequence component of 1. Only one revision must be locked by the caller if the new revision is to be omitted from the command.

`-r<rev>` Checks in revision `rev`. The working file is deleted.

`-r` With no revision specified to the `-r` option removes the working file and unlocks the predecessor revision. This overrides any `-l` or `-u` options given to the command**.**

`-l<rev>` Works like `-r` except a `co -l` is performed. Any requests or attributes required for the check out must be specified.

`-u<rev>` Works like `-l` except a `co` is performed.

`-f<rev>` Forces deposition of the new revision even if it does not differ from the predecessor.

`-q<rev>` Diagnostics are not printed. Revisions identical to the predecessor are not deposited unless `-f` is specified or item check-summing is not enabled.

`-mmsg` Uses the string `msg` for all deposited revisions.

`-I` Initializes: error indicated if file already exists in Dimensions.

`-j` Just checks in: error indicated if file does not exist in Dimensions.

`-sstate` Actions the deposited revision to state. If state is omitted, the next normal lifecycle state is assumed.

`-tstring` Uses string as description when creating a new item.

`-t-string` Uses string as description when creating a new item. The '`-`' character is stripped from the string.

- `rcs [-l<rev>] [-u<rev>] [-o<rev>] [-i] [-sstate[:rev] filename….`

The `rcs` command performs administrative functions.

`-l<rev>` Locks the specified revision. A check out is performed to a temporary file that is then discarded.

`-u<rev>` Unlocks the specified revision. Only the locker of a revision can unlock it.

`-o<rev>` Deletes the specified revision.

`-I` Creates a new empty item in Dimensions. The revision given is `1.1`. The +P, +T, +F and any **+C** or +A Dimensions specific options must be supplied. This differs from RCS usage which does not create an initial revision for the **`rcs -i`** command until a `co` is performed. See `ci` (on page 420) for a description of the operations performed on initialization.

`-s<state><:rev>` Action the revision specified by `rev` to state `state`. If state is omitted, assume the next normal lifecycle state.

- `rlog [-L] [-R] [-h] [-t] [-N] [-b] [-ddates] [-l<lockers>] [-r<revisions>] [-sstates] [-w<logins>] filename…`

  Prints rcs-style information about Dimensions item. Refer to `rlog(1)`. The intersection of the revisions selected with `-d -l -s` and `-w` intersected with the union of revisions selected by `-r` and `-b` are printed.

  `-L` Ignores files that have no locks set.

  `-R` Prints the name of the file prefixed by `Dimensions/`.

  `-h` Prints **only** the file name, head revision, locks and symbolic names.

  `-t` As `-h` plus description.

  `-N` Does not print symbolic names.

  `-b` Only prints information for the highest branch on the trunk.

  `-ddates` Prints information about revisions with a check in within the specified range of date/times. The *date/time* values are as described for `co` (on page 419). The '**<**', '**>**' and '**=**' characters in conjunction with one or two dates specify the range.

  - d1<d2 or d2>d1 between d1 and d2 (exclusive)
  - <d or d> revisions earlier than d
  - d< or >d revisions later than d
  - If '=' follows the '<' or '>' characters the comparisons are inclusive.
  - d single revision dated d or earlier

  `-l<lockers>` Prints information about locked revisions only. `Lockers` is a comma-separated list of user names that restrict the selection to only those items locked by the specified users.

  `-r<revisions>` This specifies a range of revisions to report on.

  - `:rev` specifies revisions from the beginning of the branch to `rev`.
  - `rev:` specifies revisions from `rev` to the end of the specified branch.

  A branch name means all revisions on that branch. A range of branches means all revisions on the specified branches and a bare `-r` means the tip of the trunk. Only one range can be specified. A separator of **' - '** is also supported.

  `-sstates` Only print information for revisions that have a status matching any one of the comma-separated list states.

-wlogins Only report on revision checked in by the comma-separated list of users logins.

- `rcsdiff [diffoptions] [-k] [-q] [-rrev1] [-rrev2] filename…`

  Compares two revisions corresponding to the specified revisions using `diff`. Refer to `rcsdiff(1)`. All options except `-k` are passed to `diff`, see `diff(1)`. The `-k` option turns off item header substitution. If `-q` is specified, diagnostics are not printed.

  - If both `rev1` and `rev2` are omitted, `rcsdiff` compares the latest revision on the trunk with the contents of the corresponding working file. That is, the file with the same leaf file name in the current working directory.
  - If `rev1` is given but `rev2` is omitted, `rcsdiff` compares revision `rev1` with the contents of the corresponding working file.
  - If both `rev1` and `rev2` are given, `rcsdiff` compares revisions `rev1` and `rev2` of the corresponding Dimensions item.
  - `rev1` can be given numerically or symbolically, `rev2` must be given explicitly.

- `help`

  Displays a list of the supported commands.

- `ls [-l] [-R]`

  Displays the list of files that are in the matching directory in the current project. The `-l` and `-R` options cannot be combined.

`l` Appends latest revision to file name.

`R` Performs List recursively.

- `grep [-r<rev>] [grep_options] [grep_string] filename…`

  Performs the `grep` command on the tip trunk revision of the specified files in the current project. The `-r` option can be used to specify a particular revision or named branch. The `grep` options are as documented in `grep (1)`. The `grep` string should be enclosed in single quotation characters (') if the search string contains spaces or shell meta-characters.

- `eval [-r<rev>] [command] [command_options] -- filename…`

  Performs the specified command on the tip trunk revision of the specified files in the current project. The `-r` option can be used specify a revision or named branch. The options are as documented for the specified **Any** parameters that contain spaces or shell meta-characters which should be enclosed in single-quotation characters ('). The -- separator is required after the last command parameter.

Dimensions Options

The **prcs** command takes a number of subcommand specific options and, additionally, Dimensions specific options may be specified. Dimensions-specific options for `prcs` must appear after the subcommand and optional fIlename arguments. Options for a given subcommand must appear after the subcommand argument. These options are specific to each subcommand, and are described with the subcommands themselves (see Subcommands on page 419).

- +V

  Displays the Dimensions commands and messages generated by the `prcs` operation. The project selected by `prcs` will also be displayed.

- +W <project>

  Defines the Dimensions project to be used for the prcs command. This option is required if prcs is unable to determine the project from the current directory. The working location is set to the current working directory unless a +D option is also specified.

- +D <working location>

  Defines the working location to be used for the project given in the +W Dimensions Option. This cannot be specified without the +W option.

- +C <Request List>

  This option allows a list of requests to be supplied to the Dimensions command invoked by prcs where this is appropriate. This applies to checking-out with lock creation (rcs -i, ci) and check in commands that force an update. The format of the option is:

  ```
  +C 'CHANGE_DOC_ID_1[, CHANGE_DOC_ID_2,…]'
  ```

- +A <Attribute List>

  This option allows a list of Dimensions item attribute values to be supplied to the Dimensions command invoked by prcs where this is appropriate. This applies to checking-out with lock creation (rcs -i, ci) and check in commands that force an update. The format of the option is:

  ```
  +A 'ATTRIBUTE_VARIABLE1=value[, ATTRIBUTE_VARIABLE2=value,…]@
  ```

- +N <New Revision>

  This option allows a specific new revision to be specified instead of that determined by prcs.

- +P <Design Part>

  This option allows a Dimensions design part to be specified for the prcs commands that create new Dimensions items. The format of the option is:

  ```
  +P 'PRODUCT_ID:PART_ID.VARIANT[;PCS]'
  ```

- +T <Item Type>

  This option allows a Dimensions item type to be specified for the prcs commands that create new Dimensions items.

- +F <Item Format>

  This option allows a Dimensions item format to be specified for the prcs create and prcs enter commands. This is required only if there is no extension on the file being entered into Dimensions. Otherwise the format is determined from the extension.

  All values to be passed to Dimensions incorporating embedded spaces must be protected by double-quotation characters (""). The whole parameter should be enclosed in single- quotation characters (') to protect it from interpretation by the shell.

Description

The prcs command is a RCS-like front end to the version control commands of Dimensions.

prcs applies the indicated *subcommand* to the Dimensions 'item' associated with each of the specified files.

The mapping between the file name(s) specified the prcs command and the associated Dimensions 'item' is derived from the file name by searching for a matching project file name in the current project.

The prcs command expects these 'items' to reside in a project directory that when appended to the current project root equates to the current working directory. Thus, when you invoke prcs from directory "/usr/work" which is marked as being the working location for project WS1 with a file name argument, the subcommand will be applied to a Dimensions item in the root directory of project WS1 with project file name equivalent to that specified to the command. If the command is issued in a subdirectory of "/usr/work", then the file name is appended to the offset from the working location to the current directory and this name is used to find the 'item'. If the file name is given as PCMS or PCMS/<wildcard>, then the command is applied to all Dimensions items which are in the same relative sub-directory of the project and which match the wildcard. The wildcard is in standard Dimensions format. Thus executing the prcs command in the working location prcs co program.c would apply the co (check out) subcommand to an 'item' with project file name program.c. However, executing the same prcs command in the project directory "src" would apply the co subcommand to an item with project file name src/program.c, while the command prcs co PCMS executed in a directory "/usr/work/src" in project WS1 with root directory "/usr/work/" would apply the co subcommand to every item in Dimensions with the project directory "src".

prcs uses the same mechanism as Dimensions Make to determine the correct project for operation. If it is unable to determine a project, then a diagnostic message is issued and the +W and +D Dimensions-specific options must be used (see Dimensions Options on page 423).

Revisions

prcs accepts revision specifications using the following syntax:

```
-option<revision_specification>
```

where option depending on the prcs subcommand is one of:

```
l, p, q, r, f, u
```

There should be no space between the -option and the revision specification. The revision specification can contain Dimensions named branches, for example:

```
-rmaint#1
```

prcs will also accept a symbolic name for a revision. prcs will use the revision of the relevant Dimensions item contained in the Dimensions baseline identified by the symbolic name. The Dimensions product-id can be omitted, in which case, the product-id is taken from the project that prcs is operating in.

```
prcs co -rfs:source_1.0 foo.c
```

Will check out the revision of foo.c contained in baseline fs:source_1.0.

prcs by default uses the RCS style of revisioning. This is composed of the following components.

```
Release.Level[.Branch.Sequence.[ Branch.Sequence…]]
```

By convention RCS revisions start at 1.1 and the main trunk consists of revisions with just Release and Level components. The default action of prcs is to operate on the tip trunk revision. This is the revision with the highest Release and Level numbers. The default next revision for the tip is to increment the level component. Branches are identified by adding branch and sequence components to the trunk revision specifier. A branch is initiated by checking in a new revision following a check out with lock on a non-tip trunk revision or by explicit use of a branch revision specifier. The first branch from Revision 1.2 would be 1.2.1.1. Subsequent branches from Revision 1.2 would have the Branch component incremented. A branch from a branch is identified by appending 1.1 to the to the current branch revision.

Dimensions revisions do not follow the same convention**. prcs treats Dimensions revisions with a single numerical field as belonging to Release '0'**. A branch revision e.g. 1.1 will be interpreted by prcs as a trunk revision in Release 1. Dimensions style revisions can be accessed by prcs commands by using default action or explicit specification. If default operation is used, the latest revision with a style revision or a RCS style trunk revision would be used. Specifying a release of '0' to a prcs command restricts the search to Dimensions style single numeric field revisions. Dimensions style revisions can be accessed explicitly by prefixing their revision with the non-existent Release 0. A branch created from a style revision will have the new revision 0.revision.1.1 so that it will appear as a branch in the RCS style of revisioning.

prcs prints the revisions that will be operated on by default. These values should be checked to ensure that the desired revision is being accessed.

Dimensions Named Branches

prcs supports Dimensions named branches in the following way. A named branch is specified by using the branch name in the prcs revision specification. Specifying a branch name without the branch revision will be interpreted as the highest revision on the branch. Specifying a branch name and a revision will access that revision. Editing a non-tip named branch revision causes the new revision to be an unnamed branch from the named branch, for example:

editing foo.c;fix23#1.1 gives new revision: *foo.c;fix23#1.1.1.1*

prcs will use the default project branch if the branch name is omitted e.g. prcs co -r# will checkout the tip trunk revision on the default branch, and prcs co -r#2.3.1.4 will check out the specified revision on the default branch.

Environment

All environment variables required for correct Dimensions functioning must be present.

prcs Dimensions Files

$DM_PROG/prcs	prcs program

Constraints

This command is available to all users who have the roles to perform the associated Dimensions operations, but not from Windows or z/OS systems.

Currently only emulations of the co, ci, rcs, rlog, and rcsdiff RCS commands are supported. The additional non-rcs commands ls, grep and eval are supported to provide additional functionality.

The following RCS commands are not supported.

- rcsmerge
- ident

The following command options for prcs subcommands are not supported:

```
co
        -I -M -j -V -x –z
ci
        -k -d -M -n/N -I -T -w -V -x -z

rcs
        -a/A -e -b -c -k -L -U -I -m -M -n/N -T -V

rlog
        -V -x -z
```

Refer to the UNIX manual pages for the equivalent RCS commands for further information.

NOTE

There is a conflict between the Forced Revision Generation flag on Dimensions projects and the new revisions generated by prcs. Do not set this flag if the project is going to be used for prcs operations.

The use of automatic Item-Id generation is detected and a null Item-Id will be used for the creation of items using the ci and rcs -i subcommands. The implicit get or check out when the -u or -l option is supplied to ci will not be performed. An explicit co or co -l should be performed.

The calling syntax for prcs is different from the common usage of RCS in which the individual subcommands are called directly from the shell prompt. Users of shells with an alias capability can alias the prcs subcommands to their common form. For example, the following syntax can be used in the csh:

```
% alias co prcs co.
```

PSCCS–SCCS-like Front End to Dimensions

```
psccs subcommand [ option…] [ filename…]
   [ Dimensions option…]
```

NOTE psccs cannot be run from Windows or Dimensions for z/OS.

Examples

1 To create a Dimensions item for file `program.c`, ensure that the desired project is set and that you are in a subdirectory within the scope of the working location. Then use `psccs create`:

```
example% psccs create program.c +P PAYROLL:PAYROLL.A +T src
program.c:
1.1
```

2 To check out a copy of `program.c` for editing, edit it, and then check it back in:

```
example% psccs edit program.c
1.1
new delta 1.2
25 lines
example% vi program.c
your editing session
example% psccs delget program.c
comments? clarified cryptic diagnostic 1.2
1.2
```

3 To check out all files under Dimensions in the current directory:

```
example% psccs edit PCMS
```

4 To check in all files currently checked out to you:

```
example% psccs delta 'psccs tell -u'
```

5 To search for `printf` in all files in the current project directory held in Dimensions.

```
example% psccs grep printf PCMS
```

6 To run `wc` on all files in the current project directory held in Dimensions.

```
example% psccs eval wc -- PCMS
```

(See page 433 for **Description**.)

Subcommands

The following `psccs` subcommands invoke programs that provide functionality with similar semantics to the standard UNIX SCCS commands. Many of these subcommands accept additional arguments that are documented in the reference page for the equivalent SCCS utility. (See **Constraints** on page 434 for non-supported SCCS options.)

- `check [-b] [-u[<user-name>]]`

  Checks for files currently being edited. Like `info` and `tell`, but returns an exit code, in addition to producing a listing of files. `check` returns a non-zero exit status if anything is being edited.

  `-b` Ignores branches.

  `-u[<user-name>]` Only checks files being edited by you. When `<user-name>` is specified, only check files being edited by that user.

- `create [-r<release>] filename…`

  Creates (initializes) Dimensions item for specified file(s). `create` performs the following steps:

  - Create a Dimensions item with the following parameters.
    - Item-Id derived from the file name by replacing all "_ . $ # ~" characters with a space character and truncating to 25 characters.
    - Item type given by the +T Dimensions specific option.
    - Revision defaults to `1.1` unless a release is given using the `-r` option. The initial revision will be `<release>.1`.
    - Owned by the design part given by the +P Dimensions specific option.
    - This project file name consists of a project directory that matches the difference between the current working directory and working location and the file name specified to the create command.
    - The +F Dimensions specific option must be used to specify the format if the file name has no extension.
  - Performs a `psccs get` on the specified file to retrieve a read-only copy of the initial revision.

- `deledit [-s] [-y[`*comment*`]]filename…`

  Equivalent to a `psccs delta` and then a `psccs edit`. `deledit` checks in a new revision, and checks the file back out again.

  `-s` Silent. Does not report revision numbers or statistics.

  `-y[comment]` Supplies a comment for the revision commentary. If `-y` is omitted, the command will prompt for a comment. A NULL *comment* results in an empty comment for the checked in revision.

- `delget [-s] [-y[`*comment*`]]filename…`

  Performs a `psccs delta` and then a `psccs get` to check in the new revision and retrieve a read-only copy of the resulting new revision. See the `deleted` subcommand for a description of `-s` and `-y`. `psccs` performs a `delta` on all the files specified in the argument list, and then a `get` on all the files. If an error occurs during the `delta`, the `get` is not performed.

- `delta [-s] [-n] [-y[`*comment*`]]filename…`

  Checks-in pending changes. The effective user ID must be the same as the ID of the person who has the file checked out. Refer to `sccs-delta(1)`. See the `deledit` subcommand for a description of `-s` and `-y`. The `-n` option keeps the working file.

- `diffs [-C] [-c`*date-time*`] [-r`*revision*`] filename…`

Compares (in diff(1) format) the working copy of a file that is checked out for editing, with a revision from the Dimensions history. Use the most recent checked-in revision by default. The diffs subcommand does not accept any diff, options with the exception of the -c option to diff which must be specified as -C.

-C Passes the -c option to diff.

-cdate-time Uses the most recent version checked in before the indicated date and time for comparison. date-time takes the form: mm/dd/yyyy [hh:mm] dd-mon-yyyy [hh:mm] Omitted units default to zero values; that is -c08/23/1996 is equivalent to -c08/23/1996 00:00.

-r<*revision*> Uses the revision specified for comparison.

- edit

  Retrieves a revision of the file for editing. psccs edit checks out a version of the file that is writable by the user and marks the new revision as *checked out* in Dimensions, so that no one else can check that revision in or out. Item header substitutions are not performed. edit accepts the same options as get, below.

- enter

  Similar to create, but omits the final psccs get. This may be used if an psccs edit is to be performed immediately after creation.

- get [-e-k-p-s] [-c*date-time*] [-r*revision*] filename…

  Gets a revision from Dimensions. By default, this is a read-only working copy of the most recent revision on the trunk; Dimensions item header substitutions are performed if enabled. Refer to sccs-get(1).

  -e Retrieves a revision for editing. Same as psccs edit.

  -k Does not perform Dimensions **item header substitution**. This is the same as the /NOEXPAND option to the FI command.

  -p Produces the retrieved revision on the standard output. Reports that would normally go to the standard output (revisions, statistics) are directed to the standard error.

  -s Silent. Does not report revision numbers or statistics.

  -c*date-time* Use the most recent version checked in before the indicated date and time for comparison. date-time takes the form: mm/dd/yyyy [hh:mm] dd-mon-yyyy [hh:mm] Omitted units default to zero values; that is -c08/23/1996 is equivalent to -c08/23/1996 00:00. Any setting of the locale that affects date/time formats will be honored.

  -r<revision> Retrieves the specified revision.

- help

  Displays a list of the supported commands.

- info [-b] [-u[<user-name>]]

  Displays a list of files being edited, including the revision checked out, the revision to be checked in, the name of the user who holds the lock, and the date and time the file was checked out.

  -b Ignores branches.

`-u[<user-name>]` Only lists files checked out by you. When `<user-name>` is specified, only list files checked out by that user.

- `print`

  Prints the entire history of each named file. Equivalent to an `psccs prs -e`

- `prs [-el] [-c`*`date-time`*`] [-r`*`revision`*`] filename…`

  Peruses (displays) the history of a Dimensions item. Refer to `sccs-prs(1)`.

  `-e` Displays history information for all revisions earlier than the one specified with `-r` (or all revisions if none is specified).

  `-l` Displays information for all revisions later than, and including, that specified by `-c` or `-r`.

  `-cd`*`ate-time`* Specifies the latest revision checked in before the indicated date and time. The *date-time* argument takes the form:
  `mm/dd/yyyy [hh:mm] dd-mon-yyyy [hh:mm]`

  `-rr`*`evision`* Specifies a given revision

- `sccsdiff -r`*`old-revision`* `-r`*`new-revision`* `-C` *`filename`*`…`

  Compares two revisions corresponding to those specified using `diff`. Refer to `sccs-sccsdiff(1)`. Only the `-c` option to `diff` is supported which must be specified as `-C`.

- `tell [-b] [-u[<user-name>]]`

  Displays the list of files that are currently checked out, one file name per line.

  `-b` Ignores branches.

`-u[<user-name>]` Only lists files checked-out to you. When `<user-name>` is specified, only list files check out to that user.

- `unedit filename…`

  "Undoes" all pending `edits` or `get -e`, and checks in the working copy to its previous condition. `unedit` backs out all pending changes made since the file was checked out.

- `unget [-n]`

  Same as `unedit`. Refer to `sccs-unget(1)`. The `-n` option keeps the working file.

- `ls [-l] [-R]`

  Displays the list of files that are in the matching directory in the current project. The `-l` and `-R` options cannot be combined.

  `-l` Appends revision to file name.

  `-R` Performs List recursively.

- `grep [grep_options] [grep_string] filename…`

  Performs the `grep` command on the tip trunk revision of the specified files in the current project. The grep options are as documented in `grep (1)`. The grep string should be enclosed in single quotation characters (') if the search string contains spaces or shell meta-characters.

- `eval [command] [command_options] -- filename…`

  Performs the specified command on the tip trunk revision of the specified files in the current project. The options are as documented for the specified command. Any parameters that contain spaces or shell meta-characters should be enclosed in single quotation characters ('). The -- separator is required after the last command parameter.

Dimensions Options

As well as the subcommand specific options, the psccs command can take a number of Dimensions specific options. Such-specific options for psccs must appear after the subcommand and optional file name arguments.

(Options for a given subcommand must appear after the subcommand argument. These options are specific to each subcommand, and are described along with the subcommands themselves, see Subcommands on page 428.)

- +V

  Displays the Dimensions commands and messages generated by the psccs operation. The project selected by psccs will also be displayed.

- +W <project>

  Defines the Dimensions project to be used for the psccs command. This option is required if psccs is unable to determine the project from the current directory.

- +D <working location>

  Defines the working location to be used for the project given in the +W Dimensions Option.

- +C <Request List>

  This option allows a list of requests to be supplied to the Dimensions command invoked by psccs where this is appropriate. This applies to edit and creation commands only. The format of the option is:
  `+C ''CHANGE_DOC_ID_1[, CHANGE_DOC_ID_2,…]'`

- +A <Attribute List>

  This option allows a list of Dimensions item attribute values to be supplied to the Dimensions command invoked by psccs where this is appropriate. This applies to edit and creation commands only. The format of the option is:
  ```
  +A 'ATTRIBUTE_VARIABLE1=value
        [,ATTRIBUTE_VARIABLE2=value,…]'
  ```

- +N <New Revision>

  This option allows a specific new revision to be specified instead of that determined by psccs. This applies to the edit and get -e commands only.

- +P <Design Part>

  This option allows a Dimensions design part to be specified for the psccs create and psccs enter commands. The format of the option is:
  `+P 'PRODUCT_ID:PART_ID.VARIANT[;PCS]'`

- +T <Item Type>

This option allows a Dimensions item type to be specified for the psccs create and psccs enter commands.

- +F <Item Format>

  This option allows a Dimensions item format to be specified for the psccs create and psccs enter commands. This is only required if there is no extension on the file being entered into Dimensions. Otherwise the format is determined from the extension.

  All values to be passed to Dimensions incorporating embedded spaces must be protected by double-quotation characters ("). The whole parameter should be enclosed in single-quotation characters (') to protect it from interpretation by the shell.

Description

The psccs command is a SCCS-like front end to the version control commands of Dimensions.

psccs applies the indicated subcommand to the Dimensions item associated with each of the specified files.

The mapping between the file name(s) specified to the psccs command and the associated Dimensions item is derived from the file name by searching for a matching project file name in the current project.

The psccs command expects these 'items' to reside in a project directory that when appended to the current project root equates to the current working directory. Thus, when you invoke psccs from directory "/usr/work" which is marked as being the working location for project WS1 with a file name argument, the subcommand will be applied to an item in the root directory of project WS1 with project file name equivalent to that specified to the command. If the command is issued in a sub-directory of "/usr/work", then the file name is appended to the offset from the working location to the current directory and this name is used to find the Dimensions item.

If the file name is given as PCMS or PCMS/<wildcard>, then the command is applied to all Dimensions items which are in the same relative sub-directory of the project and which match the wildcard. The wildcard is in standard Dimensions format. Thus, executing the psccs command in the working location psccs get program.c would apply the get subcommand to an item with project file name program.c

However, executing the same psccs command in the project directory "src" would apply the get subcommand to an 'item' with project file name src/program.c while the command psccs get PCMS executed in a directory "/usr/work/src" in project WS1 with root directory "/usr/work/" would apply the get command to every item in Dimensions with the project directory "src".

psccs uses the same mechanism as Dimensions Make to determine the correct project for operation. If it is unable to determine a project, then a diagnostic message is issued and the +W and +D Dimensions specific options (see Dimensions Options on page 432) must be used.

Revisions

psccs uses the SCCS style of revisioning. This is composed of the following components.

```
Release.Level.Branch.Sequence
```

By convention SCCS revisions start at 1.1 and the main trunk consists of revisions with just Release and Level components. The default action of psccs is to use the tip trunk revision. This is the revision with the highest Release and Level numbers. A branch is initiated by editing a non-tip trunk revision. The first branch from revision 1.2 would be 1.2.1.1. Subsequent branches from revision 1.2 would have the branch component incremented. A branch from a branch is created by incrementing the branch component and setting the sequence component to 1. If this branch was already used the branch component is incremented until a non-used branch number is found.

Dimensions revisions do not follow the same convention. **psccs treats Dimensions revisions with a single numerical field as belonging to Release '0'**. A branch revision e.g. 1.1 will be interpreted by psccs as a trunk revision in Release '1'. Dimensions style revisions can be accessed by psccs commands by using default action or explicit specification. If default operation is used then the latest revision with a style revision or a SCCS style trunk revision would be used. Specifying a release of 0 to a psccs command restricts the search to Dimensions style single numeric field revisions. Dimensions style revisions can be accessed explicitly by prefixing their revision with the non-existent Release '0'. A branch created from a style revision will have the new revision 0.revision.1.1 so that it will appear as a branch in the SCCS style of revisioning.

psccs prints the revisions that will be operated on by default. These values should be checked to ensure that the desired revision is being accessed.

Dimensions Named Branches

psccs supports Dimensions named branches in the following way. A named branch is specified by using the branch name in the psccs -r option. Specifying a branch name without the branch revision will be interpreted as the highest revision on the branch. Specifying a branch name and a revision will access that revision. Editing a *non-tip* named branch revision causes the new revision to be an unnamed branch from the named branch e.g. editing foo.c;fix23#1.1 gives new revision: foo.c;fix23#1.1.1.1

psccs Dimensions Files

$DM_PROG/psccs	psccs program

Constraints

Currently only emulations of the check, create, deledit, delget, delta, diffs, edit, enter, get, help, info, prs, sccsdiff, tell, unedit and unget SCCS commands are supported. The additional non-sccs commands eval, grep, and ls are supported to provide additional functionality.

The following SCCS commands are not supported:

- admin
- cdc
- clean
- print
- rmdel
- val

- comb
- fix
- what

The following command options for psccs subcommands are not supported:

- admin
  -h -z -a -d -e -f -l -m
- get
  -m -n -l[p] -a -i -x
- delta
  -p -g -m

Refer to the UNIX manual pages for the equivalent SCCS commands for further information.

Appendix 1

New Commands in Serena Dimensions CM 10.1

This appendix contains a list of links to the new commands in Dimensions CM 10.1.

Index

A

B

C

D

E

F

G

H

I

L

M

N

O

P

Q

R

S

T

U

V

W

X

Z