opentext™

Dimensions CM

Administration Guide

Copyright © 2023 Open Text.

The only warranties for products and services of Open Text and its affiliates and licensors (“Open Text”) are as may be set forth in the express warranty statements accompanying such products and services. Nothing herein should be construed as constituting an additional warranty. Open Text shall not be liable for technical or editorial errors or omissions contained herein. The information contained herein is subject to change without notice.

Last updated: December 8, 2023

Table of Contents

Part 1 **General Administration 23**

Chapter 1 **Configuring Web Aspects of OpenText™ Dimensions CM Server 25**

Introduction 26
Overview 26
Web Client Components 26
Starting and Stopping Web Tools 27
Introduction 27
Starting Tomcat 27
Stopping Tomcat 29
Installing or Removing the Tomcat Windows Service 29
Running Web Tools with Alternative Web Servers 29
Configuring Microsoft IIS to Work with Tomcat 30
Installing Native Components on a Windows or UNIX Client Node 30
Starting the Web Client on a Client Node 30
Reassigning Default Web Client Connection Parameters 31
Introduction 31
Reassigning the Default listener Port Number 31
Changing or Adding Connection Strings or Parameters 33
Configuring Web Client Tools 34
Configuring Web Client Tools 34
Disabling Web Client Tools 35
Support for Your Own Web Application Container 35
Installing Web Tools into Tomcat 35
Problems with Unicode Characters in File Names 37

Chapter 2 **Setting Dimensions Server and Client UNIX Symbols 39**

Introduction 40
Dimensions CM Operating System Environment Variables 40
The Dimensions dm.cfg File 40
Additional Dimensions Configuration Symbols 41
Modifying or Adding dm.cfg Entries 41
Installation and File Location Symbols 42
DM_TMP 42
DM_REMOTE_FILE_AREA 42
Using a Proxy Server 42
DM_NET_PROXY_MODE 43
DM_NET_PROXY 44
DM_NET_NOPROXY 44
Sending Email to Dimensions Users 44
DM_MAIL 45
DM_MAIL_SCRIPT 46

DM_DMEMAIL 46
Disabling Email Notifications on UNIX Systems 46
Paging Data 46
Printing Data 47
DM_PRINT 47
DM_CHD_PRINT 47
Cleaning Up Temp Files Created when Running User Reports 48
Merging Item Revisions 48
Networking and Replication 49
Heterogeneous Networking 49
DM_SERVICE_PCMS_SDP_TCP 49
DM_SERVICES_FILE 49
Code Page Settings 49
DM_CODEPAGE 49
DM_EBCDIC_CODEPAGES 50
DM_DEFAULT_CODEPAGE 50
Oracle Collating Sequence for 'ORDER BY' Queries 50
DM_NLS_SORT_PARAM 50
Dimensions Web Tools URL Parameters 51
DM_WEB_URL 51
DM_INET_PORT 51
DM_INET_SERVER 51
Overriding Host Name Assigned to DMSERVER Certificates 51
Additional Dimensions Configuration Symbols 52
Adding Template Variables 52
Restricting Visibility of Objects in CM Clients 52
Running Multiple CM Instances on a UNIX Host Machine 53
Smart Attribute Validation 54
Using PAM on Dimensions Servers 54
Remote Execution on a Tertiary Node Running HP-UX 11.23 on Itanium 54
Specifying the Server Domain Name 54
File Compression Settings 55
Item Header Substitution Last-Record Behavior 55
Disabling Branch Check-Ins 55
Enabling Global Project Structure Change Control 56
Privilege Caching 57
Cleaning Up Temporary Build Files 57
Build Error Handling 58
DM_BLD_ERROR_INVALID_REVISIONS 59
DM_BUILD_DMORDERING 59
DM_BLD_GETSRC_FUNCTION_OR 59
Supporting Refactoring Changes When Creating Revised Baselines . . . 59
Ordering of Attributes 59
Visibility of Requests on Design Part Structure 60
Item Library Security 60
Disabling the Automatic Selection of Build Targets 61
Contents of Temporary Build Project 61
Pre-version 8 Item Header Substitution Functionality 61

Display Related Items for Child Requests in Request Browse Templates 62
Processing Items or Requests not Owned Locally 62
Checking Mandatory Multi-Valued Attributes in Block Attributes 63
Event Tracing 63
Disabling Editing of Block Attributes on Multiple Requests 63
Pre-10.1.3 Behavior For Deleting Non-Tip Item Revisions. 64
Pre-2009 R1 Item Library Access Mechanism 64
Visual Studio Clients: Prompting for Check Out 65
Controlling Deployment Regression 65
Automatically Spreading Item Library Files across Multiple Subfolders . 66
Retaining UNIX File Permissions 66
Checking a Client's Version. 66
Checking Baseline Consistency 66
Preserving REXEC Command Log Files 67
Setting the DBIO Library 68
Specifying the Database Vendor 68
Dimensions for z/OS Configuration Symbols 68
Defaults for Copying Build Information 69
SSO Configuration Symbols 70
CAC Configuration Symbols. 71

Chapter 3 **Setting Dimensions Server Windows Symbols 73**

Introduction 74
The Dimensions dm.cfg File 74
Additional Dimensions Configuration Symbols 75
Modifying or Adding dm.cfg Entries 75
Installation and File Location Symbols 76
Using a Proxy Server 76
DM_NET_PROXY_MODE 76
DM_NET_PROXY 77
DM_NET_NOPROXY 78
Sending Email to Dimensions Users 78
DM_MAILS 78
The Default Email Client. 79
Accessing the Email Client from Dimensions for z/OS 81
DM_DMEMAIL 82
Printing Data 82
DM_PRINT 82
Clearing Up Temporary Files Created by User Reports 82
DM_DELETE 82
Reporting and Optional Symbols 83
Windows Server Log On Domain 83
DM_LOGON_DOMAIN. 83
Networking and Replication 84
Code Page Settings 84
DM_CODEPAGE. 84
DM_EBCDIC_CODEPAGES 84
DM_DEFAULT_CODEPAGE 84

Oracle Collating Sequence for 'ORDER BY' Queries 84
DM_NLS_SORT_PARAM 84
Dimensions Web Tools URL Parameters 84
DM_WEB_URL. 84
DM_INET_PORT 84
DM_INET_SERVER. 84
Additional Dimensions Configuration Symbols 85
Adding Template Variables 85
Restricting Visibility of Objects in Web Tools 85
Using Network Attached Storage Devices for Item Libraries 85
Smart Attribute Validation 85
Using PAM on Dimensions Servers. 85
Unable to Log In from Dimensions Desktop Client 85
Using the Library Cache with the VS .NET SCC Integration 86
Integrating with Visual Basic Through SCC 86
Specifying Server Domain name 86
File Compression Settings 86
Item Header Substitution Last Record Behavior 86
Disabling Branch Check-ins 86
Enabling Global Project Structure Change Control 86
Privilege Caching. 86
Cleaning Up Temporary Build Files 87
Build Error Handling 87
DM_BLD_ERROR_INVALID_REVISIONS 87
DM_BUILD_DMORDERING 87
DM_BLD_GETSRC_FUNCTION_OR 87
Supporting Refactoring Changes When Creating Revised Baselines . . . 87
Ordering of Attributes 87
Visibility of Requests on Design Part Structure 87
Item Library Security. 87
Disabling the Automatic Selection of Build Targets 87
Contents of Temporary Build Project 87
Old Style Item Header Substitution Functionality. 88
Display Related Items for Child Requests in Request Browse Templates 88
Processing Items or Requests That You Do Not Locally Own 88
Checking of Mandatory Multi-Valued Attributes in a Block Attribute . . . 88
Event Tracing 88
Disabling Editing Block Attributes on Multiple Requests 88
Pre-10.1.3 Behavior For Deleting Non-Tip Item Revisions. 88
Pre-2009 R1 Item Library Access Mechanism 88
Visual Studio Clients: Prompting for Check Out 88
Controlling Deployment Regression 89
Checking the Client Version 89
Checking Baseline Consistency 89
Setting the DBIO Library 89
Specifying the Database Vendor 90
Overriding Host Name Assigned to DMSERVER Certificates 90
Dimensions for z/OS Configuration Symbols 90

Defaults for Copying Build Information 90
SSO Configuration Symbols 90
CAC Configuration Symbols 90

Chapter 4 **Setting Dimensions Windows Client Symbols 91**
All Windows Client Nodes 92
Modifying or Adding dm.cfg Entries 92
Installation and File Location Symbol 93
DM_TMP 93
Printing Data 93
DM_PRINT 93
Connecting from a Windows Client to a Non-Default Listener on UNIX 93
Turning Off SCC Server Confirmation Pop-Up Messages 94
Using Dimensions CM SCC with Visual Studio Web Projects 94
Visibility of Requests on Design Part Structure 94

Chapter 5 **Administering Licenses and the License Server 95**
Overview 96

Chapter 6 **Configuring and Using Auto Update 97**
Introduction 98
Changes to Installation Files 98
Silent Install Options 99
Configuring Auto Update on a CM Server 99
Example Schedules 102
Configuring Auto Update on Clients and Agents 102
Running Client Auto Update 102
Running Agent Auto Update 103
Listing Upgrade History 103
Getting Update Statistics 104
Scheduling Auto Update on a Client or Agent 104
Solutions to Common Issues 105

Chapter 7 **The Dimensions CM Server HTTP Connector 107**
Introduction 108
Configuring the Connector 108

Chapter 8 **Configuring the Default Merge Tools 111**
Introduction 112
Merge Configuration File 112

Part 2 Database and Tool Administration 117

Chapter 9 **Introducing Database and Tool Administration 119**
Overview of Dimensions Use of an RDBMS 120
RDBMS DBA 121
System Administrator and Their Responsibilities 121
Users with Administrative Privileges and Their Responsibilities 123

Types of Dimensions Schema 124
Base Database Schema 124
PCMS_SYS Schema 124
Types of Dimensions User 125
Registered Users 125
Proxy Users 125
Summary of Administrative Personas v Specific Roles or Privileges 126

Chapter 10 **General Operating Guide 131**
General Prerequisites 132
Oracle Privileges Required for DMDBA Commands 133
Running the Dimensions DBA Tools 133
Invoking the dm_control Tool 136
Invoking the dmdba Tool 136
Invoking Dimensions Server Pooling 138
Invoking dmcli 141
Invoking the Administration Console 142
Accessing the Serena-Supplied Runtime RDBMS DBA Utilities 142
Accessing Oracle Enterprise or SQL Server Enterprise DBA Utilities 143
Database Connectivity Mechanisms 143
Connecting to a Serena-Supplied Runtime RDBMS or Oracle Enterprise 143
Connecting to SQL Server Enterprise on a Remote Host 143
Database-Dependent Schema Limits 144

Chapter 11 **PostgreSQL Backup and Recovery 145**
Overview 146
SQL Dump 146
File System Level Backup 146
Continuous Archiving 147

Chapter 12 **System Administration 149**
Introduction 150
Starting, Stopping, and Restarting Dimensions 151
Starting Dimensions for Windows 151
Starting Dimensions for UNIX 152
Stopping Dimensions for Windows 153
Stopping Dimensions for UNIX 153
Restarting Dimensions for Windows 153
Restarting Dimensions for UNIX 154
Administering Dimensions Schemas 155
CRDB – Create Dimensions Base Database 155
DLDB – Delete Dimensions Base Database 157
LSDB – List Dimensions Base Databases 158
EXPM – Exporting a Process Model to a File 159
SSEQ – Resetting the Sequence Value for a Current Base Database . . 160
CPAS – Changing Passwords for Databases 161
SPAC– Reporting on Tablespace Usage 161
INCR – Increasing Database Space Available to a Tablespace 161

STATISTICS – Manage Statistics for the Database 162
INSV – Installing Report Views 162
DELV – Deleting Report Views 163
GRTV – Granting Report Views to a Dimensions User 163
REKV – Revoking Report Views from a Dimensions User 163
LSGV – Listing Dimensions Users Granted Report Views Access 164
Recreating and Resizing Indexes in a Specified Base Database 164
MIGRATE– Migrate an MBCS Encoded Database to UTF-8 165
ITEMLIBMODE - Item Library Access Policy 165
Configuring Dimensions Server Pooling 165
Starting and Configuring Dimensions Server Pooling 165
Displaying Connection Pooling Statistics 170
Connection Pooling Statistics Main Window 170
How to View Connection Pooling Statistics 171
Managing Remote Jobs 171
About Remote Jobs 172
Remote Jobs Main Window 172
Remote Jobs Menu Area 172
Remote Jobs Navigation Area 172
Remote Jobs Content Area 173
How to List and View Remote Jobs 173
How to Edit and Delete Remote Jobs 173
Administering Dimensions Users 174
UREG – Register Users 176
XREG – Unregister Users 178
UUA– Update User Attributes 179
RREG – Reassign User Registration 180
CCS – Create Credential Set 181
DCS – Delete Credential Set 183
UCS – Update Credential Set 183
LCS – List Credential Sets for Current User 183
Assigning General Administrative Privileges 184
Backing Up Item Libraries 184
Configuring the dmlcasrv Server 185
Encrypting Usernames and Passwords 185

Chapter 13 **Serena-Supplied Runtime RDBMS Database Administration . 187**

Chapter Overview 188
Disclaimer 188
Starting the Serena-Supplied Runtime RDBMS 188
Starting the Serena-Supplied Runtime RDBMS for Windows 188
Starting the Serena-Supplied Runtime RDBMS for UNIX 189
Stopping the Serena-Supplied Runtime RDBMS 189
Stopping the Serena-Supplied Runtime RDBMS for Windows 189
Stopping the Serena-Supplied Runtime RDBMS for UNIX 190
Restarting the Serena-Supplied Runtime RDBMS 190
Restarting the Serena-Supplied Runtime RDBMS for Windows 190
Restarting the Serena-Supplied Runtime RDBMS for UNIX 191

Backing Up Serena-Supplied Runtime RDBMS Dimensions Databases 191
Overview 191
The dm_backup Syntax 192
Restoring Serena-Supplied Runtime RDBMS Dimensions Databases 192
Overview 192
The dm_restore Syntax 193
Preparing the Restored Database Schemas for Use 194
Reducing Database Fragmentation 195
Additional Serena-Supplied Runtime RDBMS Control Files 195
Use of Serena-Supplied Runtime RDBMS Tablespaces 196
Recovery After Media Failure 197

Part 3 Data Migration 199

Chapter 14 **Introducing Data Migration 201**
Migrating Operating System or Existing Configuration Management Data . . 202
Web Client Data Migration Facility 202
Desktop Client Data Migration Facility 202
Exporting a User's Settings to Another Desktop Client 203
Dimensions dmcli Commands 203
Standalone Utilities 203
Data Interchange File Format (PDIFF) Import/Export Utility 204

Chapter 15 **Dimensions Migration Standalone Utilities 205**
Foreword 206
General Information 207
Case Translation 207
Wildcard Characters 207
Execution Authority 208
Dimensions Download 208
Dimensions Upload 210
RCS-Like Front End to Dimensions: prcs 212
SCCS-Like Front End to Dimensions: psccs 213

Chapter 16 **Data Interchange File Format (PDIFF) 215**
Foreword 216
PDIFF Specification 216
Change Document (Dimensions CM Request) POID 217
System Change Document (Request) Attributes 218
User Attribute – (Optional) 219
Action - (Optional) 219
Update – (Optional) 220
Relate – (Mandatory: at Least One Design Part Relate Required) 220
Delegate – (Optional) 221
Attribute History – (Optional) 221
Attribute Update History – (Optional) 221
Request Attachments – (Optional) 222
Design Part POID 222

System Design Part Attributes . . . 223
User Attribute – (Optional) . . . 223
Relate – (Optional) . . . 224
Relate History – (Optional) . . . 224
Control Plan (Process Model) POIDs . . . 224
Process Model Attribute Valid Sets . . . 225
Process Model Delegatees . . . 225
Process Model Change Request and Item Browse Templates . . . 226
Replicator Changes to POID Definitions . . . 226
Example PDIFF Files . . . 227
Change Document (Request) Example . . . 227
Design Part Example . . . 229
Process Model Example . . . 232

Chapter 17 **Using PDIFF to Import/Export Data . . . 235**
General Description . . . 236
Process Model Export . . . 236
Process Model Import . . . 237
Design Part Export . . . 238
Design Part Import . . . 238
Request Export . . . 239
Request Import . . . 241
Submitting PDIFF Commands from a Parameter File . . . 242
Limitations . . . 242

Chapter 18 **PDIFF Syntax . . . 243**

Part 4 **Network Administration . . . 249**

Chapter 19 **Introducing Network Administration . . . 251**
Overview . . . 252
Network Architecture . . . 253
Physical Node Types . . . 253
Client Node . . . 253
Library Server Node . . . 253
Application Server Node . . . 254
Logical Nodes . . . 254
Code Page . . . 256
What is a Code Page? . . . 256
Code Page Conversion Between Platforms . . . 256
Code Page Parameters . . . 257
Byte Order Mark . . . 258
What is a Byte Order Mark (BOM)? . . . 258
BOM Implementation in Dimensions . . . 258
Content Encoding . . . 259
General . . . 259
Mainframe Restrictions . . . 260
Valid Encodings . . . 260

File System Registration for Remote Nodes . . . 261
Item Operations in a Dimensions Network . . . 261
Privileged and User Command Modes . . . 262
Dimensions Network Limitations . . . 262
Using Network Attached Storage Devices for Item Libraries . . . 263
Setting Up Physical Nodes . . . 264
Dimensions Network Node Performance Issues . . . 264
Default Dimensions TCP/IP Port Allocations . . . 265
Dimensions Server or Agents . . . 265
Dimensions Web Applications . . . 265
Remote Serena-Supplied Runtime or Oracle Enterprise RDBMS . . . 267

Chapter 20 **Dimensions CM Network Guide . . . 269**
Overview . . . 270
Specifying Operating Systems . . . 270
Specifying Network Nodes . . . 270
Defining Logical Nodes . . . 271
Specifying Network Client and Server Connections . . . 271
Specifying Code Pages . . . 271
Specifying File Systems for Remote Nodes . . . 271
Registering Resident Software . . . 272
Dimensions CM Listener . . . 272
Listener Startup and Shutdown . . . 272
Listener Functions . . . 272

Chapter 21 **Network Applications . . . 275**
About Network Administration . . . 276
Network Administration Main Window . . . 276
Network Administration Menu Area . . . 276
Network Administration Navigation Area . . . 277
Network Administration Content Area . . . 278
Network Administration Status Area . . . 280
Managing Network Nodes . . . 280
Creating Nodes . . . 280
Managing Contacts . . . 281
Managing Network Objects . . . 282
Managing Base Databases . . . 282
Editing or Deleting Nodes . . . 283
Managing Client/Server Connections . . . 284
Creating Client/Server Connections . . . 284
Editing or Deleting Connections . . . 284
Viewing Objects . . . 285
Viewing Objects . . . 285

Chapter 22 **Network Administration Command Line Interface . . . 287**
Using the Dimensions CM Command Line Interface . . . 288
Using Key Qualifiers . . . 288
Listing Qualifiers . . . 288

Sample Network Administration Scenario 289
Base Database 291
Description 291
Commands 291
Qualifiers 291
Examples 292
Codeset 293
Description 293
Commands 293
Qualifiers 293
Examples 293
Contact. 293
Description 293
Commands 293
Qualifiers 294
Examples 294
Database Instance 295
Description 295
Commands 295
Qualifiers 295
Examples 296
File System 296
Description 296
Commands 296
Qualifiers 296
Examples 296
Network Node 297
Description 297
Commands 297
Qualifiers 297
Examples 297
Network Node Connections 298
Description 298
Commands 298
Qualifiers 298
Examples 299
Network Object 300
Description 300
Commands 300
Qualifiers 300
Examples 300
Node Objects. 301
Description 301
Commands 301
Qualifiers 301
Operating System 301
Description 301
Commands 301

Qualifiers . . . 302
Examples . . . 302
Protocols . . . 302
Description . . . 302
Command . . . 302
RSD . . . 303
Description . . . 303
Commands . . . 303
Qualifiers . . . 303

Part 5 Replication . . . 305

Chapter 23 **Multi-Site Development Using Replicator . . . 307**

Introduction . . . 308
Overview . . . 308
Item Replication . . . 308
Baseline Replication . . . 310
Request Replication . . . 311
Processing Items or Requests That You Do Not Locally Own . . . 312
Integrating Dimensions RM and CM in a Replicated Environment . . . 313
Getting Started . . . 313
Familiarization . . . 313
Planning your Replication Scenario . . . 314
Defining an Item Replication Configuration . . . 315
Defining a Baseline Replication Configuration . . . 316
Defining a Request Replication Configuration . . . 316
Ensuring the Minimum Level of Compliance between Process Models . . . 316
Defining a Temporary Work Area . . . 316
Online Replication Network Objects . . . 317
Running Replicator . . . 317
Replication Results . . . 317
Replicator Published Views . . . 317

Chapter 24 **Online Replication Concepts and Scenarios . . . 319**

Basic Concepts . . . 320
Related Dimensions Concepts . . . 320
Types of Synchronization and Replication . . . 322
Item Synchronization . . . 322
Local Item Replication . . . 323
Remote Item Replication . . . 323
Primary and Replica Base Database Sites . . . 324
Site Ownership of Named Branches . . . 324
Replication Configurations . . . 326
Invoking Remote Replication . . . 328
Determining which Item Revisions to Replicate . . . 328
Replication of Updates to Item Metadata . . . 330
Remote Item Replication Example Scenarios . . . 331
General Guidelines . . . 331

Dimensions-Specific Details . . . 333
Setting Up . . . 334
Baseline Replication . . . 334
Primary and Replica Base Databases . . . 335
Replication Configuration . . . 335
Initiating Baseline Replication . . . 335
Baseline Replication Item Transfer Failures . . . 335
Differences between Replicated and Release Baselines . . . 336
Determining which Item Revisions to Replicate . . . 336
Replication of Design Parts to Replica Sites . . . 336
Request Replication . . . 336
Request Replication Configuration . . . 337
Reference Sites . . . 337
Initiating Request Replication . . . 337
Request Replication Transfer Failures . . . 337
Request Replication Example Scenarios . . . 337
Overview of Replication Models . . . 337
Example Development Environment . . . 338
Implementing a Major New Feature . . . 338
Summary of Delegation Model . . . 339
Fixing Issues in a Distributed Development Environment . . . 340
Summary of Request Model . . . 340

Chapter 25 **Offline Replication Concepts and User Scenarios . . . 341**
Basic Concepts . . . 342
Air-Gap Replication . . . 343
Setting Up Air-Gap Replication . . . 343
Performing an Air-Gap Replication Run . . . 344
Dumping on the Primary Site . . . 344
Loading on the Replica Site . . . 345
Capturing the Replica Site Load Log Directory . . . 345
Loading the Replica Load Log Directory on the Primary Site . . . 346
What Gets Created on the Operating System Disk? . . . 346
Air-Gap Directory Structure Example . . . 346
The Replication Metadata File (RMF) . . . 347
Considerations . . . 347
Replication Logs . . . 347
Data Integrity . . . 347
Baselines . . . 347

Chapter 26 **Preparing for Replication . . . 349**
Defining and Controlling Remote Sites for Replication . . . 350
Overview . . . 350
Defining a Physical Node Name and Database . . . 351
Registering Valid Databases for Replication Against a Node . . . 351
Setting Up Replication Configurations . . . 352
Setting Up Replication Configurations Using the Administration Console 352
Setting Up Replication Configurations Using dmpmcli . . . 352

Constraints . . . 352
Setting Up Remote Users for Replication . . . 353
Remote Password Security . . . 353

Chapter 27 **Replication Administration . . . 355**
About Replication Administration . . . 356
What is a Replication Configuration? . . . 356
What is a Replication Log? . . . 356
Replication Administration Main Window . . . 357
Replication Administration Menu Area . . . 357
Replication Administration Navigation Area . . . 358
Replication Administration Content Area . . . 359
Replication Administration Status Area . . . 361
Managing Replication Configurations for Projects . . . 361
Defining Replication Configurations for Projects . . . 361
Defining Replica Sites for Project Replication . . . 362
Modifying Project Replication Configurations . . . 363
Managing Replication Configurations for Baselines . . . 364
Defining Replication Configurations for Baselines . . . 364
Defining Remote Replicas for Baselines . . . 365
Modifying Baseline Replication Configurations . . . 365
Managing Replication Configurations for Requests . . . 366
Defining Replication Configurations for Requests . . . 366
Defining Remote Replicas for Request Replications . . . 367
Modifying Request Replication Configurations . . . 368
Managing Replication Logs . . . 368
Listing and Viewing Replication Logs . . . 368

Chapter 28 **Replication Rules and Constraints . . . 371**
Introduction . . . 372
General Item-Specific Restrictions . . . 373
General Baseline-Specific Restrictions . . . 374
General Request-Specific Restrictions . . . 374
Minimum Process Model Compliance Requirements . . . 375
Metadata Replication Rules . . . 375
Integrity Constraints on Replicated and Transferred Version Branches . . . 377
Data Integrity and Extending Replication Configurations . . . 378

Chapter 29 **Running Replicator . . . 379**
Overview . . . 380
Launching Replicator from the Command Line . . . 380
Replicator Syntax . . . 381

Part 6 Archive, Retrieval, and Transfer Facilities . . . 389

Chapter 30 **Introduction to Archive, Retrieval, and Transfer Facilities . . . 391**
Dimensions ART Overview . . . 392
Access to Dimensions ART Facilities . . . 393

Baselines for Archives . . . 393
Creating an Archive . . . 394
Deleting an Archive . . . 394
Removing Online Archived Material . . . 394
Restrictions . . . 395
Transferring and Receiving Baselines . . . 395
Enhancements to Early Versions of TBO/TBI . . . 397
Cross-Platform Transfers . . . 397
Routing Errors Accepted by TBI . . . 398
Transfer of Requests . . . 398
Transfer of User-Defined Attributes . . . 399
Transfer Baseline Identity . . . 399
Tokens and Reports . . . 400
Change Management Rules . . . 401
Archive and Retrieval Across Dimensions Networks . . . 401
Magnetic Tape Utility . . . 402
Compatibility with Earlier Versions of Dimensions ART . . . 402
Incompatibility Between MBCS and UTF-8 Database Character Sets . . . 403

Chapter 31 **ART Command Mode Applications . . . 405**
Introduction . . . 406
CAR - Create Archive . . . 407
DAR - Delete Archive . . . 409
RAMA - Remove Archived Material Selected by Archive . . . 410
RAMP - Remove Archived Material Selected by Product . . . 411
RAI - Remove Archived Item . . . 412
RAT - Read Archive Tape . . . 413
ROA - Retrieve Offline Archive . . . 414
TBO - Transfer Baseline Out . . . 415
TBI - Transfer Baseline In . . . 417
Magnetic Tape Utility . . . 419

Chapter 32 **Viewing Archives and Transfers . . . 421**
About Archives and Transfers . . . 422
Archives and Transfers Main Window . . . 422
Archives and Transfers Menu Area . . . 422
Archives and Transfers Navigation Area . . . 422
Archives and Transfers Content Area . . . 423
Archives and Transfer Status Area . . . 423
Viewing Archives and Transfers . . . 424
How to View Archive and Transfer Contents . . . 424

Part 7 OpenText PulseUno . . . 425

Chapter 33 **PulseUno Installation Files . . . 427**
Common Tools Directory . . . 428
PulseUno Data Directory . . . 428
Configuration Settings . . . 429

. 429

Chapter 34 **PulseUno: Mapping Plugin Severity 431**
Checkstyle Plugin. 431
SonarQube Plugin 431
PMD Plugin 432

Part 8 Appendices 433

Appendix A **Configuring Centralized Network Authentication 435**
Overview 436
Introduction 436
About LDAP 437
About PAM 437
About SSO 438
About CAC 439
Password Security Operating System Limitations 441
Authentication Architecture. 441
Native, LDAP, and PAM Authentication Methods 441
Web Tools Authentication without SSO 445
Web Tools Authentication with SSO 446
Eclipse and Visual Studio Authentication with SSO 447
Event Driven Authentication with SSO 448
Web Tools Authentication with SSO/CAC 449
Desktop Client, Eclipse, and Visual Studio Authentication with SSO/CAC 450
Dimensions Support for LDAP 451
Introduction 451
Configuring CM for Secure LDAP 451
LDAP Authentication and Tertiary Node Access 452
Dimensions LDAP Configuration Variables. 452
Dimensions LDAP Examples 454
Example 1 454
Example 2 454
Example 3 454
Example 4 455
Example 5 455
Synchronizing CM and LDAP Groups. 455
Introduction 455
Configuring Synchronization 456
Enabling Synchronization 457
Dimensions Support for PAM 459
Introduction 459
PAM Configuration Variables and Files 459
PAM Example: Configuring Solaris to use CM with Active Directory 460
Introduction 460
Overview of CM Authentication via Active Directory 461
Implementing the Scheme 462
Possible Issues 463

Example Configuration Files 464
Dimensions Support for SSO 467
Introduction 467
Prerequisites for SSO Authentication 467
Supported SSO Scenarios. 469
Configuring a Dimensions Server for SSO. 470
Configuring Trusted Certificate Authorities for SSO Installations 475
Establishing a Certificate Revocation List for CAC Certificates 478
Dimensions Utility for Generating Encrypted Passwords Strings for IDP 479
Configuring the Dimensions Web Client for SSO 480
SSO-Enabled Dimensions Web Client Login 481
Other SSO Logout Functionality for Dimensions Clients 482
Adding a System Administrator as an SBM User 482
SSL Connection Considerations for an SBM SSO Server 482
Updating an IDP Certificate 483
Troubleshooting SSO 484
Dimensions Support for SSO with CAC 485
Introduction 485
Prerequisites for SSO with CAC Reader Authentication 485
Configuration Entries for SSO with CAC Authentication 488
Not Allowing Accounts for CAC-Only Authorized Users 489
Implementing CAC Support after Installing CM Server and SSO 489

Appendix B **Secure Sockets Layer Support. 491**
About SSL. 492
SSL Certificates. 492
Validating SSL Certificates 493
Configuring CM to use SDP over SSL 494
Configuring Web Tools for HTTPS. 498
Initializing the Random Number Generator 498
Troubleshooting. 498
Enable Logging 498
URL Issues 499
Incorrect Host Name 499
Certificate Issues 499

Appendix C **Using a Non-Default Service Name and Port Number 501**
Default Network Service Name and Port Number 502
Setting Non-Default Service Name and Port Number 502
UNIX 502
Windows 504
Command-Line and Desktop Client Considerations. 504
Remote Tertiary Node Considerations. 505

Appendix D **Replicator Utilities 507**
Introduction 508
Dimensions Label Branch Utility. 508
Description 508

Syntax . . . 508
Options . . . 508
Restrictions . . . 509

Appendix E **Replicator Error and Warning Messages . . . 511**

Appendix F **Using and Configuring Library Cache Areas . . . 537**
Introduction . . . 538
Example Scenario . . . 540
Overview . . . 540
Workflow . . . 540
Configuring the Environment . . . 541
Performance Improvements . . . 541
When to use Library Cache Areas . . . 542
Library Cache Area User Types . . . 542
Network Latency . . . 542
Server Proximity . . . 543
Project Characteristics . . . 543
File Prerequisites . . . 543
Work Area Requirements . . . 544
Configuring and Using Library Cache Areas . . . 544
Dimensions CM Privileges . . . 544
Dimensions CM Listener . . . 545
Adding a Physical Network Node . . . 545
Installing the Dimensions CM Listener . . . 545
Starting the Dimensions CM Listener . . . 545
Creating Library Cache Areas . . . 546
Associating Library Cache Areas with Projects or Streams . . . 547
Testing Library Cache Areas . . . 547
Populating Library Cache Areas . . . 547
Updating Library Cache Area Properties . . . 548
Deleting Library Cache Area Definitions . . . 548
Purging Library Cache Areas . . . 548
Optimizing Library Cache Area Performance . . . 549
Using the Command-Line to Manage Library Cache Areas . . . 549

Appendix G **Executing Operations on Remote Nodes . . . 551**
Overview . . . 552
Remote Job Flow . . . 552
Step 1 Submit REXEC, BLD, or BLDB Command . . . 553
Step 2 Dimensions CM Server Processing . . . 553
Step 3 Dimensions Network Node Processing . . . 553
Step 4 Dimensions Server Processing (Optional) . . . 553
Executing Jobs on Remote Nodes . . . 554
Preserving Job Logs . . . 554
Controlling Log Preservation for REXEC Jobs . . . 554
Controlling Log Preservation for Deployment Scripts . . . 555

Appendix H **Schedule Job Utilities and Configuration Parameters 557**
About Schedule Jobs 558
dmschedule Utility 558
Modes of Operation 558
Security 558
Command-Line Syntax 558
Schedule Job Parameters in dm.cfg 559
Schedule Job Configuration Parameters 560
Contents of the Configuration File 561

Appendix I **Migrating MBCS Data to UTF-8 Data 565**
Chapter Overview 566
Disclaimer 566
DMDBA Migrate Utility Features and Prerequisites 566
Overview of the Migration Process 567
DMDBA Migrate Syntax 567

Appendix J **Logging and Tracing Facilities 569**
Chapter Overview 570
Displaying Connection Pooling Statistics 570
Enabling Dimensions CM Listener Tracing 570
Logging ALF Events 570
Logging Schedule Jobs 571
Logging Web Services 571
Managing Replication Logs 572
Tracing API Events and Merge Events 572
Tracing Template Processing 573

Appendix K **VRS Data Caching 575**
Overview 576
Pre-Caching VRS Data 576

Part 1

General Administration

Configuring Web Aspects of OpenText™ Dimensions CM Server	25
Setting Dimensions Server and Client UNIX Symbols	39
Setting Dimensions Server Windows Symbols	73
Setting Dimensions Windows Client Symbols	91
Administering Licenses and the License Server	95
Configuring and Using Auto Update	97

Chapter 1

Configuring Web Aspects of OpenText™ Dimensions CM Server

Introduction	26
Starting and Stopping Web Tools	27
Running Web Tools with Alternative Web Servers	29
Configuring Microsoft IIS to Work with Tomcat	30
Installing Native Components on a Windows or UNIX Client Node	30
Starting the Web Client on a Client Node	30
Reassigning Default Web Client Connection Parameters	31
Configuring Web Client Tools	34
Support for Your Own Web Application Container	35
Problems with Unicode Characters in File Names	37

Introduction

NOTE In this section, the location of the Tomcat directory in the installation directory structure is referred to as <Tomcat-Root>.

- **Windows:** <Tomcat-Root> is:

  %Program Files%\OpenText\common\tomcat\<tomcat-version>

- **UNIX:** <Tomcat-Root> is

  $DM_ROOT/../common/tomcat/<tomcat-version>/

  where $DM_ROOT is the location of the Dimensions CM for UNIX server installation.

The Tomcat software is located relative to a directory whose root is one level up from that used for Dimensions CM.

Overview

This section provides instructions for configuring and running the Dimensions CM server-side components of installed Dimensions CM web tools (Dimensions CM web client and Administration Console):

- Starting and stopping the Dimensions CM Administration Console.
- Configuring the supported web server to work with the Dimensions CM web tools.
- Installing and initially running the Dimensions CM web client or Administration Console on a client node.

Web Client Components

The combined services of the Dimensions CM web client server software, the web server, and the Dimensions CM web client applet enable you to access Dimensions CM features remotely. Each component is described below:

- **Dimensions CM web client server software:** Dynamically generates the HTML pages you view from the Dimensions CM web client and processes the tasks you initiate from the Dimensions CM web client.
- **Tomcat:** Web application container that enables the Dimensions CM web client to run. It can be used standalone or in conjunction with any of the supported web servers.
- **Web server:** Enables information to be passed between the Dimensions CM web client server and the Dimensions CM web client through HTTP. The web server must be running before you can access the Dimensions CM web client. If you are using an external web server, it must be running.
- **Dimensions CM web client applet:** Provides access to Dimensions CM features through a browser-based interface.

- **Single Sign-On (SSO) Components**: Security Server Identity Provider (IDP) and SSO Gatekeeper.

IMPORTANT! Check the Support Matrix to confirm that SSO and configuring the remote Windows Common Access Card (CAC) smart cards is supported on your platform.

Starting and Stopping Web Tools

Introduction

To start the Dimensions CM web tool server components, you start Tomcat, the servlet engine that enables the web applications to run. You can:

- Use Tomcat standalone and access it at the port number specified during installation.

  —or—

- Configure the supported web servers to work in conjunction with Tomcat, see "Configuring Microsoft IIS to Work with Tomcat" on page 30.

  —or—

- Use your own supported web application container.

You must start Tomcat and any configured web server on the server machine before the Dimensions CM web tools can be accessed in a web browser. After you have started Tomcat, it continues to run until you shut it down manually or shut down the machine.

NOTE Start and stop Tomcat when logged in as the user who owns the Dimensions CM web tools directory, normally the Dimensions CM owner, which is `dmsys` by default. Running these tasks under other user accounts may result in errors due to inconsistent file permissions, especially on UNIX.

During a Dimensions CM installation on UNIX, you need to nominate the user account for starting or stopping Tomcat.

Starting Tomcat

NOTES

- The first time Tomcat starts after a Dimensions CM installation, various post-installation tasks are performed, which can take time to complete. While this is taking place, make sure not to shut down Tomcat. Subsequent restarts of Tomcat happen much faster.
- When you stop Tomcat, Dimensions CM web tools become unavailable.

Windows

Automatic Startup of Tomcat as a Windows Service Through the Windows Service Manager

The Dimensions CM installation default is for a Windows service called **OpenText Common Tomcat** to be set to `Automatic` so that Tomcat is started automatically every

time Windows is started. You should allow this service to start up before trying to access any of the Dimensions CM web tools.

Once the startup procedure completes, you can access the Dimensions CM web client in your web browser on your client machine. See "Starting the Web Client on a Client Node" on page 30 for instructions.

To check or start the OpenText Common Tomcat service:

1 On the Dimensions CM server machine, go to:

```
Start | Windows Administrative Tools | Services
```

2 Locate the OpenText Common Tomcat service in the list of services and check that it has started.

3 To manually start the service (if for any reason it has not started automatically), right-click the service name and then select `Start`.

Windows

Starting Tomcat by Starting Its Windows Service Through an Operating System Command

As an alternative to manually starting the Tomcat Windows service as described above, you can also start the service be using the following operating system command:

```
C:\ net start OpenTextTomcat
```

UNIX

Manually Starting Tomcat on a Dimensions CM for UNIX Server

To start Tomcat:

1 Log in as the Dimensions System Administrator (DSA). By default this is `dmsys`.

CAUTION! Do not start Common Tomcat as user `root`. It must be started as the DSA to function correctly.

2 Set the Dimensions CM environment variables for yourself by running the appropriate Dimensions CM login script. For example:

- Bourne Shell

```
$ cd /opt/opentext/dimensions/<version>/cm
$ . ./dmprofile
```

- C Shell

```
$ cd /opt/opentext/dimensions/<version>/cm
$ source ./dmlogin
```

3 Change directory to:

```
$ cd <Tomcat-Root>/bin
```

4 Start Common Tomcat by running the following command:

```
$ ./startup.sh
```

5 Check that Common Tomcat is running by using the following command:

```
$ ps -eaf | grep tom
```

You can now access the Dimensions CM web client and Administration Console from your web browser.

Stopping Tomcat

Stopping Tomcat on Windows

You can stop the OpenText Common Tomcat service through the Windows Service Manager, or use an operating system command, for example:

```
C:\ net stop OpenTextTomcat
```

Manually Stopping Tomcat on a Dimensions CM for UNIX Server

Enter `./shutdown.sh` from the `<Tomcat-Root>/bin` directory.

Installing or Removing the Tomcat Windows Service

To (re-)install or remove the Tomcat Windows service, execute the appropriate batch file:

```
<Tomcat-Root>\bin\install_service.bat
<Tomcat-Root>\bin\uninstall_service.bat
```

Running Web Tools with Alternative Web Servers

The following terms appear in the remainder of this chapter:

- A *Server* is a single XML element in the `server.xml` configuration file that represents the entire Java Virtual Machine (JVM), which may contain one or more *service* instances. The server listens for a shutdown command on the indicated port.
- A *Service* is a collection of one or more *Connector elements* that share a single *container* (and therefore the web applications visible within that container). Normally, that container is an "Engine".
- A *Connector element* represents an endpoint by which requests are received and responses returned. Each connector passes requests on to the associated container for processing. For the default Dimensions CM Tomcat installation, there is a connector at port 8080.

Dimensions CM supports running Dimensions CM web tools with HTTP connectors that do one of the following:

- Enable Tomcat to work as a standalone web server.
- Enable the use of other web servers (for example, Microsoft IIS) along with Tomcat to serve HTTP contents to your web browser. In such configurations, the web server and Tomcat communicate with each other using Tomcat's web server connectors.

For details on how connectors integrate with web servers and particularly the Apache web server, see:
http://www.idevelopment.info/data/Programming/web/connecting_apache_tomcat/Web_Server_Connectors.shtml

It also summarizes the reasons why you might want to use another web server in conjunction with Tomcat.

Configuring Microsoft IIS to Work with Tomcat

Download the Tomcat connector and see the following configuration information:

https://tomcat.apache.org/connectors-doc/webserver_howto/iis.html

https://tomcat.apache.org/connectors-doc/reference/iis.html

See also this Microsoft website: https://www.iis.net/

Installing Native Components on a Windows or UNIX Client Node

To be able use certain Dimensions CM subcomponents, you must install "native components" by clicking the **native components** link on the Dimensions CM web client login page and clicking the **here** link on the associated web page to run the installer. Installation of native components is required:

When using all supported combinations of Dimensions web browsers and JREs: To install the native components required to use the Merge Tool.

For details about installing native components, see the *Dimensions CM online help*.

Starting the Web Client on a Client Node

To start the Dimensions CM web client on a client node:

1. Confirm that Tomcat is running on the Dimensions CM server.
2. Open your web browser.

Windows or UNIX

3. Open the URL of Dimensions CM web client:

   ```
   http(s)://hostname:port/dimensions/
   ```

   where `hostname` is the name of the server system hosting the Dimensions CM web client software and `port` is the web server port number.

Windows-only alternative

4. From the Start menu, select the following:

   ```
   Dimensions CM <version> | Web Client
   ```

   This link brings up your web browser with the correct server and port.

For details on using the Dimensions CM web client, see the online help.

Reassigning Default Web Client Connection Parameters

Introduction

The Dimensions CM web client and Administration Console have associated `web.xml` files with initialization parameter elements of the form

```
<init-param>
   <param-name>NAME</param-name>
   <param-value>VALUE</param-value>
</init-param>
```

and are located in the Dimensions CM web tools directory:

Windows

```
<Tomcat-Root>\webapps\adminconsole\WEB-INF\web.xml
<Tomcat-Root>\webapps\dimensions\WEB-INF\web.xml
```

UNIX

```
<Tomcat-Root>/webapps/adminconsole/WEB-INF/web.xml
<Tomcat-Root>/webapps/dimensions/WEB-INF/web.xml
```

See also "Setting Non-Default Service Name and Port Number" on page 502.

You can configure these files to make the web tools behave in a non-default manner as described in "Reassigning the Default listener Port Number" on page 31 and "Changing or Adding Connection Strings or Parameters" on page 33.

Reassigning the Default listener Port Number

NOTE After the process below is completed and Tomcat is restarted, you can log in to the Dimensions CM web client or Administration Console using a non-default TCP/IP port number (for example, `5000`) in the same manner as when using the default `671` value.

By default, the Dimensions `listener` process is set up to work on TCP/IP port number `671`. If your Dimensions CM installation was customized so that the listener uses a TCP/IP port number other than `671`, the Dimensions CM Administrator must edit various `web.xml` files, as explained below, to customize the port number before the Dimensions CM web client and Administration Console software works in default **Sample database** mode.

Editing the `dm_host` name

After shutting down Tomcat, open the `web.xml` files specified in "Introduction" on page 31 and then change the initialization parameter that has the `<param-name>` of `dm_host`:

```
your_Dimensions_listener_host_name
```

becomes

```
your_Dimensions_listener_host_name>:<port_number>
```

For example:

```
<init-param>
   <param-name>dm_host</param-name>
   <param-value>dmserver</param-value>
</init-param>
```

needs to be changed to

```
<init-param>
   <param-name>dm_host</param-name>
   <param-value>dmserver:5000</param-value>
</init-param>
```

if the machine with the Dimensions listener were called `dmserver` and the port number of the listener were `5000`.

Editing the `dimensions_systems` parameter

As well as editing the `dm_host` parameter, you must also edit the `dimensions_systems` parameter, found near the end of the `web.xml`, as follows:

Edit the customized initialization parameter that has the `<param-name>` of `dimensions_systems` so that its value is changed from

```
your_Dimensions_listener_host_name
```

to

```
your_Dimensions_listener_host_name>:<port_number>
```

For example:

```
<init-param>
   <param-name>dimensions_systems</param-name>
   <param-value>Demonstration
    database|INTERMEDIATE|DIM10|dmserver</param-value>
   <description>
    Comma-separated list of Dimensions systems to show on the log in
    page.
     Systems are represented as a pipe-separated list of strings,
    like
     &quot;System-Name|Database-Name|Connection-Alias|Server-
    Hostname&quot;.
     For example, &quot;Demonstration
    database|INTERMEDIATE|DIM10|localhost,
     Production DB|CORPDB|DMDSN|dmserver:672&quot;.
     Customized by installer.
     Supersedes dm_host and dm_connalias.
   </description>
  </init-param>
```

needs to be changed to

```
<init-param>
   <param-name>dimensions_systems</param-name>
   <param-value>Demonstration
    database|INTERMEDIATE|DIM10|dmserver:5000</param-value>
   <description>
    Comma-separated list of Dimensions CM systems to show on the log
    in page.
     Systems are represented as a pipe-separated list of strings,
    like
     &quot;System-Name|Database-Name|Connection-Alias|Server-
    Hostname&quot;.
     For example, &quot;Demonstration
    database|INTERMEDIATE|DIM10|localhost,
     Production DB|CORPDB|DMDSN|dmserver:672&quot;.
```

```
        Customized by installer.
        Supersedes dm_host and dm_connalias.
      </description>
    </init-param>
```

if the machine with the Dimensions listener were called **dmserver** and the port number of the listener were **5000**.

Changing or Adding Connection Strings or Parameters

NOTE When the process below is completed and Tomcat is restarted, you can log in to the Dimensions CM web client or Administration Console using non-default or additional connection strings or parameters. These modified or additional connection strings or parameters are available through entries in the **Connect to** list.

As well as editing the `dimensions_systems` parameter to change the port number as described in "Editing the dm_host name" on page 31, you can also edit it or add comma-separated strings to it to modify or add additional entries to the **Connect to** drop-down list. For example:

```
<init-param>
   <param-name>dimensions_systems</param-name>
   <param-value>Demonstration
    database|INTERMEDIATE|DIM10|dmserver</param-value>
   <description>
    Comma-separated list of Dimensions CM systems to show on the log
    in page.
     Systems are represented as a pipe-separated list of strings,
    like
     &quot;System-Name|Database-Name|Connection-Alias|Server-
    Hostname&quot;.
     For example, &quot;Demonstration
    database|INTERMEDIATE|DIM10|localhost,
     Production DB|CORPDB|DMDSN|dmserver:672&quot;.
     Customized by installer.
     Supersedes dm_host and dm_connalias.
   </description>
  </init-param>
```

could be changed to

```
<init-param>
   <param-name>dimensions_systems</param-name>
   <param-value>Sample
    database|QLARIUS_CM|DIM10|dmserver:5000,Production|ACME|
    ACME10|prod-server</param-value>
   <description>
    Comma-separated list of Dimensions CM systems to show on the log
    in page.
     Systems are represented as a pipe-separated list of strings,
    like
     &quot;System-Name|Database-Name|Connection-Alias|Server-
    Hostname&quot;.
     For example, &quot;Demonstration
```

```
        database|INTERMEDIATE|DIM10|localhost,
         Production DB|CORPDB|DMDSN|dmserver:672&amp;quot;.
         Customized by installer.
         Supersedes dm_host and dm_connalias.
       </description>
      </init-param>
```

This results in the following **Connect to** strings, in addition to those you can always manually enter through **Manual...**:

- Sample database:

  Server: dm_server:5000

  DB Name: QLARIUS_CM

  DB Connection: DIM10

- Production:

  Server: prod-server

  DB Name: ACME

Configuring Web Client Tools

The web client uses Web Client Tools for version management operations such as update and deliver.

NOTE Web Client Tools are not supported on Linux and masOS clients.

Configuring Web Client Tools

To specify how Web Client Tools communicate with the CM server, set the following parameters in the web client configuration file:

`<Tomcat-Root>/webapps/dimensions/WEB-INF/web.xml`

Network Protocol

Specify one of the following network protocols in the `tunnelingProtocol` parameter:

- Standard Dimensions Protocol: `sdp`
- HTTP protocol: `http`
- HTTPS protocol: `https`

For example:

```
<init-param>
   <param-name>tunnelingProtocol</param-name>
   <param-value>http</param-value>
</init-param>
```

Server Port Number

Specify the server port number in the `tunnelingPort` parameter, typically one of the following:

- SDP: `671`
- HTTP: `80`
- HTTPS: `443`

For example:

```
<init-param>
   <param-name>tunnelingPort</param-name>
   <param-value>80</param-value>
</init-param>
```

If you specify HTTP/HTTPS, use the same port as the *Dimensions CM Server HTTP Connector*.

Disabling Web Client Tools

Web Client Tools are enabled by default. To disable them, change the value of the parameter `enable_web_tools` to 'no' in `web.xml`:

```
<init-param>
   <param-name>enable_web_tools</param-name>
   <param-value>no</param-value>
</init-param>
```

Support for Your Own Web Application Container

CAUTION! Only experienced administrators should attempt to install Dimensions CM into their own web application container (or application server). Serena generally recommends using the Tomcat container that can be installed as part of Dimensions CM.

It is possible to install the Dimensions CM web client and Administration Console into application containers other than the Tomcat container that can be installed with Dimensions CM. This enables you to adhere to in-house corporate standards where applicable.

Instructions on how to perform this installation for specific versions of Tomcat and WebShere are detailed in the following subsections.

Installing Web Tools into Tomcat

This subsection explains how to install the Dimensions CM web tools into an existing Tomcat application server.

IMPORTANT! This section assumes that you are familiar with deploying web applications into Tomcat.

Pre-installation Requirements

Before installation, you need the following:

1. A correctly installed Tomcat.
2. For the Dimensions CM web client, the following files:
   - dimensions.war
   - dim_applet.war
   - dim_images.war
   - inet_webhelp.war
3. For the Dimensions CM web client Administration Console:
   - adminconsole.war
   - admin_applet.war
   - admin_images.war
   - admin_webhelp.war
   - poolstats.war

NOTE By default, if the web application container is located on a different machine to the Dimensions listener, the Poolstats service is not available. To make the service available in such circumstances, perform the following steps to tell Poolstats the location of the listener:

1. Shut down Tomcat.
2. With a text editing tool that understands `<LF>` (UNIX) line endings, open the following file:
   - Windows: `<Tomcat-Root>\webapps\poolstats\WEB-INF\web.cml`
   - UNIX: `<Tomcat-Root>/webapps/poolstats/WEB-INF/web.cml`
3. Locate the line

   ```
   <param-value>localhost</param-value>
   ```

4. Change `localhost` to the host name of the server on which the listener is running, for example:

   ```
   <param-value>dmserver/param-value>
   ```

   and save your changes.
5. Restart Tomcat.

Performing the Installation

1. Ensure that the Tomcat server is shut down.
2. Copy whichever WAR files you want to install to:

Windows `<Tomcat-Root>\webapps/`

UNIX

```
<Tomcat-Root>/webapps/
```

3. Start the Tomcat server. This extracts each WAR file into a subfolder of `webapps` giving this folder the name of the WAR. Then stop the server again.

4. Edit `web.xml` for `dimensions` (Dimensions CM web client) and `adminconsole` (Administration Console). These files are here:

Windows

```
<Tomcat-Root>\webapps\adminconsole\WEB-INF\web.xml
<Tomcat-Root>\webapps\dimensions\WEB-INF\web.xml
```

UNIX

```
<Tomcat-Root>/webapps/adminconsole/WEB-INF/web.xml
<Tomcat-Root>/webapps/dimensions/WEB-INF/web.xml
```

5. You might also need to edit `web.xml` for `dimensions and adminconsole` to ensure that the values for the parameters `debug_file`, `stdout_file`, and `stderr_file` include absolute paths rather than relative paths. The `web.xml` files are here:

Windows

```
<Tomcat-Root>\webapps\adminconsole\WEB-INF\web.xml
<Tomcat-Root>\webapps\dimensions\WEB-INF\web.xml
```

UNIX

```
<Tomcat-Root>/webapps/adminconsole/WEB-INF/web.xml
<Tomcat-Root>/webapps/dimensions/WEB-INF/web.xml
```

6. If you need to enable `Poolstats`:

   a. Edit `web.xml` in `poolstats` so that the `exec` parameter shows the full path to `getpoolstats`; for example:

Windows

```
%DM_ROOT%\prog\getpoolstats.exe /html
```

`web.xml` can be found in:

```
<Tomcat-Root>\webapps\poolstats\WEB-INF/
```

UNIX

```
$DM_ROOT/prog/getpoolstats /html
```

`web.xml` can be found in:

```
<Tomcat-Root>/webapps/poolstats/WEB-INF/
```

UNIX only

   b. Add:

```
$DM_ROOT/lib
```

to the library path environment variable.

7. Restart the Tomcat server.

Dimensions CM web client and Administration Console are now correctly installed.

Problems with Unicode Characters in File Names

If Unicode characters in file names are not preserved correctly in a UTF-8 database on UNIX installations, you can configure Tomcat to set a locale. To perform this configuration, proceed as follows:

1. Open the following file in an appropriate UNIX text editor

```
<Tomcat-Root>/bin/setenv.sh
```

2 Locate the section that looks like this:

```
# Uncomment the following if you encounter a problem delivering
    files with
# Unicode characters in their filenames (to force UTF-8 character
    encoding).
# Use 'locale -a' to check that installed locales include
    en_US.UTF-8.

#LANG=en_US.UTF-8
#LC_CTYPE=
#LC_ALL=
#export LANG
#export LC_CTYPE
#export LC_ALL
```

3 If `en_US.UTF-8` is installed (and available) on the server operating system, uncomment the above assignments as directed by the comments. However, if `en_US.UTF-8` is not installed (and available) on the server operating system, you need to choose another appropriate Tomcat locale that is installed and available on the server operating system (and which uses the UTF-8 character encoding).

4 Tomcat needs to be restarted before any changes become operative.

NOTE The above commented-out assignments cannot be present in `setenv.sh` as default assignments because they may not work on all operating system configurations. It is your responsibility to install and select an appropriate UTF-8 Tomcat locale on the system if you have the problem with Dimensions `deliver` discussed here, and use this in `setenv.sh`.

Chapter 2

Setting Dimensions Server and Client UNIX Symbols

Introduction	40
Modifying or Adding dm.cfg Entries	41
Installation and File Location Symbols	42
Using a Proxy Server	42
Sending Email to Dimensions Users	44
Paging Data	46
Paging Data	46
Printing Data	47
Cleaning Up Temp Files Created when Running User Reports	48
Merging Item Revisions	48
Networking and Replication	49
Code Page Settings	49
Oracle Collating Sequence for 'ORDER BY' Queries	50
Dimensions Web Tools URL Parameters	51
Overriding Host Name Assigned to DMSERVER Certificates	51
Additional Dimensions Configuration Symbols	52
Dimensions for z/OS Configuration Symbols	68
Defaults for Copying Build Information	69
SSO Configuration Symbols	70
CAC Configuration Symbols	71

Introduction

Dimensions CM Operating System Environment Variables

The script `dmlogin` (C-shell) or `dmprofile` (Bourne-shell) located in directory `$DM_ROOT` sets a number of standard operating system environment variables and scripts for every user of that particular Dimensions CM server or client installation. These are primarily with respect to Dimensions CM path and access parameters.

You can, on per-user basis, override a particular installation-wide variable assignment by making alternative local assignments through the `.login` or `.profile` login file or at the operating system command prompt.

The Dimensions dm.cfg File

Each Dimensions CM UNIX server or client installation contains a `dm.cfg` configuration file located in the directory `$DM_ROOT`. This file specifies Dimensions CM UNIX configuration symbols for that particular installation.

If the installation is part of a larger network of Dimensions CM installations, make sure that configuration settings affecting communication with other installations are consistent.

The directory `$DM_ROOT` is set during the installation and is the default Dimensions CM configuration for all users of the machine.

Settings in the `dm.cfg` file are installation-wide. Parameters that are to be specific to you must be set in your own particular environment. If you modify settings both in the `dm.cfg` configuration file and in your own particular environment, your environment takes precedence.

These configuration parameters and environment variables are collectively referred to as the **Dimensions CM symbols**. Most Dimensions CM symbols are supported on all the operating systems.

This section describes only those symbols that are recommended for modification.

Dimensions CM symbols and scripts are categorized as follows:

- Installation and location symbols
- Using a proxy server
- Sending email to Dimensions CM users
- Editing data
- Browsing data (usually by invoking an editor in read-only mode)
- Printing data files
- Cleaning up temporary files
- Showing item differences
- Merging item revisions
- Networking and replication

- Code page settings
- Command security
- Oracle collating sequence
- Dimensions CM web tools URL parameters
- Miscellaneous Dimensions CM configuration symbols

By default, Dimensions CM makes use of standard system tools. But you can specify to use third party tools instead, for example, installed proprietary packages for word processing or publishing.

Additional Dimensions Configuration Symbols

In addition to the standard Dimensions CM UNIX configuration symbols mentioned earlier, Dimensions CM provides specialized configuration symbols that are either specified in `dm.cfg` or can be added. For details, see "Additional Dimensions Configuration Symbols" on page 52.

Modifying or Adding dm.cfg Entries

CAUTION! When modifying or adding `dm.cfg` symbols, make sure that the entries do not contain trailing whitespace characters, as this may result in an error. For example, an entry of "`DM_PROG<tab>%DM_ROOT%prog\<space>`" results in failure to connect to a Dimensions CM application.

Most installation-wide entries in `dm.cfg` should not be changed by end users. Remember that these installation-wide values can always be superseded in each user's environment by setting the appropriate environment variable.

IMPORTANT! Typically, for changes in `dm.cfg` to take effect, you need to restart the Dimensions listener after updating `dm.cfg`.

The following symbols in `dm.cfg`, and a few outside `dm.cfg`, may be changed or overridden by environment settings if needed. For more details about each parameter, see the relevant page.

dm.cfg symbols	page
DM_TMP	42
DM_REMOTE_FILE_AREA	42
DM_MAIL	45
DM_MAIL_SCRIPT	46
DM_DMEMAIL	46
MORE (outside **dm.cfg**)	46
DM_PRINT	47
DM_CHD_PRINT	47

dm.cfg symbols	page
DM_DELETE	48
MERGE_TOOL/DIFF_TOOL/MERGE_PARAMS/ DIFF_PARAMS	48
DM_CODEPAGE	49
DM_EBCDIC_CODEPAGES	50
DM_DEFAULT_CODEPAGE	50
DM_NLS_SORT_PARAM	50
DM_WEB_URL	51
DM_INET_PORT *(backward compatibility only)*	51
DM_INET_SERVER *(backward compatibility only)*	51

Installation and File Location Symbols

These symbols are defined in the installation `dm.cfg` file and are specific to the installation. Normally, you do not need to modify them. Two of note are `DM_TMP` and `DM_REMOTE_FILE_AREA`.

DM_TMP

This symbol is included in the installation `dm.cfg` file and is specific to the installation. It controls the location of a temporary area for general tools.

DM_REMOTE_FILE_AREA

For the primary DB site and all the replica DB sites, you can define the temporary work area that Replicator uses when sending and receiving items by setting the `DM_REMOTE_FILE_AREA` symbol on each primary and replica machine. Replicator does not run unless this variable is set.

By default, this symbol is set to:

- **UNIX:** `/tmp`
- **Windows:** `%DM_ROOT%\remote_file_area`

Using a Proxy Server

You can set up Dimensions CM to use a proxy server for external connections:

- Tunneled connections via HTTP(S) from the Dimensions CM clients to the server.
- Connections to the integrating tools such as external request providers, Deployment Automation, and PulseUno.

- Connections to the AutoPass License Server (APLS).

To configure your Dimensions CM installation for a proxy, add the following symbols to `dm.cfg`:

Symbol	Description
`DM_NET_PROXY_MODE`	Specifies a list of external tools and applications to which to connect via a proxy server.
`DM_NET_PROXY`	Specifies the proxy server through which to route traffic.
`DM_NET_NOPROXY`	Specifies a list of exclusions for which you do not want to use a proxy.

These symbols are described next.

DM_NET_PROXY_MODE

Use the `DM_NET_PROXY_MODE` symbol to specify one or more servers or applications to which to connect through a proxy server.

Enter a list of applications, separating each entry with a space.

The following table lists the possible values:

Value	Description
`TUNNEL` (or legacy `TUNNEL_AUTO`)	HTTP(S)-tunneled connections from the Dimensions CM clients to the server.
`PULSE`	Connections to PulseUno.
`OCTANE`	Connections to ALM Octane for working with external requests.
`JIRA`	Connections to Jira for working with external requests.
`SDA`	Connections to Deployment Automation for continuous deployment.
`APLS`	Connections to the AutoPass License Server (APLS) for managing licenses.
`AUTO`	All connections listed in this table, except the AutoPass License Server (APLS).

Examples:

To use a proxy for connections to the ALM Octane and Jira request providers, add the following line to the `dm.cfg` file on the server:

```
DM_NET_PROXY_MODE OCTANE JIRA
```

To use a proxy for all connections, add the following line to the `dm.cfg` file on the server:

```
DM_NET_PROXY_MODE AUTO APLS
```

To avoid using a proxy for tunneled connections from a client to the server, add the following line to the `dm.cfg` file on the client:

```
DM_NET_PROXY_MODE N
```

DM_NET_PROXY

Use the DM_NET_PROXY symbol to define the URL of the proxy server through which to route traffic for specified applications. Enter the URL in the following format:

<scheme>://<server>:<port>

where:

- **<scheme>** is the protocol scheme, such as HTTP(S).
- **<server>** is the host name of the proxy server.
- **<port>** (optional) is the port number for the proxy server.

For example:

```
DM_NET_PROXY https://proxyURL:1383
DM_NET_PROXY socks5h://proxyURL:1080
```

DM_NET_NOPROXY

Use the DM_NET_NOPROXY symbol to define hosts that should not use a proxy.

Guidelines:

- For multiple hosts, separate each host with a space, comma (,), or semicolon (;).
- To exclude all local (intranet) hosts, use the value <local>.

Examples:

To exclude all hosts in the **intranet.com** domain, such as **my.intranet.com** and **my.other.intranet.com**, add the following line:

```
DM_NET_NOPROXY intranet.com
```

To exclude all local hosts and all hosts in the **example.com** domain, add the following line:

```
DM_NET_NOPROXY <local>,example.com
```

Sending Email to Dimensions Users

The following traditional UNIX Dimensions CM symbols control email:

Symbol	Description
DM_MAIL	Specifies the Dimensions CM email tool.
DM_MAIL_SCRIPT	Specifies a shell script that provides an interface to the email tool specified by DM_MAIL.

The use of these variables is described next.

DM_MAIL

The value of the DM_MAIL symbol specifies the name of the email tool used to communicate information to users. Many Dimensions CM commands email users when certain events have taken place; for example:

- When Dimensions CM products are created, administrators are notified through email as to how they can start using the products.
- Whenever an item revision or request is awaiting the attention of specific Dimensions CM users as a result of a change in its lifecycle state, the users are notified through email.
- Some Dimensions CM commands run batch scripts that email users regarding the outcome of the job on termination.

Examples of email tools include:

- email automatic (BDS and HP-UX UNIX)
- mailx automatic (System V UNIX)
- mailtool interactive (OpenWindows)

Default:

UNIX (C shell):

```
setenv DM_MAIL /usr/ucb/mail
```

UNIX (Bourne shell):

```
DM_MAIL="/usr/ucb/mail"; export DM_MAIL
```

Syntax:

```
/usr/ucb/mail -s <subject> -f <filename> <user list>
```

HP-UX 11

Email generated by Dimensions CM on HP-UX has incorrect MIME encoding. HP-UX tries to determine the MIME character set from the user's locale if one has not been explicitly specified in /usr/share/lib/mailx.rc or $HOME/.mailrc. Unfortunately, it produces "charset=X-roman8" for the C locale and "charset=X-iso8859-1" for the en_GB.iso88591 locale.

MIME aware-email programs on other systems do not understand this—they are looking for "charset=iso-8859-1" for ISO 8859-1.

The workaround is to change the Dimensions CM initialization scripts so that mailx is passed the '-m' flag on HP-UX. This prevents a MIME header being generated. The scripts are in the $DM_MMI directory and are called:

```
dm_init_cts.csh
    dm_init_cts.sh
    dm_init_vmb.csh
    dm_init_vmb.sh
```

DM_MAIL_SCRIPT

This Dimensions CM symbol provides an interface to a user-specified email tool. The information to be emailed is contained within the file passed to the script. It is left to the script to delete the file at the end of processing.

Default:

UNIX (C shell):

```
setenv DM_MAIL_SCRIPT $DM_MMI/dm_mail.csh
```

UNIX (Bourne shell):

```
DM_MAIL_SCRIPT=$DM_MMI/dm_mail.sh; \
    export DM_MAIL_SCRIPT
```

Syntax:

```
$DM_MAIL_SCRIPT file -s subject user1 user2 ...
```

DM_DMEMAIL

For details about this Dimensions CM symbol, the associated subsymbols DM_DMEMAIL_*, the dmemail utility itself, and notification templates, the *online help*.

Disabling Email Notifications on UNIX Systems

To disable email notifications on UNIX systems:

- For the Bourne Shell:

  Add the following lines to the *end* of `$DM_ROOT/dmprofile`

  ```
  DM_MAIL="!"; export DM_MAIL
  DM_MAILS="!"; export DM_MAILS
  ```

- For the C Shell:

  Add the following lines to the *end* of both `$DM_ROOT/dmlogin` and `$DM_ROOT/dmprofile`

  ```
  setenv DM_MAIL "!"
  setenv DM_MAILS="!"
  ```

IMPORTANT! For the C Shell you must update both `$DM_ROOT/dmlogin` and `$DM_ROOT/dmprofile` to ensure that email notifications are disabled.

Paging Data

The `MORE` environment variable (which is outside of `dm.cfg`) provides an interface to a pager tool allowing files to be viewed page by page or line by line.

Examples of Pagers are:

more
imagetool-image viewer
pageview-PostScript viewer

Default:

UNIX (C shell):

```
setenv MORE more
```

UNIX (Bourne shell):

```
MORE="more"; export MORE
```

Syntax:

```
more <filename>
```

Printing Data

DM_PRINT

This Dimensions CM symbol provides an interface to a print tool. When Dimensions CM commands require printing of various reports they call this interface.

Examples of printer tools include:

lpr, pr, printtool, printps and enscript

Default:

UNIX (C-shell):

```
setenv DM_PRINT "lpr -p"
```

UNIX (Bourne-shell):

```
DM_PRINT="lpr -p"; export DM_PRINT
```

Syntax:

```
lpr -p <filename>
```

NOTE If you prefer *not* to have Dimensions CM automatically print out reports, set the DM_PRINT symbol to "echo".

DM_CHD_PRINT

This symbol is used to specify a printer tool specifically for request operations. It is similar to DM_PRINT described above.

Cleaning Up Temp Files Created when Running User Reports

The DM_DELETE Dimensions symbol is used in conjunction with running Dimensions User Reports. The value specified in the dm.cfg file is used to specify the tool/command to be used to clean up temporary files created during the construction of the script files that run a particular user report.

Default:

UNIX (C-shell):

```
setenv DM_DELETE "rm -f"
```

UNIX (Bourne-shell):

```
DM_DELETE="rm -f"; export DM_DELETE
```

Syntax:

```
rm -f <filename>
```

Example:

The following is an example of temporary files generated when running the reports:

```
@echo off
rem invoke user report - generated 11:42:17 7 March 2023
call /opt/opentext/dimensions/<version>/cm/temp/pt9344/myreport.out
    "PAYROLL"
rm -f /opt/opentext/dimensions/<version>/cm/temp/pt9344/myreport.out
rm -f pcms9342
```

Merging Item Revisions

You can configure different tools for comparing and merging files using the MERGE_TOOL, DIFF_TOOL, MERGE_PARAMS, and DIFF_PARAMS symbols.

To specify other diff/merge tools, use the following configuration file:

```
$DM_ROOT/../common/tomcat/<tomcat-version>/webapps/dim_applet/
    binaries/native.properties
```

This file is created on your machine if you run the native code option when accessing the web client (see also the Dimensions CM online help). If this directory and file do not exist, you must create them.

For details and examples of how to configure this file, see "Connecting from a Windows Client to a Non-Default Listener on UNIX" on page 93.

Networking and Replication

Heterogeneous Networking

The Dimensions Network and Replicator products have a client-server architecture that uses TCP/IP sockets to communicate between processes.

These TCP/IP sockets are administered by assigning a number to a socket name. These assignments must be the same for all installations within the Dimensions Network.

- On UNIX, you set the values in the `/etc/services` file and then use NIS to distribute the assignment around the network.
- On Windows, there are differences between "flavors" of Windows.

Dimensions CM gives administrators the option to place the socket name/number assignment within a configuration file on each node. This file, of course, is not distributed through the network, so the assignments must be made in each configuration file in the Dimensions CM network. Serena recommends that you use the NIS mechanism to define these socket names on UNIX.

DM_SERVICE_PCMS_SDP_TCP

The Network Connection Object, as defined under Network Administration, is the socket name that must be defined. The default names used by Dimensions Network is `pcms_sdp`.

This is defined as follows:

- Services file:
  `pcms_sdp<tab>671/tcp`
- Configuration file:
  `DM_SERVICE_PCMS_SDP_TCP<tab>671`

DM_SERVICES_FILE

As an alternative to the above, the symbol `DM_SERVICES_FILE` can be used to specify the location of a UNIX style services file, for example:

```
DM_SERVICES_FILE<tab>C:\TCP\services.dat
```

Code Page Settings

DM_CODEPAGE

`DM_CODEPAGE` specifies the directory to contain the mapping tables used when translating between EBCDIC and ASCII. Presupplied tables are installed here, and any user-defined mapping tables must be placed here also.

CAUTION! Make sue that user-defined tables do not overwrite the supplied tables.

Example:

```
PDM_CODEPAGE      /usr/dimensions/codepage
```

Default:

```
$DM_ROOT/codepage
```

DM_EBCDIC_CODEPAGES

DM_EBCDIC_CODEPAGES specifies a list of EBCDIC code set identifiers. This information is important as translations are performed only when moving items from an ASCII environment to EBCDIC (or vice versa). It is not possible to detect automatically which code pages are EBCDIC or ASCII.

Example:

```
DM_EBCDIC_CODEPAGES<tab>0037,0500
```

Default:

If no code pages are defined, then they are all assumed to be ASCII.

DM_DEFAULT_CODEPAGE

DM_DEFAULT_CODEPAGE specifies a default code page to be used with connections for which a code page has not been defined. This effectively defines the code page for the system running the Dimensions CM database server.

Example:

```
DM_DEFAULT_CODEPAGE<tab>00819
```

Default:

If not defined, then 00819 is used.

Oracle Collating Sequence for 'ORDER BY' Queries

DM_NLS_SORT_PARAM

If you want to disable or modify the Oracle collating sequence used for 'ORDER BY' queries in an NLS_SORT, locate the following commented-out entries in the dm.cfg file

```
#
# If you wish on Oracle to disable or alter the locale used for
# NLS_SORTing, please uncomment or alter one of the following lines.
#
# - To disable NLS_SORTing locale
#DM_NLS_SORT_PARAM  disable
#
# - To modify the NLS_SORTing
#DM_NLS_SORT_PARAM  <locale>
#
```

Uncomment and edit the appropriate entry.

Dimensions Web Tools URL Parameters

DM_WEB_URL

This Dimensions CM configuration symbol specifies the host-id and port number (8080 by default) of the Dimensions CM web server. It is normally automatically assigned during a Dimensions CM server installation.

DM_WEB_URL has the format:

```
DM_WEB_URL        http://<host-id>:<port-no>
```

DM_INET_PORT

NOTE DM_INET_PORT has been superseded by DM_WEB_URL. DM_WEB_URL also supports HTTPS web containers, unlike DM_INET_PORT, and should, therefore be used in preference. DM_INET_PORT should only be used in the context of continued support of old dm.cfg files—to this end it is commented out by default in dm.cfg.

This Dimensions CM configuration symbol specifies the port number (by default 8080) of the server hosting the Dimensions CM web server.

DM_INET_PORT has the format:

```
DM_INET_PORT      <port-no>
```

DM_INET_SERVER

NOTE DM_INET_SERVER has been superseded by DM_WEB_URL, which supports HTTPS web containers. DM_INET_SERVER should be used only in the context of continued support of old dm.cfg files—to this end, it is commented out by default in dm.cfg.

This Dimensions CM configuration symbol specifies the host ID of the server hosting the Dimensions CM web server:

```
DM_INET_PORT      http://<host-id>
```

Overriding Host Name Assigned to DMSERVER Certificates

The dm.cfg flag DM_CERT_HOSTNAME_ALIAS overrides the host name assigned to the DMSERVER variable and generated certificates. By appropriately setting this flag, you can reconfigure your Dimensions CM server installation so that the DMSERVER variable and the generated certificates use a logical (DNS) name instead of the physical host name of the Dimensions CM server.

Restart the Dimensions CM listener after making changes.

Additional Dimensions Configuration Symbols

Adding Template Variables

You can add Dimensions CM template variables to `dm.cfg`. The following table summarizes the available variables—for a full description see the "Templating Language and Processor" chapter in the *Developer's API Reference* and "Logging and Tracing Facilities" on page 569:

Variable	Description
DM_TEMPLATE_CATALOG<n>	Specifies a data set or directory search path where templates are located.
DM_TPLB_LOG	Turns-on tracing of template processing.
DM_TPLB_SPOOL	Specifies the name of the file or data set where the JCL is written.
DM_TPLB_TTLOG	Specifies that any trace sent to the file that you specify in DM_TPLB_LOG is also sent to the message routine.
DM_TPLB_SEQUENCE_PATH	Specifies the directory where the HFS file that maintains the build sequence number is stored.

Restricting Visibility of Objects in CM Clients

You can add the configuration symbol `DM_PRODUCT_USER` to `dm.cfg` to restrict the visibility of Dimensions CM objects in the GUI clients, for example, the desktop and web clients, the Administration Console, and the Eclipse plugin. This symbol is disabled by default.

To set the `DM_PRODUCT_USER` symbol:

1. Stop the Dimensions CM server.
2. Add the following line to the `dm.cfg` file on the Dimensions CM server:

```
DM_PRODUCT_USER y
```

3. Restart the Dimensions CM server.

When this symbol is set to `'y'`, the user interfaces list only objects owned by products on which the current user holds a role.

For example, imagine that you have two Dimensions CM products called `PAYROLL` and `DEMO`. A user has the role `REVIEWER` on some design part on product `PAYROLL`, but no

role assignments at all on product DEMO. When using the Dimensions CM web tools, that user sees only the following:

- Projects from product PAYROLL.
- Requests belonging to product PAYROLL.
- Baselines of product PAYROLL
- Any other objects are limited to product PAYROL

IMPORTANT! The one exception to the above is that $GENERIC:$GLOBAL is always listed (regardless of role assignment), because all users must be able to get into the $GLOBAL project, However, when looking at objects in the $GLOBAL project, a user sees only those objects owned by products on which there are roles, for example, PAYROLL in the above example.

Running Multiple CM Instances on a UNIX Host Machine

CAUTION! To connect to several Dimensions servers on a single UNIX host, perform both the server-installation listener.dat edits and client-installation dm.cfg edits described next.

By default, a Dimensions for UNIX server uses the following hard-coded UNIX domain socket (two-way communications pipe) for communication with the associated Dimensions listener:

```
/tmp/dimensions_local_connect
```

To run more than one Dimensions server on a single UNIX host, add the following entry for each associated $DM_ROOT/dfs/listener.dat file on the UNIX host:

```
-local_pipe        <new_socket_name>
```

for example,

```
-local_pipe        /tmp/dimensions_local_connect_custom
```

For details about the Dimensions listener, see "Configuring Dimensions Server Pooling" on page 165.

After you have made the appropriate listener.dat edits, add the following entry to the dm.cfg file on each Dimensions for UNIX or Windows client installation, specifying the socket/listener to which you want to connect:

```
DM_LOCAL_CONNECT   <new_socket_name_to_connect_to>
```

For example:

```
DM_LOCAL_CONNECT   /tmp/dimensions_loccal_connect_custom
```

NOTE You cannot use PDIFF on Windows on a non-default port.

Smart Attribute Validation

The DM_SMART_ATTR_VALIDATION configuration symbol specifies how the auto-completion of multi-value attribute blocks behaves.

Suppose a valid set has been defined with three columns, and a block attribute has been created for Dimensions requests of type CR in which Column 1, Column 2, and Column 3 represent Name, Surname and Area for developers. In such a scenario, the value of the DM_SMART_ATTR_VALIDATION configuration symbol modifies auto-completion behavior as follows:

- When DM_SMART_ATTR_VALIDATION is *turned off*, the restriction on values displayed in the drop-down lists applies both from left to right and from right to left.

  From left to right, for example, if the user selects Peter for the Name, the list for Surname includes only the surnames of the developers named Peter. The same behavior applies to the Area field.

  From right to left, if the user selects DC for Area, it restricts the Name and the Surname fields according to the Valid Set. This means that the user cannot select another developer unless the Area field is cleared.

- When DM_SMART_ATTR_VALIDATION is *turned on*, this restriction applies only from left to right. Selections made on fields on the left restrict the lists of fields on the right, but not the other way around.

  For example, if the user selects Peter for Name, both the Surname and the Area fields are restricted, just as in the previous case. However, if the user selects DC as Area, the list for Name and Surname displays all developers. No restriction is made on fields on the left. This enables the user to select a different developer without having to clear the Area field first.

For details about the Administration Console functions, see the *online help*.

Using PAM on Dimensions Servers

Preliminary support, by means of the DM_AUTH_TYPE_DBS and DM_PAM_SERVICE_NAME configuration symbols, is available for the industry standard pluggable authentication modules (PAM) mechanism—refer to "Dimensions Support for PAM" on page 459 for details.

Remote Execution on a Tertiary Node Running HP-UX 11.23 on Itanium

If you are using REXEC to connect to a tertiary node running HP-UX 11.23 on Itanium hardware, the operation fails. This is a known issue with the operating system. To work around the problem, add the following variable to the dm.cfg file on the Dimensions server:

```
DM_TPLB_AVOID_HP_BUG Y
```

Specifying the Server Domain Name

Including the parameter below forces the Dimensions CM server to specify its fully qualified domain name in certificates passed to remote nodes.

DM_CERT_FULL_HOSTNAME

File Compression Settings

The `dmcli` commands `DDF` and `SDF` have an optional `/COMPRESSION_LEVEL=<level>` qualifier. This qualifier enables you to specify whether files of the associated format should be considered for compression during transfer (thus reducing bandwidth) and what level of compression to apply. The levels are:

0	No compression. If no level is specified, this is the default for binary file formats.
1	Fastest speed of compression, but least compression obtained. If no level is specified, this is the default for text file formats.
2–8	For each increase in value, a corresponding decrease in speed of compression with an increase in compression obtained.
9	Slowest speed of compression, but with best compression obtained.

To enable compression, set the following symbols in the `dm.cfg` file on the Dimensions CM server node:

```
DM_COMPRESS_FILES_ON_TRANSFER yes
DM_COMPRESSION_MIN_FILELENGTH 63990
```

The first symbol enables compression. The second symbol specifies the default compression threshold in bytes; files smaller than this value are not considered for compression.

Item Header Substitution Last-Record Behavior

To be compatible with the behavior of item header substitution in previous releases, Dimensions CM does not, by default, add an end-of-line character sequence (such as CR-LF) to the last record of a text file if it does not already have one.

If you want to modify this default behavior, add the following variable to the `dm.cfg` file on the Dimensions CM server:

```
DM_IHS_FIX_LAST_RECORD YES
```

Disabling Branch Check-Ins

Dimensions CM can be configured to determine whether an item has any leaf revisions that need to be merged. The desktop client can then prompt the user with a number of options when they are checking in or updating an item and Dimensions CM has detected a conflict. The user has a choice of actions to take, including to merge the work file with the version in the repository (invoking the file merge tool) and to check in the file as the tip revision. (For details, see the Dimensions CM online help.)

If you want Dimensions CM to check for conflicts with leaf revisions and force all users in the desktop client to merge their changes with the tip revision in the repository, set the following symbol in the dm.cfg file:

```
DM_DISABLE_BRANCH_CHECKINS YES
```

Note that, if you do not set this symbol on the server, you can also choose to set this option in the desktop client Preferences dialog box.

Enabling Global Project Structure Change Control

Project structure change control is a Dimensions CM feature that enables you to track changes to Dimensions CM projects over time. This enables you to store information about item additions and removals in change requests, and to require change requests for any change in a project structure. You can then include this data about project structure changes in any baseline that you create.

Recorded Project Structure Changes

When project structure change control is enabled, the following changes to project structure are recorded against change requests:

Change	Description
Item addition and removal	An item revision is added or removed from a project. The change is recorded in one of the following: ■ The change request that lead to the structure change ■ The default change request
Item rename	An item is renamed.
Directory creation, deletion, and rename	A directory is created, deleted, or renamed.
Item deletion	An item is deleted. When you delete an item, however, no history is stored and all structure change requests associated with the item are deleted.
Project deletion	A project is deleted.

The variable DM_PATH_CONTROL determines whether project structure change control is in effect for all Dimensions CM projects. It determines the default setting for all new projects; structure change control is either enabled or disabled by default for all new projects. However, you can override this default on a per-project basis using the DWS and UWA commands. For details, see the *Dimensions CM Command Line Reference*.

You can set project structure change control globally by editing or inserting the DM_PATH_CONTROL variable in dm.cfg on the Dimensions server machine.

Set this variable to one of the following values:

- DM_PATH_CONTROL 0

  To disable structure change control for all existing and new Dimensions projects.

- DM_PATH_CONTROL 1

To enable structure change control for all existing and new Dimensions projects.

IMPORTANT! To prevent affecting projects that are in progress, consider setting this variable to `0`.

Privilege Caching

To improve performance, Dimensions CM application servers cache certain privilege and role information. In general this cached information is refreshed only when an application server is allocated to a session, that is, on connection and when an idle session is reactivated.

The cache for the application server allocated to a session is also refreshed if the session executes a command that affects privileges or roles (for example, `PRIV` or `AUR`).

This means that changes made to the role and privilege settings in the Administration Console do not take effect immediately for other sessions.

This default behavior can be modified through editing or inserting the following variables in `dm.cfg` on the Dimensions CM server machine:

- `DM_DISABLE_PRIVILEGE_CACHE`

  To disable all privilege caching, set this to any non-empty string.

- `DM_PRIVILEGE_PURGE_INTERVAL`

  This can be set to value equal to the desired number of seconds between privilege cache refreshes. The value set must be less than the `listener.dat` idle timeout value (see "-idle_timeout" on page 168) to have any effect.

IMPORTANT! Using either of these variables decreases performance.

As an alternative to using the above `dm.cfg` symbols, to force refreshment of privilege and role information you can simply restart the Dimensions CM listener to ensure that a new generation of application servers is created

Cleaning Up Temporary Build Files

Various temporary files are left on a build node after a Dimensions Build. Some of these files can be deleted by the node itself, while others can be deleted by the build server once it knows that it has finished using the file in question. Proceed as follows:

- **To control file deletions issued by the build node itself:**

NOTE The default position is for the build node *not to delete* temporary files.

To enable deletion of temporary files by the build node, add the `DM_BUILD_CLEAN_TEMP_FILES` symbol to the *build node's* `dm.cfg` configuration file and set the symbol to any value. For example:

```
DM_BUILD_CLEAN_TEMP_FILES YES
```

- **To control file deletions issued by build server with respect to the build node:**

NOTE The default position is for the build server *to delete* temporary files on the build node.

To prevent the build server deleting temporary files on the build node, add the `DM_BUILD_CLEAN_TEMP_FILES` symbol to the *Dimensions server's* `dm.cfg` configuration file and set the symbol to value `FALSE`. That is:

```
DM_BUILD_CLEAN_TEMP_FILES FALSE
```

Build Error Handling

When a build is launched, the user now has a choice as to how to handle error conditions related to source files in a build when the build occurs in a deployment area.

You can make all errors appear:

as fatal errors which fail the build

or

as warnings which allow the build to proceed.

Firstly, the following symbol can be used to control the overall behaviour:

```
DM_BLD_ERROR_INVALID_REVISIONS true
```

This causes all errors relating to build sources to be fatal to the build. Only if the exact requested revision is present in the build area, the build succeeds. The error messages provided in the build launch history document the filenames found in error.

```
DM_BLD_ERROR_INVALID_REVISIONS false
```

This causes all errors relating to the presence of source files in the area to be downgraded to warnings, and allow the build to proceed.

In addition, advanced users can configure exactly which errors are fatal and which are not. This is done by calculating a hexadecimal number from a set of individual flags. The number required can be formed by adding together the following values, for the errors which are required to be warnings:

```
NOFAIL_MISSING 0x00100000 // don't fail if source file is missing from
    search path
NOFAIL_NEWER 0x00200000 // don't fail if source file revision found is
    higher than requested
NOFAIL_OLDER 0x00400000 // don't fail if source file revision found is
    lower then requested
NOFAIL_NOITEM 0x00800000 // don't fail if source file is missing from
    the project/stream
NOFAIL_SPEC 0x01000000 // don't fail if the spec is wrong
```

For example, if the only error that you want to make a warning is the case where the revision found on disk is NEWER than the item requested, the the value required is 00200000. Compound selections are made by adding together the individual options. This value is coded in the dm.cfg file in the key DM_BLD_GETSRC_FUNCTION_OR. Here is an example, which turns the NEWER and OLDER errors into warnings:

DM_BLD_GETSRC_FUNCTION_OR 00600000

DM_BLD_ERROR_INVALID_REVISIONS

For details of this symbol, see the *Dimensions Build online help*.

DM_BUILD_DMORDERING

For details of this symbol, see the *Dimensions Build online help*.

DM_BLD_GETSRC_FUNCTION_OR

For details of this symbol, see the *Dimensions Build online help*.

Supporting Refactoring Changes When Creating Revised Baselines

Starting with Dimensions 10.1.3, the Create Revised Baseline (CRB) command has been enhanced to support refactoring changes by default. This means that when requests that have been used to track refactoring changes are included in the list of update requests, those refactoring changes are applied to the baseline.

Ordering of Attributes

In versions of Dimensions up to, and including 10.1.3, the attribute rules are ordered by role; and rules that are non-mandatory take precedence over mandatory rules. This means that an attribute rule with no `From` and `To` states can take precedence over one that has specified `From` and `To` states.

For example, consider the following attribute rules:

Attribute	From	To	Role	Rules
PLAN_FINISH	$TO_BE_DEFINED	REVIEW	$ORIGINATOR	Required, Writeable
PLAN_FINISH			AUTHOR	Writeable

In the above example, if the user has the `AUTHOR` role, then it takes precedence over the other one that is a more specific mandatory attribute rule.

Starting with Dimensions CM 2009 R1, by default, attribute rules are ordered by:

1. `From` state, then,
2. `To` state, then,
3. `Rules` (not Required ahead of Required), and finally,
4. `Role`.

In the example below the attributes rules are ordered as follows:

Attribute	From	To	Role	Rules
PLAN_FINISH	REVIEW	WORK	CRB	Required, Writeable
PLAN_FINISH	REVIEW	WORK	IMPLEMENTOR	Writeable
PLAN_FINISH			TESTER	

Visibility of Requests on Design Part Structure

In versions of Dimensions prior to 10.1.3, you can navigate through all requests using the Design Part structure. Starting with Dimensions CM 10.1.3, by default, this behavior was changed so that the parts display only showed requests related to the current Dimensions CM project.

To revert to the earlier behavior, add the following symbol to dm.cfg on the Dimensions CM server:

```
DM_SHOW_PART_ALL_REQUEST Y
```

Item Library Security

Starting with Dimensions CM 10.1.3.3, there is an option that enables you to *prevent* users from performing item library operations through a different server from the one for which the item libraries were initially defined. This option is used to control access from any other node that starts a library server process for item operations.

If you include the following symbol in `dm.cfg`:

```
DM_ALLOWED_HOSTS=<localhost_id>,<host_id1>,...,<host_idN>
```

then only connections from the comma-separated list of specified hosts is allowed.

The value of `DM_ALLOWED_HOSTS` must include the host name of the local host (or its IP address) and a comma-separated list of additional allowed hosts, but do not use the value `localhost` to specify the local host, as it is not supported. An example of a `DM_ALLOWED_HOSTS` specification for a local host and a second host is as follows:

```
DM_ALLOWED_HOSTS=10.130.12.66,xpserver
```

NOTE

You can secure a Dimensions CM server/agent hosting item libraries by disabling the ability to start `dmlibsrv` processes in privileged mode. Add the following variable to `listener.dat` on the server/agent:

```
-protected_mode
```

When protected mode is enabled, all item libraries hosted on the server/agent must have an associated credential set, for details see:

- "CCS – Create Credential Set" on page 181.
- The DPL command in the *Command-Line Reference*.

Disabling the Automatic Selection of Build Targets

By default, targets are automatically selected by Dimensions Build. To disable this feature, uncomment and edit the following variable in `dm.cfg`:

```
NO_BUILD_TARGET_PRESELECT PRODUCT1:PROJECT1, PRODUCT2:PROJECT2 …,
```

Projects/streams that you specify must be separated by a comma or a space.

If a project/stream specification itself contains commas or spaces, it has to be specified in double quotes, for example:

```
DM_NO_BUILD_TARGET_PRESELECT QLARIUS:PROJ1,QLARIUS:PROJ2
    QLARIUS:PROJ3 "QLARIUS:MY NEW PROJECT", "QLARIUS:PROJ A, WITH
    CHANGES"
```

Contents of Temporary Build Project

When the build of a stream is complete, the temporary project that was used contains all the build targets and their dependent sources. Using the "made-of" relationships as a guide, only the build targets used during the delivery are added to the XML file because the source files should already exist in the stream. If these relationships are incorrect or not complete, set the following variable in dm.cfg to TRUE to force all items from the project (sources and targets) to be added to the delivery XML:

```
DM_BLD_MERGE_ALL_ITEMS
```

When a delivery occurs using an input file that contains file revisions already in the stream, the delivery ignores these items but adds a warning message in the delivery log file. Only those listed files that do not exist in the stream are merged (sources or targets).

Pre-version 8 Item Header Substitution Functionality

In Dimensions CM 8, the item header markers `%PCMS_HEADER_SUBSTITUTION_START%` and `%PCMS_HEADER_SUBSTITUTION_STOP%` were introduced, these optional markers delineating the item header substitution variables. This enabled performance to be enhanced, as, if present, Dimensions only had to scan the file contents between those two markers for item header substitution variables. However, if the markers were not present ("old style" item header substitution), Dimensions continued by default to perform item header substitution, albeit with a performance overhead.

Starting with Dimensions CM 2009 R1, processing of "old style" item substitution is no longer supported by default, again to improve performance. Any "old style" item header substitution variables are ignored.

If you have such "old style" item header substitutions that you want to continue to use, you can make Dimensions CM process them by adding the following setting to the server `dm.cfg` file:

```
DM_IGNORE_IHS_START y
```

Display Related Items for Child Requests in Request Browse Templates

The request template variable %affected_item% displays details of items that are related to the specified request. If you want the template to display details of items also related to child requests, you can specify the following setting:

```
DM_BROWSE_RELATED_DOCS_ITEMS y
```

Processing Items or Requests not Owned Locally

Dimensions CM 2009 R1 or later includes new optional functionality (off by default), first made available as an earlier customer-specific patch, namely, the ability to process items or requests that you do not locally own.

Prior to this new functionality, you were not allowed to process such items or requests. For example, if you replicated an item and a request to a replica, the replica cannot relate these objects, and any attempt to do so results in the error message:

```
Error: Operation is not allowed because item revision "x" has been
    replicated and is owned by another site.
```

To enable the processing of items or requests that you do not locally own, add the following settings to the server `dm.cfg` file:

```
DM_DONT_ENFORCE_ITEM_OWNERSHIP  Y
DM_DONT_ENFORCE_REQ_OWNERSHIP   Y
```

CAUTION! These server settings enable concurrent updates of objects between different sites. This may cause data loss.

Once the item ownership flag is set, the following operations are permitted on an item that is not locally owned:

AI	Action item.
UIA	Update item attributes.
RII/XII	Relate/unrelate item to item.
RIP/XIP	Relate/unrelate item to design part.

Once the request ownership flag is set, the following operations are permitted on a request that is not locally owned:

RICD/XICD	Relate/unrelated item to change request.

Changes to Administration Console Behavior

By default, when configuring a project replication remote replica, the Administration Console prevents the selection of any version branch names that are not owned by the replica being configured. To enable the Administration Console to bypass this ownership check when the `DM_DONT_ENFORCE_ITEM_OWNERSHIP` and `DM_DONT_ENFORCE_REQ_OWNERSHIP` symbols are set, add the following setting to the server `dm.cfg` file:

```
DM_REPL_NO_PROJECT_BRANCH_CHECK    1
```

The remote replica site can then be suitably configured to allow item and request updates to be replicated back to the site which owns them.

In addition to a suitable configured configuration, you also need to use the following `replicator` option:

```
-replicate_remote_branches
```

to replicate branches not owned by the sending site. This option should be used when dumping items that are on a version branch that is not owned by the "sending" site.

Checking Mandatory Multi-Valued Attributes in Block Attributes

Prior to Dimensions CM 2009 R2, if a block attribute contains multi-value (MVA) attributes whose assigned values are defined as mandatory, for each such MVA attribute, only one populated row had to have an assigned mandatory value—any additional populated rows did not require to be assigned values.

This behavior is, now, no longer the default; instead, each populated row in a MVA attribute within a block attribute must contain assigned mandatory values.

To revert back to the earlier behavior, add the following symbol to the Dimensions CM server `dm.cfg` file and set it to any value, for example:

```
DM_MANDATORY_MVA_ANY_ROW          Y
```

NOTE This new behavior has also been back ported to Dimensions CM 10.1.3 patches (but not to Dimensions CM 2009 R1.x).

Event Tracing

To enable event tracing, set the following symbol to the Dimensions CM server `dm.cfg` file to a non-null value, for example:

```
DM_EVENT_TRACE          Y
```

For details see, Appendix J, "Logging and Tracing Facilities".

Disabling Editing of Block Attributes on Multiple Requests

Prior to Dimensions CM 2009 R2, and by design, editing block attributes on multiple requests from the desktop client overwrites existing data, which can cause data loss (see the example scenario below). If you want to disable such editing from the desktop client, you can now add the following symbol to the Dimensions CM server `dm.cfg` file and set it to any value, for example:

```
DM_DISABLE_MULTIPLE_EDIT_ATTRS          Y
```

NOTE This new behavior has also been back ported to Dimensions CM 10.1.3 patches (but not to Dimensions CM 2009 R1.x).

Example scenario of data loss with DM_DISABLE_MUTIPLE_EDIT_ATTRS unset

- Steps to reproduce:
  - **a** Add two or more single-field, multi-valued attributes to a request type.
  - **b** Create a new multi-field (block) attribute using the attributes created above
  - **c** In desktop client, create two new requests of the above request type and populate the block attribute with two or more lines.
  - **d** Select both requests in desktop client and Edit Attributes from the context menu.
  - **e** On the Attributes Tab, note that the block attribute is not displaying any of the previously entered lines.
  - **f** Change one or more values in the block and click **OK**.
- Commonly expected result (and that which is now obtained if `DM_DISABLE_MUTIPLE_EDIT_ATTRS` is set):

  The information chosen (or entered) in the edit attributes operation is added as a new line to the block in both requests.
- Actual default result (with `DM_DISABLE_MUTIPLE_EDIT_ATTRS` unset):
  - The existing value of the first row in the block for both requests contains the newly entered value.
  - All other rows are empty for this field.

Pre-10.1.3 Behavior For Deleting Non-Tip Item Revisions

Prior to Dimensions CM 10.1.3 it was possible to delete multiple item revisions from the Dimensions repository. Starting in Dimensions CM 10.1.3 and later, however, the default behavior has been changed such that only the tip item revision is deleted (even if several item revisions were selected for deletion). The reason for this change of behavior is that missing non-tip revisions lead to problems when using the Merge Tool to merge item revisions.

Starting in Dimensions CM 10.1.3.2, if you want to revert to being able to delete non-tip item revisions, you can now add the following symbol to the Dimensions CM server `dm.cfg` file and set it to any value, for example:

```
DM_DELETE_NON_TIP          Y
```

The Dimensions listener needs restarting for the change to become effective.

Pre-2009 R1 Item Library Access Mechanism

Starting in Dimensions CM 2009 R1, each Dimensions `dmappsrv` process attempts to directly access local item libraries for read-only operations (as opposed to staring a privileged `dmlibsrv` process). This revised access mechanism provides both performance enhancement and increased security (it precludes the necessity of having to run many `root` mode `dmlibsrv` processes).

If you want to revert to the earlier less secure access mechanism (for example to enable you to put item libraries in a location that is not readable by the Dimensions System Administrator (by default user dmsys), comment out the following symbol in the Dimensions CM server `dm.cfg` file:

```
# DM_DIRECT_LOCAL_LIBRARY_ACCESS          Y
```

The Dimensions listener needs restarting for the change to become effective.

Visual Studio Clients: Prompting for Check Out

IMPORTANT! The check out prompting described below is only available for Dimensions CM projects (which support pessimistic locking); it is not supported for Dimensions CM streams (which support optimistic locking).

In Dimensions CM 12.x and later, if you have Dimensions CM for Windows clients installed on a remote Windows node and those clients utilize the Dimensions CM Visual Studio clients integration to communicate with the Dimensions CM for UNIX server, you can specify that you want to receive a prompt to check out a file every time you modify it. To do this, add the following symbol to the Dimensions CM server `dm.cfg` file and set it to any value, for example:

```
DM_VISUALSTUDIO_PESSIMISTIC          Y
```

The Dimensions CM listener needs restarting for the change to become effective.

Controlling Deployment Regression

In Dimensions CM 12.1 a dm.cfg flag `DM_NO_DEPLOYMENT_REGRESSION` was introduced to control deployment regression.

- When this flag is set to `TRUE` or `1`, each deployed item is only written to disk if it is newer than the existing item revision in the area (that is, no regressions). In this situation, each item that is not deployed is logged in the deployment history log (so that the user knows why the requested item was not deployed). This type of issue does not stop the overall deployment from completing—Dimensions CM treats it as a warning and continues processing any additional files in the overall deployment job.

  When upgrading from versions of Dimensions CM that used the pre-12.1 deployment behavior, the flag is normally set to `YES` to mirror that behavior, for example:

  ```
  DM_NO_DEPLOYMENT_REGRESSION          Y
  ```

- When this flag is set to `FALSE` (or is not defined), each deployed item is always written to disk—regardless of an existing revision.

  For fresh installations of Dimensions CM, this flag is normally set to `NO` as this is the recommended behavior, for example:

  ```
  DM_NO_DEPLOYMENT_REGRESSION        N
  ```

The Dimensions CM listener needs restarting for changes to become effective.

Automatically Spreading Item Library Files across Multiple Subfolders

When you deliver many thousands of files to Dimensions CM for the first time, the item library typically puts all files with the same item type into the same library folder. To avoid file system performance issues, add the following flag to dm.cfg on the server:

```
DM_RANDOMIZE_LIBRARY_PATHS y
```

The server automatically spreads new files mapped to the same item type across X/Y/ sub folders of the corresponding item library folder, where X and Y are the first two hex digits from a cryptographically secure checksum hash of the corresponding item library path name.

Retaining UNIX File Permissions

To retain UNIX file permissions, add the following flags to dm.cfg on the server or agent:

- `DM_GET_KEEP_PERMS`

  Keeps file permissions in a remote UNIX work area when you perform a Get Copy operation from a project or stream.

- `DM_DEPLOY_KEEP_PERMS`

  Keeps file permissions in a remote UNIX deployment area when you deploy items, requests, and baselines in a project or stream.

- `DM_RELEASE_KEEP_PERMS`

  Keeps file permissions in a remote UNIX release area when you create a release from a baseline.

Checking a Client's Version

To make sure that all Dimensions CM clients are the same version as the server, add the following variable:

`DM_REJECT_OLDER_CLIENTS`

If a client version is older than the server version, the client is not connected and the following message is displayed:

```
The version of your Dimensions CM client is older than the version of
the server. Please upgrade your client.
```

If you do not add `DM_REJECT_OLDER_CLIENTS`, all clients are connected.

Checking Baseline Consistency

Baseline consistency checks enforce additional verifications on the requests that are used to create a revised baseline (CRB) or a request template baseline (CBL). These checks ensure that:

- The requests used to create a new baseline provide a complete set of changes.
- Any dependent changes are not missing.

- The changes have been fully implemented.

To enable baseline consistency checks add the following variable:

```
DM_ENABLE_BASELINECHECKS Y
```

For more details see the online help.

Preserving REXEC Command Log Files

The REXEC (Execute a Job on a Network Node) command includes an optional qualifier, /PRESERVE, that enables users to save job logs as items in Dimensions CM. To specify where logs are preserved add the following variables to the Dimensions CM configuration file on a network node:

- DM_REXEC_LOGS_RULE

  Defines the rules for constructing the workset filename. The % values are replaced as follows:

Value	Replaced with
%j	Job number
%f	■ Return code equals 0: WORKED ■ Return code does not equal 0: FAILED
%r	Process return code
%n	Network node name
%u	Dimensions CM user ID
%t	Template name

  Important

  - You must enter two percent symbols for each value: %%. These are resolved by the dm.cfg logic to one percent symbol. For example, %%j becomes %j.
  - Use '/' for folder separators not '\'.
  - Do not start a DM_REXEC_LOGS_RULE with '/' (this is a relative path).
  - To ensure that filenames are unique, use the value %j (revisions of logs for different jobs is not supported).

- DM_REXEC_LOGS_PROJECT

  Specifies where REXEC job logs are preserved. Can be one of these values:

  - *CURGENERIC

    Preserves logs in $GENERIC:$GLOBAL for the current product.

  - *CURRENT

    Preserves logs in the current project or stream.

  - <product>:

    Preserves logs in $GENERIC:$GLOBAL using the item library and upload rules for the product that you specify, for example:

```
DM_REXEC_LOGS_PROJECT MYPRODUCT:
```

- `<product>:<project>`

  Preserves logs in the product and project that you specify, for example:

  ```
  DM_REXEC_LOGS_PROJECT QLARIUS:PAYROLL
  ```

- `<stream>`

  Preserves logs in the stream that you specify (product ID is not required), for example:

  ```
  DM_REXEC_LOGS_PROJECT MYSTREAM
  ```

- `DM_REXEC_LOGS_FORMAT <value>`

  Overrides any format computed by upload rules to the value that you specify.

- `DM_REXEC_TRACE Y/N`

  Switches REXEC SDP tracing on and off.

Setting the DBIO Library

To set the DBIO library used to connect to the database, add the following symbol:

`DBIO_LIBRARY`

Specify one of the following variables:

- Oracle 11 Unicode UTF-8 AL32UTF8: `libdbio_srv_oci8_11201u.so`
- Oracle 12 Unicode UTF-8 AL32UTF8: `libdbio_srv_oci8_12101u.so`
- PostgreSQL: `libdbio_srv_postgres.so`

Specifying the Database Vendor

To set the DBIO library used to connect to the database, add the following symbol:

`DM_DEFAULT_DBMS`

Specify one of the following variables:

- `ORACLE`
- `PSQL`

Dimensions for z/OS Configuration Symbols

Please see the "Customizing Variables in the Dimensions Configuration File" section in the "Installing Dimensions for z/OS" chapter of the *Dimensions for z/OS Guide*.

Defaults for Copying Build Information

There are some commands that are able to copy build information from one project or stream to another. This information relates to build configurations and relationships. By specifying the relevant qualifiers, certain build information can be copied to the target project or area involved. These commands are:

- AIWS - Add Item to Project
- CBP - Copy Build Project
- CS - Create Stream
- DWS - Define New Project
- RAWS - Relate Area to Project/Stream.

For details, see the relevant sections in the *Command-Line Reference*.

By default, these commands do not copy any build information, but the default behavior can be changed by specifying the following variables in the `dm.cfg` file:

- To change the defaults for copying build relationships, add the following symbol:

  ```
  DM_BUILD_COPY_RELATIONSHIPS    1
  ```

  Defining this symbol changes the default value `/NOCOPY_RELATIONSHIPS` in the `AIWS`, `CBP`, `CS`, `DWS`, and `RAWS` commands to `/COPY_RELATIONSHIPS`. This applies to the copying of build relationship data, such as made-of records held in the database.

  To unset this symbol, change the entry to:

  ```
  DM_BUILD_COPY_RELATIONSHIPS    0
  ```

- To change the defaults for copying build configurations, add the following symbol:

  ```
  DM_BUILD_COPY_CONFIGS    1
  ```

  Defining this symbol changes the default value /NOCOPY_CONFIGS in the `CBP`, `CS`, and `DWS` commands to `/COPY_CONFIGS`. This applies specifically to the build configurations held in the database.

  To unset this symbol, change the entry to:

  ```
  DM_BUILD_COPY_CONFIGS    0
  ```

- To change the default error processing when copying build information, add the following symbol:

  ```
  DM_BUILD_COPY_FORCE    1
  ```

  Defining this symbol changes the default value `/NOFORCE` in the `CBP`, `CS`, and `DWS` commands to `/FORCE`. This causes the command to perform as much processing as possible in certain contexts, instead of failing at the first error.

  To unset this symbol, change the entry to:

  ```
  DM_BUILD_COPY_FORCE    0
  ```

NOTE These settings accordingly affect the behavior of the related functions in the Dimensions CM web client and desktop client that invoke these commands when:

- Creating a project or stream from an existing one
- Assigning an area to a project or stream
- Adding an item to a project.

SSO Configuration Symbols

If you chose to install the Dimensions CM Single Sign-On (SSO) server, the following configuration symbols are automatically added by the installer to the Dimensions CM server `dm.cfg` file and assigned values (some of which depend on your answers to the installer prompts):

- DM_AUTH_TYPE_DBS
- SSO_STS_ENDPOINT
- SSO_TRUST_CERTIFICATE
- SSO_RELYING_PARTY
- SSO_TOKEN_LIFE_TIME
- SSO_CLOCK_TOLERANCE
- SSO_DONT_USE_BEFORE_EXPIRATION_TIME
- SSO_ENDORSING_USER
- SSO_ENDORSING_USER_PASSWORD

For details about these symbols, see Appendix A, "Configuration Entries for SSO Authentication" on page 470.

IMPORTANT! The Dimensions CM 14.1 installer assigns a value of `28800` seconds (eight hours) to `SSO_TOKEN_LIFETIME` in `dm.cfg`. The value is consistent with the parameter value assignment in the XML file detailed below (and if this value is altered, it should be consistently altered in both dm.cfg and the XML file):

- Windows

  ```
  <Tomcat-Root>\alfssogatekeeper\conf\gatekeeper-core-config.xml
  ```

- UNIX

  ```
  <Tomcat-Root>/alfssogatekeeper/conf/gatekeeper-core-config.xml
  ```

parameter-value assignment:

```
<parameter name="SecurityTokenTTL" Type="xsd:decimal">28800
    </parameter>
```

CAC Configuration Symbols

If you chose to configure Common Access Card (CAC) smart card setup at the same time as installing the Dimensions CM SSO server (CAC is dependent on SSO), the following SSO configuration symbols—additional to those for SSO—are also automatically added by the installer to the Dimensions CM server `dm.cfg` file and assigned values (some of which depend on your answers to the installer prompts):

- SSO_SERVER_CERTIFICATE
- SSO_SERVER_PRIVATE_KEY
- SSO_SERVER_PRIVATE_KEY_PASSWORD

IMPORTANT! If you install only the SSO component to work in conjunction with SSO and CAC located on a Solutions Business Manager (SBM) installation, then the above entries need to be manually added to the Dimensions CM server `dm.cfg` file.

For details about these symbols, see Appendix A, "Configuration Entries for SSO Authentication" on page 470.

Chapter 3

Setting Dimensions Server Windows Symbols

Introduction	74
Modifying or Adding dm.cfg Entries	75
Installation and File Location Symbols	76
Using a Proxy Server	76
Sending Email to Dimensions Users	78
Printing Data	82
Clearing Up Temporary Files Created by User Reports	82
Reporting and Optional Symbols	83
Windows Server Log On Domain	83
Networking and Replication	84
Code Page Settings	84
Oracle Collating Sequence for 'ORDER BY' Queries	84
Dimensions Web Tools URL Parameters	84
Additional Dimensions Configuration Symbols	85
Overriding Host Name Assigned to DMSERVER Certificates	90
Dimensions for z/OS Configuration Symbols	90
Defaults for Copying Build Information	90
SSO Configuration Symbols	90
CAC Configuration Symbols	90

Introduction

The Dimensions dm.cfg File

Each Dimensions CM server installation contains a `dm.cfg` configuration file located in the directory `%DM_ROOT%`. This file specifies Dimensions CM Windows configuration symbols for that particular installation (a smaller similar named file is created for client installations on client nodes).

If the installation is part of a larger network of Dimensions CM installations, make sure that configuration settings that affect communication with other installations are the same.

The directory `%DM_ROOT%` is specified by the registry key:

```
HKEY_LOCAL_MACHINE | SOFTWARE | Serena | Dimensions | <version> |
    DM_ROOT
```

This is set during the installation and is the default Dimensions CM configuration for all users of the machine.

NOTE The value of the `DM_ROOT` variable must always end with `\` (backslash).

The `dm.cfg` settings are installation-wide. Parameters specific to you must be set in your own particular environment. If you modify settings both in the `dm.cfg` configuration file and in your own particular environment, your environment takes precedence.

These configuration parameters and environment variables are collectively referred to as the **Dimensions CM symbols** to reflect that analogous assignment mechanisms are used for the Dimensions CM UNIX product (utilizing environment variables with the same legend). Most Dimensions CM symbols are supported on all the operating systems.

This section describes only those symbols that are recommended for modification.

Dimensions CM symbols and scripts principally set working environments or invoke tools in following areas:

- Installation and location symbols
- Using a proxy server
- Sending email to Dimensions CM users
- Printing data files
- Clearing up temporary files
- Reporting and optional symbols
- Logon domain
- Networking and replication
- Code page settings
- Command security
- Oracle collating sequence
- Dimensions CM web tools URL parameters

- Additional Dimensions CM configuration symbols.

By default, Dimensions CM makes use of standard system tools. But you can specify to use third-party tools instead, for example, installed proprietary packages for word processing or publishing.

Additional Dimensions Configuration Symbols

In addition to the standard Dimensions CM Windows configuration symbols, Dimensions CM provides specialized configuration symbols that are either specified in `dm.cfg` or can be added. For details, see "Additional Dimensions Configuration Symbols" on page 85.

Modifying or Adding dm.cfg Entries

CAUTION! When modifying or adding `dm.cfg` symbols, make sure that the entries do not contain trailing whitespace characters, as this may result in an error. For example, an entry of "`DM_PROG<tab>%DM_ROOT%prog\<space>`" results in failure to connect to a Dimensions CM application.

Most installation-wide entries in `dm.cfg` should not be changed by end users. Remember that these installation-wide values can always be superseded in each user's environment by setting the appropriate environment variable.

To open the `dm.cfg` settings for editing, go to:

`Dimensions <version> | Configuration Parameters`

IMPORTANT! Typically, for changes in `dm.cfg` to take effect, you need to restart the Dimensions listener after updating `dm.cfg`.

You can change the following entries in `dm.cfg`, and a few outside `dm.cfg`, if needed. For more details about each parameter, see the relevant page.

dm.cfg symbol	page
DM_TMP	76
DM_REMOTE_FILE_AREA	76
DM_MAILS	78
DM_DMEMAIL	82
DM_PRINT	82
DM_DELETE	82
DM_CODEPAGE	84
DM_EBCDIC_CODEPAGES	84
DM_DEFAULT_CODEPAGE	84
DM_NLS_SORT_PARAM	84

dm.cfg symbol	page
DM_WEB_URL	84
DM_INET_PORT *(backward compatibility only)*	84
DM_INET_SERVER *(backward compatibility only)*	84

Installation and File Location Symbols

See "Installation and File Location Symbols" on page 42.

Using a Proxy Server

You can set up Dimensions CM to use a proxy server for external connections:

- Tunneled connections via HTTP(S) from the Dimensions CM clients to the server.
- Connections to the integrating tools such as external request providers, Deployment Automation, and PulseUno.
- Connections to the AutoPass License Server (APLS).

If you have set up a proxy in Windows, by default, your system proxy settings are used only for HTTP(S)-tunneled connections.

To configure your Dimensions CM installation for a proxy, add the following symbols to `dm.cfg`:

Symbol	Description
`DM_NET_PROXY_MODE`	Specifies a list of external tools and applications to which to connect via a proxy server.
`DM_NET_PROXY`	Specifies the proxy server through which to route traffic.
`DM_NET_NOPROXY`	Specifies a list of exclusions for which you do not want to use a proxy.

These symbols are described next.

DM_NET_PROXY_MODE

Use the `DM_NET_PROXY_MODE` symbol to specify one or more servers or applications to which to connect through a proxy server.

Enter a list of applications, separating each entry with a space.

The following table lists the possible values:

Value	Description
TUNNEL (or legacy TUNNEL_AUTO)	HTTP(S)-tunneled connections from the Dimensions CM clients to the server. If the DM_NET_PROXY_MODE symbol is not set, by default, all the tunneled connections go through the proxy you set up.
PULSE	Connections to PulseUno.
OCTANE	Connections to ALM Octane for working with external requests.
JIRA	Connections to Jira for working with external requests.
SDA	Connections to Deployment Automation for continuous deployment.
APLS	Connections to the AutoPass License Server (APLS) for managing licenses.
AUTO	All connections listed in this table, except the AutoPass License Server (APLS).

Examples:

To use a proxy for connections to the ALM Octane and Jira request providers, add the following line to the dm.cfg file on the server:

```
DM_NET_PROXY_MODE OCTANE JIRA
```

To use a proxy for all connections, add the following line to the dm.cfg file on the server:

```
DM_NET_PROXY_MODE AUTO APLS
```

To avoid using a proxy for tunneled connections from a client to the server, add the following line to the dm.cfg file on the client:

```
DM_NET_PROXY_MODE N
```

DM_NET_PROXY

Use the DM_NET_PROXY symbol to define the URL of the proxy server through which to route traffic for specified applications. Enter the URL in the following format:

<scheme>://<server>:<port>

where:

- **<scheme>** is the protocol scheme, such as HTTP(S).
- **<server>** is the host name of the proxy server.
- **<port>** (optional) is the port number for the proxy server.

For example:

```
DM_NET_PROXY https://proxyURL:1383
DM_NET_PROXY socks5h://proxyURL:1080
```

If the DM_NET_PROXY symbol is not set, Dimensions CM uses Windows system proxy settings.

DM_NET_NOPROXY

Use the DM_NET_NOPROXY symbol to define hosts that should not use a proxy.

Guidelines:

- For multiple hosts, separate each host with a space, comma (,), or semicolon (;).
- To exclude all local (intranet) hosts, use the value <local>.

Examples:

To exclude all hosts in the **intranet.com** domain, such as **my.intranet.com** and **my.other.intranet.com**, add the following line:

```
DM_NET_NOPROXY intranet.com
```

To exclude all local hosts and all hosts in the **example.com** domain, add the following line:

```
DM_NET_NOPROXY <local>,example.com
```

If you do not specify the URL of the proxy server using the DM_NET_PROXY symbol, Dimensions CM ignores the DM_NET_NOPROXY values and uses exceptions defined in Windows system proxy settings.

Sending Email to Dimensions Users

DM_MAILS

DM_MAILS is the Dimensions CM Windows server symbol used to specify the default interface to the email tool used to communicate information to users. Many Dimensions CM operations need to email users in a multi-user environment. For example:

- When Dimensions CM products are created, the users who are Product Managers are notified through email as to how they can start using their products.
- Whenever an item revision or request is awaiting the attention of specific Dimensions CM users, as a result of a change in its lifecycle state, they are notified through email to read their pending lists.
- Some Dimensions CM commands run batch scripts which on termination email users the outcome of the job.

Control of procedural time-scales and discipline remain with the project management. While Dimensions CM does provide report and query facilities to aid the management of a project, it does not chase outstanding actions.

The default Dimensions CM configuration file dm.cfg contains the following commented out default configuration record for the email client:

```
DM_MAILS  "%DM_PROG%pcmsmail.exe" -dt -h "<mail_host>" -f "<tool manager
    email>" -file "%%s" -s "%%s" -tf "%%s" -cf "%%s" -bf "%%s"
```

To send email in HTML format specify the `-i` switch in `pcmsmail`, the default email client. For details see the next section.

If you entered a Simple Mail Transfer Protocol (SMTP) host name during installation of the Dimensions CM server, the host name is displayed in the above string. If you have an SMTP mailer and did not enter its identity during installation, edit `dm.cfg` to insert it now.

Dimensions CM applications substitute, in the following order, for each of the `"%%s"` parameters:

- The name of a file containing the body of the email.
- The subject of the email.
- A comma-separated email address list.

Any other email client can be substituted provided the above substitutions, in that order, are allowed for using the `"%%s"` notation, and are surrounded with double quotes.

Examples:

```
DM_MAILS<tab>mailxx "%%s" "%%s" "%%s"
DM_MAILS<tab>mymail /file="%%s" /subject="%%s" /to="%%s"
```

The Default Email Client

A Dimensions CM Windows server has an SMTP email client application, `pcmsmail.exe`, that is preconfigured by default. The syntax is:

```
pcmsmail.exe
   -dt
   -h "<host>"
   -f "<from>"
   -file "<file>"
   -s "<subject>"
   -tf "<to>"
   -cf "<cc>"
   -bf "<bcc>"
   –m "<mapping file>"
   -v verbose
   -u "<user>"
   -p "<password>"
   -utf8 <subject>
   -utf8 <address>
   -utf8 <body>
   -i
   -a <file>
   -e <file>
```

where:

`-dt`

Deletes temporary files.

`-h "<host>"`

(Mandatory) `<host>` specifies a network host that is the SMTP email server.

`-f "<from>"`

(Mandatory) Specifies the return email address.

`-file "<file>"`

(Mandatory) Specifies an absolute path to a file that contains the body of the email message.

`-s "<subject>"`

(Mandatory) Specifies the subject of the email.

`-tf "<to>"`

(Mandatory) Specifies a comma-separated list of recipients.

`-cf "<cc>"`

Specifies a comma-separated list of recipients of a copy.

`-bf "<bcc>"`

Specifies a comma-separated list of recipients of a blind copy.

`-m "<mapping file>"`

Specifies a text file that maps Dimensions CM user names to email addresses. Each line maps a user name to an email address, in the following format:

```
<Dimensions CM user name> <email address>
```

Examples:

```
DMCM_admin CM_Administrator@xyz.com
user1 Fred.Jones@xyz.com
```

`-v verbose`

Displays tracing information (used by Support to analyze problems).

`-u "<user name>"`

If Enhanced SMTP is used, this parameter is mandatory to enable user authentication.

`<user name>` specifies a CM user name.

`-p "<password>"`

Specifies the password for the user specified in `-u`.

`-utf8 <subject>`

Encodes the email subject using RFC2047.

`-utf8 <address>`

Encoded the To, From, and CC addresses using RFC2047.

`-utf8 <body>`

Writes the body with a Content-Type of `text/plan` and UTF-8 encoding.

`-i`

A normal email message is comprised of:

- headers generated by `pcmsmail`
- a blank line
- further headers

Use this parameter to add more headers. Note that you must insert the blank line before the body.

To send email in HTML format do the following:

- Use the `-i` switch.
- On the server, set the following symbols in the configuration file, `dm.cfg`:

```
DM_MAILS %DM_PROG%pcmsmail.exe -h "mail_host" -f "PCMS" -file
     "%%s" -s "%%s" -t "%%s"

DM_MAILS_HTTP %DM_MAILS% -i
```

`-a <file>`

Specifies a binary file attachment.

`-e <file>`

Specifies a binary file attachment. Dimensions CM handles EBCDIC to ASCII translation.

Examples:

```
pcmsmail.exe -h "moon" -f "PM@xyz.com"
     -file "C:\temp\mailbody.txt" -s "About PM"
     -t "fred" –m %DM_PROG%\mapping.txt

pcmsmail.exe -h "jupiter" -f "CM"
     -file "H:\tmp\tmpfile.txt"
     -s "Created New Product" -t "fred,bill,amy"
     -c "robert,alex@xyz.com"
```

Accessing the Email Client from Dimensions for z/OS

You can access the default email client `pcmsmail` in MVS or USS batch mode from Dimensions for z/OS. The following is some sample JCL for performing this task:

```
//STEP    EXEC PGM=MDHLMAIL,PARM='POSIX(ON)/'
//STEPLIB  DD  DISP=SHR,DSN=MDH.Vnnn.MDHLLIB
//         DD  DISP=SHR,DSN=MDH.Vnnn.MDHLLPA
//SYSOUT   DD  SYSOUT=*
//SYSPRINT DD  SYSOUT=*
//NETRC    DD  DISP=SHR,DSN=&SYSUID..R.NETRC   <- optional
//INPUT    DD  *
-s "subject line"
-f user
-t user,user,user...
-m mapping file
-h host
```

```
-file DD:TEXT
-u userid
-p password
-v                          <- must be last
//TEXT     DD  *
message text here
/*
```

The INPUT file enables many parameters to be specified. The parameters are read first from the PARM=statement, and then from INPUT.

You can continue any field over multiple lines. Embedded spaces are lost if you don't quote the string.

The NETRC is defined as for FTP. You can use it for logins to servers if this is required. However, the host name is matched exactly—there is no intelligence regarding domains.

DM_DMEMAIL

For details about this Dimensions CM symbol, the associated subsymbols DM_DMEMAIL_*, the dmemail utility itself, and notification templates, see the *Dimensions CM online help*.

Printing Data

DM_PRINT

This Dimensions CM symbol provides an interface to a print tool. When Dimensions CM commands require printing of various reports they call this interface.

The server configuration file dm.cfg contains a configuration record for the print tool. By default this is set to access the Windows server command lpr on a server caller *server*:

```
lpr -S server -P printer
```

This may be changed, either to reference the server machine and printer you use for the lpr command, or else changed entirely to whatever command you use for printing.

Clearing Up Temporary Files Created by User Reports

DM_DELETE

The DM_DELETE Dimensions symbol is used in conjunction with running Dimensions User Reports. The value specified in the dm.cfg file is used to specify the tool/command to be used to clear up temporary files created during the construction of the script files that run a particular user report. A valid value is anything that makes sense in the context of the file including a command file (remembering that the last line is to delete itself).

The following is an example of a temporary files generated during the running of the reports (the bold type illustrates the value taken from the dm.cfg file):

```
@echo off
rem invoke user report - generated 11:42:17 7 January 2019
call C:\Program Files\OpenText\Dimensions <version>\CM\
    temp\pt9344\myreport.out "PAYROLL"
del /f C:\Program Files\OpenText\Dimensions <version>\CM\
    temp\pt9344\myreport.out
del /f  pcms9342.bat
```

Reporting and Optional Symbols

The Dimensions configuration file `dm.cfg` contains configuration records to control reporting message levels and optional controls. These should *not* be adjusted without explicit instruction from Support.

Windows Server Log On Domain

DM_LOGON_DOMAIN

If the Windows Server system is one of a group of machines within a Windows Server Domain and is not the Domain Server, you may specify the name of the Windows Server Domain in the configuration variable `DM_LOGON_DOMAIN`. This Windows Server Domain is then used to log on all remote users (against this Domain Dimensions CM validates users and passwords)—the local user database is not consulted.

The server configuration file `dm.cfg` contains a record for the Windows Server Domain that is by default set to <null>. This may be changed to specify an appropriate name:

```
<name_of_domain>
```

You can test such a configuration by:

- Test 1: Dimensions CM desktop client on Machine 1, Dimensions CM Server on Machine 2 and Windows Server Domain server on Machine 3. Machine 2 is a member of Machine 3's domain.
- *Test 2:* Connect with Dimensions CM client from Machine 1 with a remote connection
  Specifying a user that is registered on Machine 2 should succeed.
- *Test 3:* Connect with Dimensions CM desktop client from Machine 1 with a remote connection
  Specifying a user that is not registered on Machine 2 but is registered on the Server domain should fail.
- *Test 4:* On Machine 3, add to the configuration file: `DM_LOGON_DOMAIN name_of_domain`.
  Restart the Dimensions CM Windows Service. Then repeat the above test, and it succeeds.
- *Test 5:* Repeat Test 3 with a user that is not defined in the domain.
  Check that the error message is the same as the first failure.

Networking and Replication

See "Networking and Replication" on page 49.

Code Page Settings

DM_CODEPAGE

See "DM_CODEPAGE" on page 49.

DM_EBCDIC_CODEPAGES

See "DM_EBCDIC_CODEPAGES" on page 50.

DM_DEFAULT_CODEPAGE

See "DM_DEFAULT_CODEPAGE" on page 50.

Oracle Collating Sequence for 'ORDER BY' Queries

DM_NLS_SORT_PARAM

See "DM_NLS_SORT_PARAM" on page 50.

Dimensions Web Tools URL Parameters

DM_WEB_URL

See "DM_WEB_URL" on page 51.

DM_INET_PORT

See "DM_INET_PORT" on page 51.

DM_INET_SERVER

See "DM_INET_SERVER" on page 51.

Additional Dimensions Configuration Symbols

Adding Template Variables

You can add template variables to `dm.cfg`. See "Adding Template Variables" on page 52.

Restricting Visibility of Objects in Web Tools

You can add the configuration symbol DM_PRODUCT_USER to dm.cfg to restrict the visibility of Dimensions CM objects in the web tools, for example, the desktop and web clients, and the Administration Console. This symbol is off by default.

See "Restricting Visibility of Objects in CM Clients" on page 52.

Using Network Attached Storage Devices for Item Libraries

To set up and access a Dimensions CM item library on a Windows Network Access Storage (NAS) device connected through a network share, you need a set of user credentials to log in to the network share. Add one of the user login configuration symbols:

```
DM_LIBRARY_USER_<product id>_<item type> <username>
DM_LIBRARY_USER_<product id> <username>
DM_LIBRARY_USER <username>
```

to the `dm.cfg` file located on the Dimensions CM *server* (*not the* Dimensions CM *library server*) and registering that user's password using the `dmpasswd` utility.

See "Using Network Attached Storage Devices for Item Libraries" on page 263 for details on which particular configuration symbol to use depending on the type of item library or item type.

Smart Attribute Validation

See "Smart Attribute Validation" on page 54.

Using PAM on Dimensions Servers

See "Using PAM on Dimensions Servers" on page 54.

Unable to Log In from Dimensions Desktop Client

Over a high-latency connection to a Dimensions CM server on Windows Server 2003, logging in from the desktop client may sometimes fail and immediately prompt you to try again to log in. To fix this problem, set the following value in the `dm.cfg` file on the Dimensions CM server:

```
DM_NETCLOSE_TIMEOUT 2
```

Restart the Dimensions CM Listener after you make these changes.

IMPORTANT! Make sure that none of the configuration symbols are duplicated in the `dm.cfg` file.

Using the Library Cache with the VS .NET SCC Integration

To use library cache areas with the Visual Studio .NET SCC integration, you must set the following configuration property in the `dm.cfg` file:

```
DM_SCC_NOEXPAND     d
```

Integrating with Visual Basic Through SCC

If you are integrating with Visual Basic 6 via SCC, to recognize files stored in subfolders set the following variable in the Dimensions server configuration file, dm.cfg:

```
DM_POPULATE_ALL Y
```

Specifying Server Domain name

See "Specifying the Server Domain Name" on page 54.

File Compression Settings

See "File Compression Settings" on page 55.

Item Header Substitution Last Record Behavior

See "Item Header Substitution Last-Record Behavior" on page 55.

Disabling Branch Check-ins

See "Disabling Branch Check-Ins" on page 55.

Enabling Global Project Structure Change Control

See "Enabling Global Project Structure Change Control" on page 56.

Privilege Caching

See "Privilege Caching" on page 57.

Cleaning Up Temporary Build Files

See "Cleaning Up Temporary Build Files" on page 57.

Build Error Handling

See "Build Error Handling" on page 58.

DM_BLD_ERROR_INVALID_REVISIONS

For details of this symbol, see the *Dimensions Build online help*.

DM_BUILD_DMORDERING

For details of this symbol, see the *Dimensions Build online help*.

DM_BLD_GETSRC_FUNCTION_OR

For details of this symbol, see the *Dimensions Build online help*.

Supporting Refactoring Changes When Creating Revised Baselines

See "Supporting Refactoring Changes When Creating Revised Baselines" on page 59.

Ordering of Attributes

See "Ordering of Attributes" on page 59.

Visibility of Requests on Design Part Structure

See "Visibility of Requests on Design Part Structure" on page 60.

Item Library Security

See "Item Library Security" on page 60.

Disabling the Automatic Selection of Build Targets

See "Disabling the Automatic Selection of Build Targets" on page 61.

Contents of Temporary Build Project

See "Contents of Temporary Build Project" on page 61.

Old Style Item Header Substitution Functionality

See "Pre-version 8 Item Header Substitution Functionality" on page 61.

Display Related Items for Child Requests in Request Browse Templates

See "Display Related Items for Child Requests in Request Browse Templates" on page 62.

Processing Items or Requests That You Do Not Locally Own

See "Processing Items or Requests not Owned Locally" on page 62.

Checking of Mandatory Multi-Valued Attributes in a Block Attribute

See "Checking Mandatory Multi-Valued Attributes in Block Attributes" on page 63.

Event Tracing

See Appendix J, "Logging and Tracing Facilities".

Disabling Editing Block Attributes on Multiple Requests

See "Disabling Editing of Block Attributes on Multiple Requests" on page 63.

Pre-10.1.3 Behavior For Deleting Non-Tip Item Revisions

See "Pre-10.1.3 Behavior For Deleting Non-Tip Item Revisions" on page 64.

Pre-2009 R1 Item Library Access Mechanism

See "Pre-2009 R1 Item Library Access Mechanism" on page 64.

Visual Studio Clients: Prompting for Check Out

IMPORTANT! The check out prompting described below is only available for Dimensions CM projects (which support pessimistic locking); it is not supported for Dimensions CM streams (which support optimistic locking).

In Dimensions CM 12.x and later, if you have Dimensions CM for Windows clients installed (either on the same node as the Dimensions CM server or on remote Windows node) and those clients utilize the Dimensions CM Visual Studio clients integration to communicate with the Dimensions CM for Windows server, you can specify that you want to receive a prompt to check out a file every time you modify it. To do this, add the following symbol to the Dimensions CM server `dm.cfg` file and set it to any value, for example:

```
DM_VISUALSTUDIO_PESSIMISTIC          Y
```

The Dimensions CM listener needs restarting for the change to become effective.

Controlling Deployment Regression

See "Controlling Deployment Regression" on page 65.

Checking the Client Version

To make sure that all Dimensions CM clients are the same version as the server, add the following symbol:

```
DM_REJECT_OLDER_CLIENTS
```

If a client version is older than the server version, the client is not connected and the following message is displayed:

```
The version of your Dimensions CM client is older than the version of
the server. Please upgrade your client.
```

If you do not add `DM_REJECT_OLDER_CLIENTS`, all clients are connected.

Checking Baseline Consistency

Baseline consistency checks enforce additional verifications on the requests that are used to create a revised baseline (CRB) or a request template baseline (CBL). These checks ensure that:

- The requests used to create a new baseline provide a complete set of changes.
- Any dependent changes are not missing.
- The changes have been fully implemented.

To enable baseline consistency checks add the following symbol:

```
DM_ENABLE_BASELINECHECKS Y
```

For details, see the *Dimensions CM online help*.

Setting the DBIO Library

To set the DBIO library used to connect to the database, add the following symbol:

```
DBIO_LIBRARY
```

Specify one of the following variables:

- Oracle 11 Unicode UTF-8 AL32UTF8: `dbio_srv_oci8_11107.dll`
- Oracle 12 Unicode UTF-8 AL32UTF8: `dbio_srv_oci8_12101m.dll`
- PostgreSQL: `dbio_srv_postgres.dll`

Specifying the Database Vendor

To set the DBIO library used to connect to the database, add the following symbol:

`DM_DEFAULT_DBMS`

Specify one of the following variables:

- `MSSQL`
- `PSQL`

Overriding Host Name Assigned to DMSERVER Certificates

See page 51.

Dimensions for z/OS Configuration Symbols

See the "Customizing Variables in the Dimensions Configuration File" section in the "Installing Dimensions for z/OS" chapter of the *Dimensions for z/OS Guide*.

Defaults for Copying Build Information

See "Defaults for Copying Build Information" on page 69.

SSO Configuration Symbols

See "SSO Configuration Symbols" on page 70.

CAC Configuration Symbols

See "CAC Configuration Symbols" on page 71.

Chapter 4

Setting Dimensions Windows Client Symbols

All Windows Client Nodes	92
Modifying or Adding *dm.cfg Entries*	*92*
Installation and File Location Symbol	93
Printing Data	93
Connecting from a Windows Client to a Non-Default Listener on UNIX	93
Turning Off SCC Server Confirmation Pop-Up Messages	94
Using Dimensions CM SCC with Visual Studio Web Projects	94
Visibility of Requests on Design Part Structure	94

All Windows Client Nodes

Each Dimensions® CM Windows client components installation contains a configuration file `dm.cfg` located in directory `%DM_ROOT%` that specifies Dimensions CM Windows configuration symbols for that particular installation (a larger similar named file is created for Dimensions CM server installations on the server node). If the installation is part of a larger network of Dimensions CM installations, care must be taken to ensure that configuration settings that affect communication with other installations are the same.

The directory `%DM_ROOT%` is specified by the registry key:

```
HKEY_LOCAL_MACHINE | SOFTWARE | Serena | Dimensions | <version> |
    DM_ROOT
```

This is set during the installation and is the default Dimensions CM configuration for all users of the machine.

NOTE The value of the `DM_ROOT` variable must always end with \ (backslash).

The `dm.cfg` settings are installation-wide. Parameters specific to you must be set in your own particular environment. If settings are made both in the `dm.cfg` configuration file and in your own particular environment, your environment takes precedence.

These configuration parameters and environment variables are generically referred to as "Dimensions CM Symbols" to reflect that analogous assignment mechanisms are used for the other Dimensions CM operating system-specific products. Most Dimensions CM Symbols are supported on all the operating systems.

Only those symbols that are recommended for modification on the client node are discussed in this chapter.

Modifying or Adding *dm.cfg* Entries

CAUTION! When modifying or adding dm.cfg entries, make sure that the entries do not contain trailing whitespace characters, as this may result in an error being generated. For example, an entry of "`DM_PROG<tab>%DM_ROOT%prog\<space>`" results in failure to connect to a Dimensions CM application.

Most of the installation-wide entries in `dm.cfg` should be left unaltered, that is, they should *not* be changed by end users.

IMPORTANT! In most cases, for changes in `dm.cfg` to become effective, the Dimensions listener must be restarted once `dm.cfg` has been edited.

The following entries in `dm.cfg` *may* be altered if desired. See the indicated page in this chapter for further discussion of these parameters.

dm.cfg Entry	Page
DM_TMP	93
DM_PRINT	93

Installation and File Location Symbol

DM_TMP

This is defined in the installation dm.cfg file and is specific to the installation—normally it should not need adjustment. This symbol controls the location of a temporary area for general tools.

Printing Data

DM_PRINT

This Dimensions CM symbol provides an interface to a print tool. When Dimensions CM commands require printing of various reports they call this interface.

The Dimensions CM configuration file `dm.cfg` contains a configuration record for the print tool.

This may be changed, either to reference the printer you use, or else changed entirely to whatever command you use for printing.

Connecting from a Windows Client to a Non-Default Listener on UNIX

"Running Multiple CM Instances on a UNIX Host Machine" on page 53 describes how to set up multiple Dimensions CM listeners on a UNIX host using a non-default UNIX socket, and the changes you have to make to the `listener.dat` file to enable this. If you want to connect to one of these non-default socket from a Dimensions CM for Windows client, you must add a `DM_LOCAL_CONNECT` symbol to the appropriate client `dm.cfg` file as explained in the above cross-reference.

Turning Off SCC Server Confirmation Pop-Up Messages

To turn off SCC server confirmation pop-up messages, add the following line to the client `dm.cfg` file:

```
DM_SCC_CONFIRMWARNINGS N
```

NOTE Warnings are still displayed in the console or output window.

Using Dimensions CM SCC with Visual Studio Web Projects

Visual Studio reports `"Unable to Retrieve Folder information from the server"` when using file share web projects. This is due to IIS not allowing periods `"."` at the start of filenames to restrict access that conflicts with the default Dimensions CM metadata directory name of `'.metadata'`.

The solution is to change the metadata directory used by Dimensions CM using the following client `dm.cfg` symbols. The following example uses `_dmmeta` as the metadata directory name. These changes should be applied to all clients where file share web projects are controlled by Dimensions CM SCC.

```
DM_CONFIG_METADATA_SUFFIX _dmmeta-config
DM_ITEM_METADATA_SUFFIX _dmmeta-item
DM_ITEM_PROPS_METADATA_SUFFIX _dmmeta-item_props
DM_DIR_METADATA_SUFFIX _dmmeta-dir
```

This changes the metadata directories to _dmmeta (you can choose another name but it needs to match the initial part of the suffixes).

The implication is that the desktop client, command-line client (dmcli), and SCC may not recognize the original default name for metadata. A clean refetch of Visual Studio Solutions and any other Dimensions CM-controlled files is required to regenerate the metadata in the new directories. A further implication is that web client may be unable to recognize the new metadata directories.

Visibility of Requests on Design Part Structure

See "Visibility of Requests on Design Part Structure" on page 60.

Chapter 5

Administering Licenses and the License Server

Overview 96

Overview

Dimensions CM uses the OpenText AutoPass License Server (APLS) to organize and manage licenses.

For details about installing, configuring, and managing APLS, see the Dimensions CM licensing.

You can install APLS by running an installer on a designated machine, or set it up as a Docker image. After installation, APLS is automatically started as a service.

The Dimensions CM server communicates with APLS via HTTPS.

By default, APLS is set up to accept connections via HTTPS only and requires an SSL certificate. If installing APLS on the same machine as Dimensions CM, you can use the sample localhost certificate. To use APLS on a remote machine, you need to configure APLS with a signed SSL certificate.

NOTE You can install Dimensions CM with an evaluation option which is valid for 30 days and doesn't require additional configuration. Select the evaluation option during installation, or specify the `DM_LICENSE` variable in the server's `dm.cfg` configuration file:
`DM_LICENSE EVAL`
After the evaluation period expires, you need to upgrade to a full license and configure Dimensions CM to use APLS.

After configuring APLS and obtaining license keys, you install them, assign licenses to users, and track license usage in the APLS web interface.

Chapter 6

Configuring and Using Auto Update

Introduction	98
Changes to Installation Files	98
Silent Install Options	99
Configuring Auto Update on a CM Server	99
Configuring Auto Update on Clients and Agents	102
Running Client Auto Update	102
Running Agent Auto Update	103
Listing Upgrade History	103
Getting Update Statistics	104
Scheduling Auto Update on a Client or Agent	104
Solutions to Common Issues	105

Introduction

Auto update is the process of managing the automatic update of Dimensions CM clients and agents. Auto update has the following benefits for administrators:

- Centralized control over your Dimensions CM software updates.
- Semi-automated roll-out of client software upgrades (Windows only).
- Fully automated roll-out of agent software upgrades (except Dimensions for z/OS).
- Flexible upgrade process to suit the needs of your environment.
- Audit trail of updates across your network.
- Scheduling of updates.

IMPORTANT! Auto update is not supported on these platforms:

- Linux on z Systems
- z/OS (MVS)
- Linux IA64 (Itanium)

Changes to Installation Files

The following changes have been made to the Dimensions CM installation files:

CM Server Installations

- New Dimensions listener service process:

  `dmupgradesrv/.x/.exe`

- New configuration file:

  `dfs/upgrade_config.dat`

CM Agent and UNIX Client Installations

- New Dimensions listener service process:

  `dmupgradecln/.x/.exe`

- New configuration file:

  `dfs/upgrade_config.dat`

CM Client Installations (Windows)

- New service:

  `Dimensions Update Service`

- Processes:

  `dmpool, dmlsnr, dmupgradecln.exe`

- New "startup" entry:

  `Dimensions CM Auto Update`

New process: `dmupgradecln_ui.exe`

- New configuration file:

  `dfs/upgrade_config.dat`

Silent Install Options

During the silent installation of clients and agents you can add values that point to a server hosting the installation packages for auto update. Add the following parameters to the installation command line:

`DM_SERVER_HOST_NAME=`

`DM_SERVER_PORT=`

The following example runs the install silently with default features, logs the installation, and installs it into a custom location:

```
<Dimensions client installer executable> /s /V"/qn /norestart /L*vx
"%TEMP%\install_client_14.7.log" INSTALLDIR=\"C:\Program Files\OpenText\Dimensions
<version>\CM\" DM_SERVER_HOST_NAME=CustomServer DM_SERVER_PORT=671"
```

Configuring Auto Update on a CM Server

1 Check that the file `dfs/upgrade_config.dat` on your server is configured:

`database=<connection string>`	Specifies a CM database connection string.
`host=<host name>`	Specifies a server host name.
`dmuser=<user>`	Specifies a user that has access to CM.
`schedule_0=* * * * *`	Specifies how often the server polls the database for installation packages. The five digits specify: ■ MINUTES: Minutes within the hour (0-59) ■ HOUR: The hour of the day (0-23). **Tip**: To poll at regular intervals, divide by the number of hours. For example, to poll every 6 hours: */6 ■ DOM: The day of the month (1-31) ■ MONTH: The month (1-12) ■ DOW: The day of the week (0-7) where 0 and 7 are Sunday. See the examples on page 102.

2 To specify the baseline type used to create the installation package add the following variable to the server's `dm.cfg` configuration file:

DM_CIP_TYPE_NAME <type>

NOTE The baseline type used to create the installation package must exist and have a valid life cycle associated with it.

3 Create the attribute types used to designate the type of installation package and the platform where it can be used. Attributes are single value string fields with a maximum length of 240 characters. The mandatory attributes are:

- AU_VERSION="<value>"

  Specifies the version of Dimensions CM in the installation package, for example: "14.5"

- AU_PRODUCT="<description>"

  Describes the installation package, for example: "Dimensions CM Client for Windows 64-bit"

- AU_TYPE=DmCmClient | DmCmAgent

  Specifies the CM installation type (client or agent).

- AU_PLATFORM=<platform>

  where <platform> specifies the target platform of the installation package:

win32	Windows 32-bit
win64	Windows 64-bit
redhat-amd64	Linux 64-bit
hpuxia64	HPUX Itanium
aix64	AIX
solaris64	Solaris
linux-ia64	Linux Itanium
linux-s390x	Linux zSeries

4 Prepare a location on disk where you create the installation package, for example:

/cm_w32_client/

5 Download the installation binaries from the Support site to the installation package directory that you created in the previous step.

6 Specific scripts are required to:

- Run the Dimensions CM installer in specific modes.
- Supply custom installation options.

Windows:

- setup.cmd: renamed from one of the following scripts:
  - setup_client_win.cmd: runs the installer in silent mode. Used by client auto update to display a notification in the Windows taskbar (rename to setup.cmd).
  - setup_agent_win.cmd: runs the installer in silent mode. Used by agent auto update (rename to setup.cmd).

- `gui_setup.cmd`: runs the installer in a reduced interactive mode. Used when client auto update is run from the desktop client (Help | About Dimensions).

UNIX scripts:

- `setup.sh`: runs the installer in silent mode and is used by agent auto update.

  NOTE Only works on Linux. To use it on Solaris and AIX, you have to translate the syntax.

- `options_agent`: used by `setup.sh` to supply the installation options.

Add the scripts that you require to the installation package folder that you created previously. Examples of the scripts are located in:

```
<DM ROOT>/AutoUpdate_Examples
```

NOTE: To enable Windows clients to be updated from the taskbar and the desktop client, add both `setup.cmd` and `gui_setup.cmd`.

7 The scripts contain references to the machines hosting the upgrade servers. Edit the following scripts to match your configuration.

Windows (`setup.cmd` for client and agent auto update):

```
AU_SERVER_HOST_NAME="localhost"
AU_SERVER_PORT="671"
DM_SERVER_HOST_NAME="localhost"
DM_SERVER_PORT="671"
```

UNIX (`options_agent`):

```
auServer="host name"
auPort="671"
```

NOTE: `gui_setup.cm` runs the installer in interactive mode and the silent options are not applicable.

After updating these scripts you are ready to create the installation package.

8 Run the `CIP` (Create Installation Package) command, for example:

```
CIP /PRODUCT=QLARIUS
/DIRECTORY="/cm_updates/cm_w32_client"
/ATTR=(AU_PLATFORM=WIN32, AU_VERSION=14.5.00, AU_TYPE=DmCmClient,
    AU_PRODUCT="Dimensions CM Client for Windows 32-bit")
```

When the command has completed successfully information is displayed about the baseline that was created.

See the *Command-Line Reference* for full details.

NOTE

- Depending on your update schedule (see page 99) the new installation package may not be available immediately from the upgrade server. To make it available run the following command:

  ```
  getupgradestats -refresh_updates
  ```

  The installation package is now registered on the upgrade server and can be picked up by clients and agents.

- To include CMD files in an installation package, create an Upload Rule to upload `.cmd` files in ASCII text format.

Example Schedules

You can specify how often a server polls a database for installation packages. Example schedules:

- (Default) Every 3 hours on the hour:

  ```
  schedule_0=0 */3 * * *
  ```

- Twice a day on the hour:

  ```
  schedule_0=0 */12 * * *
  ```

- Midnight every Sunday:

  ```
  schedule_0=0 00 * * 7
  ```

- 1st day of each month at 22:30:

  ```
  schedule_0=30 22 1 * *
  ```

- 10th of January at 23:00:

  ```
  schedule_0=00 23 10 1 *
  ```

Configuring Auto Update on Clients and Agents

Check that the configuration file `dfs/upgrade_config.dat` on your clients and agents is configured correctly:

`mode=client|agent`	Specifies if the local machine is an agent or client.
`server=<server:port>`	Specifies the server name and port hosting the installation packages.

Running Client Auto Update

NOTE Only supported on the following platforms:

- Windows
- Red Hat Linux
- AIX
- Solaris
- HP-UX

Windows:

1. If an update is available the Dimensions CM icon in the taskbar displays a notification. Click the icon.
2. In the Update Available dialog box click **Download**.

3 After the update has downloaded the Update Available dialog box appears again. Click **Install now**. The update runs silently in the background. When the installation is complete a message is displayed.

All other platforms:

If an update is available, the download and install process starts automatically.

Running Agent Auto Update

NOTE Only supported on the following platforms:

- Windows
- Red Hat Linux
- AIX
- Solaris
- HP-UX

Auto update on agents is initiated by running the following command on a server:

```
AUPG
/NETWORK_NODE=<node name>
```

This command connects to each node being updated and starts the remote update process. Examples:

- To start the upgrade process on the node ST6123:

  ```
  AUPG /NETWORK_NODE=ST6123
  ```

- To start the upgrade process on all registered Dimensions CM agent nodes whose host name matches the pattern `ST-WIN-0*`:

  ```
  AUPG /NETWORK_NODE=ST-WIN-0*
  ```

Listing Upgrade History

On agents you can list the upgrade history across your environment using the `LUPG` command:

```
LUPG
[/HISTORY_TYPE=ALL | LATEST]
[/NETWORK_NODE=<node name>]
```

where:

- `/HISTORY_TYPE=ALL | LATEST`

  Specify LATEST to only include the most recent upgrade history for each network node.

- `/NETWORK_NODE=<node name>`

Only list history for the specified node.

For example, to list the latest upgrade history for the network node ST6123:

```
LUPG /HISTORY_TYPE=LATEST /NETWORK_NODE=ST6123
```

Getting Update Statistics

Use the `getupgradestats` utility to get information about:

- Installation packages that are available.
- Clients and agents that have connected.

Command syntax:

```
getupgradestats
```

`-refresh_updates`: reloads the update map from the database.

`-dump_update_map`: displays a list of available updates.

`-dump_client_map`: displays a list of connected clients.

Scheduling Auto Update on a Client or Agent

To schedule how often a client or agent checks for updates, customize the following setting in the configuration file `dfs/update_config.dat` on each node:

`schedule_0=0 * * * *` Specifies how often the client or agent polls the server for updates. The five digits specify:

- MINUTES: Minutes within the hour (0-59)
- HOUR: The hour of the day (0-23)
  Tip: to poll at regular intervals divide by the number of hours. For example, to poll every 6 hours: */6
- DOM: The day of the month (1-31)
- MONTH: The month (1-12)
- DOW: The day of the week (0-7) where 0 and 7 are Sunday.

See the examples on page 102.

Solutions to Common Issues

Enable tracing

Use DM_SDP_TRACE to enable tracing information to be produced by all auto update processes.

A client has not notified a user that an update is available

The AUPG command fails with the message "no update available".

1. To check if the expected update are listed run this command:

   ```
   getupgradestats –dump_update_map
   ```

2. If updates are not listed run this command:

   ```
   getupgradestats –refresh_updates
   ```

3. To list the clients connected to the server run this command:

   ```
   getupgradestats –dump_client_map
   ```

4. If the client is not connected:

   a. Check the value for `host` in `upgrade_config.dat` on the client.

   b. Restart these processes:

   - Dimensions Update Service
   - Notification area task bar

   c. If the problems persists check the SDP logs on the client.

A client does not complete the download of an update

1. Restart the notification area task bar process. This should re-initialize the communication with the server.
2. Enable and examine the SDP trace logs on the client.

Chapter 7

The Dimensions CM Server HTTP Connector

Introduction 108

Configuring the Connector 108

Introduction

HTTP/S may be required for security reasons or for compatibility with network infrastructure such as proxies and firewalls. This is particularly true where the connection between a Dimensions CM server and clients goes over a public or virtual private network.

The Dimensions CM Server HTTP Connector allows clients to connect to a server using the HTTP/S network protocols instead of the default Standard Dimensions Protocol (SDP). The connector accepts network connections using HTTP/S and forwards the traffic to the server. On Windows there is a new service, `Dimensions HTTP Connector`.

The connector automatically starts and stops as part of the startup and shutdown of your Dimensions CM server. You can manually control the connector as follows:

- Windows: stop and start the *Dimensions HTTP Connector* service.
- UNIX: run the standard `dmstartup` and `dmshutdown` scripts.

The connector is installed by default as part of a server installation.

IMPORTANT! You cannot use the HTTP connector when the server is configured for SSL using the Secure Standard Dimensions Protocol (SSDP). If you need SSL support and want to use the connector, turn off SSDP on your server and use HTTP/S from the clients. For details about SSDP, see the Dimensions CM online help.

Configuring the Connector

The ports and protocol used by the HTTP connector are configured in this file:

- Windows: `%DM_ROOT%\dfs\cmhttpd_config.dat`
- UNIX: `$DM_ROOT/dfs/cmhttpd_config.dat`

Parameters:

- `log`

  Specifies the full path to the connector log file. For example:

  ```
  log=C:\\Program Files\\OpenText\\Dimensions
  <version>\\CM\\Temp\\cmhttpd.log
  ```

- `port`

  Specifies the TCP/IP port number where the connector listens for connections.

  Default: `port=80`

- `remoteHost`

  Specifies the host name of a Dimensions CM server.

  Default: `remoteHost=localhost`

- `remotePort`

  Specifies the port number that the Dimensions CM server is listening on.

  Default: `remotePort=671`

- `ssl`

  Enables HTTPS Secure Socket Layer (SSL).

  Default: `ssl=false`

- `keyStorePath`

  Specifies the full path to the keystore containing a certificate for the server that `cmhttpd` is running on. Only required if SSL is enabled.

  Example: `keyStorePath=C:\\keystore\\mystore.p12`

- `keyStoreType`

  Specifies the type of keystore used to contain the certificate. Only required if SSL is enabled.

  Example: `keyStoreType=PKCS12`

- `keyStorePassword`

  Specifies the password for the certificate keystore. Only required if SSL is enabled.

  Example: `keyStorePassword=my-secure-password`

- `shutdownPort`

  Specifies the local port number the connector listens on to receive shutdown requests.

  Example: `shutdownPort=9095`

Chapter 8

Configuring the Default Merge Tools

Introduction 112

Merge Configuration File 112

Introduction

The default merge tool is Araxis Merge.

Araxis Merge is supported on Windows for the following clients: desktop client, web client (with Web Client Tools), Synchronize Wizard, and the Visual Studio and Windows Explorer integrations.

To configure the default manual merge tool:

1. To use a specific set of manual merge tools, add values to the `native.properties` configuration file. For details, see page 112.
2. To use Araxis Merge, add the variable `DM_MERGE_TOOL` to `dm.cfg` on a client and specify the value `ARAXIS`. If Araxis Merge is not installed, the merge tool, `pvcsmerge`, is used.
3. To use the `pvcsmerge` merge tool, add the variable `DM_MERGE_TOOL` to `dm.cfg` on a client and specify one of these values:
   - `SERENA`
   - `SCRIPT`

To configure the default automatic merge tool, add the variable `DM_MERGE_TOOL` to `dm.cfg` on a client, or an agent on a remote work area, and specify one of these values:

1. To use Araxis Merge, specify `DM_MERGE_TOOL ARAXIS`. If Araxis Merge is not installed the Merge tool, `pvcsmerge`, is used.
2. To use the `pvcsmerge` merge tool, specify `DM_MERGE_TOOL SERENA`.
3. To use a script, specify `DM_MERGE_TOOL SCRIPT`. Edit this file to invoke your preferred automatic merge tool:

   `%DM_ROOT%\Prog\dmdiff.cmd`

   (or the corresponding SH file on Linux)

NOTE The priority in which merge tools are used is the same. For example:

- If you configure `native.properties` and specify `DM_MERGE_TOOL ARAXIS`, manual merge is performed by the tools specified in `native.properties` and automatic merge is performed by Araxis Merge (if installed).
- If you specify `DM_MERGE_TOOL SCRIPT`, manual merge is performed by the `pvcsmerge` merge tool and automatic merge is performed by the tool that you specify in `dmdiff.cmd`.

Merge Configuration File

You can use a configuration file to specify:

- A different default tool, such as the `pvcsmerge` merge tool or any third party application.
- Separate tools for comparing and merging files.

- (Desktop client, Synchronize Wizard, and the Windows Explorer integration) Separate tools for different file extensions.

The configuration file path is:

- Windows:

  `%USERPROFILE%\.dmwebtools\native.properties`

  where `%USERPROFILE%` is, for example:

  `C:\Documents and Settings\Administrator`

- UNIX:

  Your HOME directory, for example: `~/.dmwebtools/native.properties`

This configuration file is created on your machine if you run the native components option when accessing the web client. See the *Dimensions CM online help* for details. If this directory and file do not exist, you need to create them.

NOTE You can enclose `native.properties` values in single or double quotes to allow the use of special symbols. For example, following variable value is in quotes to avoid the truncation on the second equals sign (=):

```
MERGE_PARAMS="/ancestor=%a /left=%d1 /right=%d2 /target=%t"
```

Example Configuration File

In the configuration file example below:

- The merge tool is used for merging.
- A different tool is used for comparing files.
- Different tools are used for files with `.c` extensions and `.doc` extensions.

```
# Use default diff/merge tool
USE_DEFAULTS=0
# Diff/merge tools for all file extensions
MERGE_TOOL=<dimcm-install-dir>\CM\prog\pvcsmerge.exe
MERGE_PARAMS = -NoSplash %c
DIFF_TOOL =C:\Program Files\Microsoft Visual Studio 8\Common7\Tools\
    Bin\windiff.exe
DIFF_PARAMS= %a %d1
# Diff/merge tools for DOC file extensions
MERGE_TOOL_DOC=C:\temp\MyMergeTool.exe
MERGE_PARAMS_DOC = -i %a –f1 %d1 –f2 %d2 –f3 %d3 –o %t –k
DIFF_TOOL_DOC = C:\temp\MyDiffTool.exe
DIFF_PARAMS_DOC = %a %d*
# Diff/merge tools for C file extensions
MERGE_TOOL_C = C:\Program Files\Winmerge\winmerge.exe
MERGE_PARAMS_C = /e %a %d1 %t
# Java specific
JNI_PATH = C:\Documents and Settings\john\.dmwebtools
```

Rules for the File Parser

The first symbol in a configuration file (`USE_DEFAULTS`) defines whether the default tool must be unconditionally invoked or not. If this symbol is equal to 1, the default merge tool is launched even if other symbols specify a different tool.

There are two types of tools that can be defined: a merge and a diff tool. For each tool you must specify a set of parameters. In the example above the symbols `MERGE_TOOL` and `DIFF_TOOL` specify `pvcsmerge.exe` and *windiff.exe* respectively as the merge and diff tools for any type of file. They have their corresponding parameters specified by the symbols `MERGE_PARAMS` and `DIFF_PARAMS`.

The parameters are expanded before the tool is invoked. The expansion replaces all occurrences of:

- %a with the ancestor file path and name;
- %d* with the all derivative paths and names;
- %d1, %d2, %d3 … with the corresponding derivative path and name. Unused %d tokens are removed during expansion;
- %c with the configuration file written for the default tool (`pvcsmerge.exe`);
- %t with target path and name;

You can also define specific tools for any file extension. In the example above, different tools are defined for files with ".doc" and ".c" extensions. The rule is to add additional symbols, ending with '_' + file extension in capitals. For example, for a diff tool for files with a ".doc" extension, define the symbol `DIFF_TOOL_DOC`.

NOTE This option is not used by the web client.

You can only specify one of the tools for a file extension. In the example above, only the merge tool is specified for files with the ".c" extension and `windiff.exe` is invoked when a diff operation is requested.

The string delimiter is the "\" character. Unless a right-hand side value has a "\" char as part of the string, there is no need to use string delimiters.

The `JNI_PATH` symbol can be left blank, in which case it defaults to `%USERNAME%/.dmwebtools` for Windows, and `~/.dmwebtools` for UNIX.

Syntax rules are:

- The order in which symbol definitions appear is not important.
- Symbols cannot be defined by using other symbols.
- All definitions are case-sensitive.
- The assignment operator is the '=' character.
- Spaces are trimmed to the left and right but retained inside strings.
- Line comments are marked with a '#' character. To be considered a comment, the first non-space character of the line must be a '#' char. If a '#' is found anywhere else, it is not considered a line comment mark.
- Blank lines are ignored.
- Lines with syntax errors are ignored by the parser.

You don't need to shut down the client in order for changes made in the `native.properties` file to take effect.

NOTES

- If you specify a generic tool, the only way to identify each file is by its name. Because temporary files are generated with an arbitrary name, the association is difficult to make. When not using the default tool, the name of the temporary file generated by Dimensions CM contains the file specification in a "filename-safe" format.
- Defining specific tools for specific file extensions is not available for the web client.
- Only the extension of the ancestor is considered when trying to identify a suitable tool.
- If any of the following situations occur, the default merge tool is launched:
  - The `USE_DEFAULTS` symbol is equal to `1`.
  - There is no pair (`MERGE_TOOL` and `MERGE_PARAMS`) or (`DIFF_TOOL` and `DIFF_PARAMS`) defined for the given file extension.
  - The `native.properties` file is not present or cannot be read.

Part 2

Database and Tool Administration

Introducing Database and Tool Administration	119
General Operating Guide	131
PostgreSQL Backup and Recovery	145
System Administration	149
Serena-Supplied Runtime RDBMS Database Administration	187

Chapter 9

Introducing Database and Tool Administration

Overview of Dimensions Use of an RDBMS	120
RDBMS DBA	121
System Administrator and Their Responsibilities	121
Users with Administrative Privileges and Their Responsibilities	123
Types of Dimensions Schema	124
Types of Dimensions User	125
Summary of Administrative Personas v Specific Roles or Privileges	126

Overview of Dimensions Use of an RDBMS

A product may easily contain many thousands of hardware, software, and documentation items that require management. As the requirements for a product change, and improvements are identified in the production process, the product items change. Dimensions CM provides the means to identify, track, and control these changes.

Dimensions CM uses a Relational Database Management System (RDBMS) database to control all information relevant to the various aspects of product development. Product items and their various revisions, however, are not stored within the database; they are stored as files resident in operating system directories.

Dimensions CM makes use of the resources provided by the RDBMS to store information about the products it controls. This information is stored in a RDBMS database, referred to by Dimensions CM as a **base database**. Optionally, during installation you can create an initial base database within the RDBMS database. This base database contains all the necessary information to get a Dimensions CM product up and running.

Each base database contains a number of schema objects such as tables, constraints, views, and indexes. These objects are used to:

- Store information on various Dimensions CM products.
- Maintain the integrity of this information.
- Allow users selective access to this information.
- Enhance query response times for the various tools accessing this information.

When a Dimensions CM user is registered using the Dimensions CM Register Users (UREG) function they have access to the Dimensions CM base database within the RDBMS.

More on base databases

A single Dimensions CM server can connect to multiple RDBMS instances and these database instances can also be hosted on machines distinct from the Dimensions CM server, if needed.

Within a single database instance, you can choose to divide up your applications being managed into what Dimensions CM calls *base databases*. In implementation terms, a base database is a schema, for example, in the Serena-Supplied Runtime RDBMS or Oracle Enterprise each base database is an Oracle user with their own copy of the Dimensions CM schema.

You may decide to create different base databases to separate out different applications due to security concerns or simply because the applications are following very different processes.

Within a single base database, Dimensions CM has the concept of *products*. A *product* is a high-level container for related development projects, for example, in your in-house environment you could create a single application called "DIMENSIONS". Some aspects of the development process/workflow can then be customized on a per-application basis (for example, Dimensions CM types, rules, attributes, and functional component breakdown).

Within a single product, many development efforts may be in progress (sometimes in parallel), and Dimensions CM calls these containers *projects* or *streams*. Projects utilize the "lock, modify, unlock" methodology, while streams utilize the "copy, modify, merge" methodology. For example, you could have a Dimensions CM project called "ACME 10" and another project called "ACME 11". You can then perform Dimensions CM operations using these projects, such as:

- Checking files in or out.
- Raising defects.
- Delivering requirements.
- Building and deploying projects.

RDBMS DBA

The RDBMS Database Administrator (DBA) is responsible for maintaining the overall RDBMS database. They have responsibility for such activities as:

- Starting up and shutting down the RDBMS database.
- Taking regular image backups of the RDBMS database files.
- Exporting the entire RDBMS database and/or base databases. These steps are essential to ensure minimal damage in the event of database corruption caused by media failure. For the Serena-Supplied Runtime RDBMS, Serena supplies the `dm_backup` and `dm_restore` utilities to perform these task. If you are using your own Oracle or SQL, Oracle and Microsoft provide similar utilities.
- When necessary, increasing the size of the RDBMS database to provide more space for resident databases.
- Ensuring that the RDBMS has sufficient space to allow new base databases to be created.
- Monitoring the performance of the RDBMS to identify areas for improvement.
- Reorganizing the RDBMS database periodically to reclaim the space unavailable due to fragmentation.
- Creating a database with an RDBMS instance for use by Dimensions CM. The Dimensions CM schema is then loaded into that database by the Dimensions CM System Administrator (DSA). The DSA must be assigned a dedicated (new or existing) operating system user account (we recommend `dmsys`), with which to log in, and their responsibilities are described in the next section.

IMPORTANT! If you are using your own Oracle Enterprise or Microsoft SQL Server Enterprise rather than the Serena-Supplied Runtime RDBMS, this section describes only the responsibilities of the DSA, as it is assumed that all other RDBMS database responsibilities are handled by your own RDBMS DBA.

To support the Serena-Supplied Runtime RDBMS, however, certain maintenance utilities, for example, `dm_backup` and `dm_restore` are documented.

In small companies, the RDBMS DBA and the DSA may be one and the same person.

System Administrator and Their Responsibilities

The Dimensions CM System Administrator (DSA) is responsible for maintaining the Dimensions CM base databases (schemas) and associated system objects. The DSA is

given a dedicated operating system account name (by default `dmsys`), and has responsibility for tasks such as:

- Starting, stopping, and restarting Dimensions CM (see "Starting, Stopping, and Restarting Dimensions" on page 151).
- Loading the Dimensions CM schema into the RDBMS database instance allocated by the RDBMS DBA specifically for use by Dimensions CM. This loading process can be performed as a stage of the Dimensions CM installation procedure, and is described in the related Dimensions CM installation guides. In this section it is assumed that this has already been done.
- Taking regular image backups of Dimensions CM item libraries. This is essential to ensure minimal damage in the event of database corruption caused by media failure.
- Ensuring that the disks hosting the item libraries have an adequate allocation of free space.
- Creating Dimensions CM base databases (see "CRDB – Create Dimensions Base Database" on page 155) and assigning users with Dimensions CM General Administrative Privilege to manage them (see "Assigning General Administrative Privileges" on page 184).
- Deleting any unwanted Dimensions CM base databases (see, "DLDB – Delete Dimensions Base Database" on page 157).
- Listing all Dimensions CM base databases (see "LSDB – List Dimensions Base Databases" on page 158).
- Exporting Dimensions CM base database process model data to a file (see "EXPM – Exporting a Process Model to a File" on page 159)
- Resetting the sequence value of a Dimensions CM base database (see "SSEQ – Resetting the Sequence Value for a Current Base Database" on page 160).
- Changing passwords for databases (see "CPAS – Changing Passwords for Databases" on page 161).

NOTE Only available, and displayed, when connected to a Serena-Supplied Runtime RDBMS database or an Oracle Enterprise database.

- Reporting the space usage of Dimensions CM tablespaces (see "SPAC– Reporting on Tablespace Usage" on page 161).

NOTE Only available, and displayed, when connected to a Serena-Supplied Runtime RDBMS database or an Oracle Enterprise database.

- Increasing database space available to a tablespace (see "INCR – Increasing Database Space Available to a Tablespace" on page 161).

NOTE Only available, and displayed, when connected to a Serena-Supplied Runtime RDBMS database or an Oracle Enterprise database.

- Managing statistics for the database (see "STATISTICS – Manage Statistics for the Database" on page 162).

NOTE Only available, and displayed, when connected to a Serena-Supplied Runtime RDBMS database or an Oracle Enterprise database.

- Installing Dimensions CM Published Views into a base database for use by third-party reporting tools (see "INSV – Installing Report Views" on page 162)
- Deleting Dimensions CM Published Views from a base database (see "DELV – Deleting Report Views" on page 163)
- Granting Dimensions CM users access to the Dimensions CM Published Views in a base database (see "GRTV – Granting Report Views to a Dimensions User" on page 163).
- Revoking Dimensions CM users any access to the Dimensions CM Published Views in a base database (see "REKV – Revoking Report Views from a Dimensions User" on page 163).
- Listing the Dimensions CM users granted access to the Dimensions CM Published Views in a base database (see "LSGV – Listing Dimensions Users Granted Report Views Access" on page 164).
- Starting and configuring Dimensions server pooling (see "Starting and Configuring Dimensions Server Pooling" on page 165).

NOTE The Dimensions CM Published Views mechanism and the individual views available are described in the related document called the *Reports Guide*.

To assist the DSA to perform the above tasks, Dimensions CM provides the following tools:

- A command-line utility called `dmdba`
- The standalone tools `dmlsnr` and `refreshpoolconfig`
- The standalone tools `getpoolstats`

Invoking these tools is described in "Running the Dimensions DBA Tools" on page 133, and the individual commands that can run are described in Chapter 12, "System Administration".

Users with Administrative Privileges and Their Responsibilities

When the Dimensions CM System Administrator (DSA) creates a Dimensions CM base database, an operating system account is assigned to act as the initial manager for that base database. This user is essentially a normal Dimensions CM user to whom special administrative privileges have been extended so as to enable them to perform base database specific tasks with respect to other Dimensions CM users. Such privileged users have the responsibility for:

- Registering Dimensions CM users (see "UREG – Register Users" on page 176).
- Deleting (unregistering) Dimensions CM users (see "XREG – Unregister Users" on page 178).
- Reassigning a Dimensions CM user's registration to a new operating system account (see "RREG – Reassign User Registration" on page 180).
- Creating a credential set to be assigned uniquely to a Dimensions CMuser (see "CCS – Create Credential Set" on page 181).

- Deleting an existing credential set (see "DCS – Delete Credential Set" on page 183).
- Updating an existing credential set (see "UCS – Update Credential Set" on page 183).
- Listing credential sets for a current user (see "LCS – List Credential Sets for Current User" on page 183).
- Reassigning a Dimensions CM user's registration to a new operating system account (see "RREG – Reassign User Registration" on page 180).
- Promoting a normal Dimensions CM user to be an additional manager for a Dimensions CM base database, or demoting such an existing user back to a normal Dimensions CM user (see "Assigning General Administrative Privileges" on page 184).

NOTE Each Dimensions CM base database can have several managers in addition to that initially assigned by the DSA when the base database is created.

- Performing various process model modeling tasks such as creating and deleting lifecycles and products. For details about the tasks related to the process model, see the *online help*.

To assist these users to perform the above tasks, Dimensions CM provides the following tools:

- The standard client command-line interface `dmcli`.
- The Web Client Administration Console.

Invoking these tools is described in "Running the Dimensions DBA Tools" on page 133, and the individual commands that can run are described in Chapter 12, "System Administration".

Types of Dimensions Schema

There are two types of Dimensions CM schema:

- Base database.
- `PCMS_SYS` schema.

Base Database Schema

There may be several base database schemas in the Dimensions CM RDBMS instance. A base database is where project-related data is stored by Dimensions CM, see "More on base databases" on page 120 for a more detailed discussion.

PCMS_SYS Schema

There is only one schema of type `PCMS_SYS` in the Dimensions CM RDBMS instance. It owns tables containing data applicable to the entire Dimensions CM installation. This includes details on the network configuration used by Dimensions CM Network and the installation identification information.

The PCMS_SYS schema is created during installation and provides global data to all the base databases in a single Dimensions CM installation.

Types of Dimensions User

There are two types of Dimensions CM user registered for a given base database:

- Registered users.
- Proxy users.

Registered Users

Using the Dimensions CM UREG command (see "UREG – Register Users" on page 176), the base database manager registers an individual user to access a Dimensions base database.

Proxy Users

Using the Dimensions UREG command (see "UREG – Register Users" on page 176), the base database manager can register a pseudo-user '*' to access a Dimensions base database. This, in fact, enables 'proxy' log in for the base database. This means that any user can access the base database (but subject to the restrictions discussed below) without registering.

The proxy user facility allows large numbers of occasional users to access the base database and enter requests without the overhead of registering them individually through the UREG function. These users do not have access to functions outside the change management area.

Dimensions CM records an individual proxy user's login identity on the first access. This enables Dimensions to track requests to and from that user.

The limited range of Dimensions change management functions available to proxy users comprises:

- Creating requests (for any product).
- Actioning requests in $ORIGINATOR role.
- Browsing requests, if authorized by the applicable Dimensions Product Manager.
- Generating standard change management reports if authorized by the applicable Product Manager.
- Performing query and report functions provided in other areas of Dimensions.

Summary of Administrative Personas v Specific Roles or Privileges

The specific Dimensions CM roles or privileges for particular administrative and management tasks are identified in the following table:

Persona	Dimensions User	Dimensions BaseDB Reach	Dimensions Privileges	Comments
RDBMS DBA	Normally not	Across all base databases	Normally none	[In smaller companies could be the same person as the Dimensions CM System administrator.] Responsibilities include, for example: ■ Starting up and shutting down RDBMS. ■ RDBMS maintenance and performance tuning. ■ Creating the Dimensions CM database instance within the RDBMS. The Dimensions CM schema is loaded into this instance by the Dimensions CM installation process.

Persona	Dimensions User	Dimensions BaseDB Reach	Dimensions Privileges	Comments
Dimensions System Administrator	dmsys (by default)	Across one or more base database	Dimensions CM administrative privileges. For details about the Administration Console functions, see the *online help*.	Responsibilities include, for example: ■ Starting up and shutting down Dimensions CM. ■ Loading the Dimensions CM schema into the allocated RDBMS database instance. ■ Creating Dimensions CM base databases and assigning an initial user with administrative privileges to manage each base database. ■ Assigning a particular process model to a base database. ■ Deleting and listing base databases. ■ Installing and deleting report views to/from a particular base database. ■ Granting or revoking user's the right to access the report views in a particular base database. ■ Starting and configuring Dimensions CM server pooling. ■ Installing and deleting Distributed development (ART, Replicator, and Network Admin). ■ Backing up and maintaining Dimensions CM item libraries.
Dimensions System Administrator (continued)	dmsys (by default)	Across one or more base database		■ Automating the retrieval and deployment of files and directories across an entire network using Serena Mover.

Persona	Dimensions User	Dimensions BaseDB Reach	Dimensions Privileges	Comments
Base database manager	Initially specified by Dimensions System Administrator Each manager can then assign as many additional mangers as they wish.	One specific base database	Dimensions CM administrative privileges. For details about the Administration Console functions, see the *online help*.	Responsibilities include, for example: ■ Registering and managing Dimensions CM users for the base database. ■ Promoting or demoting normal users to or from base database managers. ■ Promoting auto-registered (proxy) users to normal users. ■ Defining one or more products. When defining a product, a Dimensions CM user must be nominated as a Product Manager. Optionally, the role of Part Controller and Change Manager can also be assigned at this point. ■ Defining lifecycles. ■ Defining file formats & MIME types. ■ Defining roles (but not assigning them). ■ Defining upload rules. ■ Defining valid version branch names.
Product Manager	Initially specified by a base database manager. Each Product Manager can then assign as many additional Product Managers as they wish.	One specific product in one specific base database. There can be one or more products in each base database—although often there is only one.	PRODUCT-MANAGER (initial nomination at product definition and subsequent assignments)	Responsibilities include, for example: ■ Administering design part types. ■ Administering design part structures. ■ Administering valid sets. ■ Administering reports. ■ Assigning roles. ■ Defining Object types. These tasks can be delegated to individual users and/or groups by granting them the appropriate privileges.

NOTES

- The Dimensions CM database contains one or more Dimensions CM base databases.
- Each Dimensions CM base database can contain one or more Dimensions CM products. It is quite common, though, just to have one product per base database.
- Although a process model is assigned to a particular base database, some of its properties are base database specific and some are Dimensions CM product specific (as it is also quite common to have just one product in a base database, this distinction is often not a consideration).

  Examples of base-database-wide (and thus potentially cross–product) properties are:

  - Lifecycles.
  - File formats & MIME types.
  - Role definitions (but not assignment).
  - Upload rules.
  - Valid version branch names.
  - User registration.
  - Product definitions.

  Examples of product-specific properties are:

  - Object (request, items, baselines and design parts) type definitions.
  - Administering design part structures.
  - Administering valid sets.
  - Administering reports.
  - Administering role assignments and/or privileges.

Chapter 10

General Operating Guide

General Prerequisites	132
Oracle Privileges Required for DMDBA Commands	133
Running the Dimensions DBA Tools	133
Accessing the Serena-Supplied Runtime RDBMS DBA Utilities	142
Accessing Oracle Enterprise or SQL Server Enterprise DBA Utilities	143
Database Connectivity Mechanisms	143
Database-Dependent Schema Limits	144

General Prerequisites

There are two broad categories of Dimensions CM utilities discussed in this guide:

- Dimensions CM System Administrator (DSA) utilities (principally `dmdba`). These are for managing one or more Dimensions CM base databases and for managing report views (Dimensions CM Published Views) for a specific base database. They can only be used by the DSA (normally user `dmsys`) and are described in Chapter 12.
- Dimensions CM base database manager utilities. These are specific to a particular base database, and are restricted to base database managers connecting to that base database (by default, this also include `dmsys`). They are also described in Chapter 12.

Access to both the above utilities requires these users to be a registered Dimensions CM user. This is automatically the case for `dmsys` (for all base databases) and for the base database managers connecting to a particular base database.

IMPORTANT!

- To use `dmdba` against a Serena-Supplied Runtime RDBMS database or an Oracle Enterprise database, you must ensure you are connected to the `dmdba` tool through an Oracle account name that has DBA privileges. This is because Dimensions CM for these particular types of database relies on database level authentication (unlike Microsoft SQL Server Enterprise, which uses operating system level authentication). Although you can connect to `dmdba` with account names without DBA privilege, execution of any `dmdba` operation returns with an error message.

  Examples of existing accounts with DBA privilege are `SYSTEM` and `SYS` (which, by default, for the Serena-Supplied Runtime RDBMS have the password `MANAGER` and `CHANGE_ON_INSTALL` respectively).

  The Oracle DBA can, of course, set up further Oracle accounts so that they have DBA privilege, see "Oracle Privileges Required for DMDBA Commands" on page 133.
- To use `dmdba` against a Microsoft SQL Server Enterprise database, you must ensure you are running the `dmdba` tool as the operating system user that is the owner of that database.

When registered, the `dmsys` and the base database managers must then ensure that their operating system environment is set up with the correct Dimensions CM access paths to enable them to work on a Dimensions CM server. This is done by invoking the following operating-specific installation/configuration files located in the Dimensions CM installation directory:

- `dmlogin` (C Shell) or `dmprofile` (Bourne Shell) on UNIX. These have to be invoked manually or be invoked from within the `.login` (C Shell) or `.profile` (Bourne Shell) log in scripts.
- `dm.cfg` on Windows. This is invoked automatically.

These files are described in more detail in the *Installation Guide for Windows* or *Installation Guide for UNIX*.

On UNIX, the DSA user account must be a member of the `dmtool` user group.

Oracle Privileges Required for DMDBA Commands

If you are using a "locked down" Oracle Enterprise where you cannot use the Oracle account `SYSTEM` to perform DMDBA commands, you can set up another Oracle account and grant it the requisite privileges as listed in the table below (refer to your Oracle DBA or vendor documentation for details on how to grant such privileges to an Oracle account):

Oracle Privilege	Oracle Privilege
ALTER ANY SEQUENCE	DROP ANY INDEX
ALTER ANY TABLE	DROP ANY PROCEDURE
ALTER DATABASE	DROP ANY SEQUENCE
ALTER SESSION	DROP ANY SYNONYM
ALTER TABLESPACE	DROP ANY TABLE
ALTER USER	DROP ANY VIEW
ANALYZE ANY	DROP USER
CONNECT WITH ADMIN OPTION	EXECUTE ANY PROCEDURE
CREATE ANY INDEX	GRANT ANY OBJECT PRIVILEGE
CREATE ANY PROCEDURE	GRANT ANY PRIVILEGE
CREATE ANY SEQUENCE	INSERT ANY TABLE
CREATE ANY SYNONYM	RESOURCE WITH ADMIN OPTION
CREATE ANY TABLE	SELECT ANY DICTIONARY
CREATE ANY VIEW	SELECT ANY SEQUENCE
CREATE SESSION	SELECT ANY TABLE
CREATE USER	UPDATE ANY TABLE
DELETE ANY TABLE	

Running the Dimensions DBA Tools

The following Dimensions CM interfaces are available for running the Dimensions CM DBA and Dimensions CM base database manager utilities described in this guide:

- **dm_control – Dimensions CM Startup Tool (UNIX only)**

  This tool enables the UNIX account holder `root` to:

  - Start, stop, or restart a Serena-Supplied Runtime RDBMS located on the same server as Dimensions CM (`dm_control rdbms_start`, `dm_control rdbms_stop`, or `dm_control rdbms_restart`).
  - Assuming that the local Serena-Supplied Runtime RDBMS or Oracle Enterprise is up and running, start, stop, or restart Dimensions CM (`dm_control cm_start`, `dm_control cm_stop`, or `dm_control cm_restart`).

See "Invoking the dm_control Tool" on page 136 for details of invoking this tool.

NOTE You must be logged in to the Dimensions CM server to invoke this tool.

- **dmdba – Dimensions CM Interactive DBA Tool**

  This tool has access to the whole Dimensions CM database, and enables `dmsys` to:

  - Create a new Dimensions CM base database (`CRDB`).
  - Delete an existing Dimensions CM base database (`DLDB`).
  - Export Process Model Data (`EXPM`).
  - List all Dimensions CM base databases (`LSDB`).
  - Reset Sequence Value for Current Base Database (`SSEQ`).
  - Change passwords for databases (`CPAS`). *This command is only present when connected to an Oracle database.*
  - Report on tablespace usage (`SPAC`). *This command is only present when connected to an Oracle database.*
  - Increase database space available to a tablespace (`INCR`). *This command is only present when connected to an Oracle database.*
  - Manage statistics for the database (`STATISTICS`). *This command is only present when connected to an Oracle database*.
  - Install Report Views (`INSV`).
  - Delete Report Views (`DELV`).
  - Grant Views to User (`GRTV`).
  - Revoke Views from User (`REKV`).
  - List Users Granted View Access (`LSVG`).
  - Migrate a Serena-Supplied Runtime RDBMS or Oracle Enterprise MBCS encoded database to UTF-8 (MIGRATE).
  - Upgrade pre-Dimensions CM 12.1 databases into the new system (normally done automatically by the upgrade installers).
  - Migrate pre-Dimensions CM 12.1 deployment areas into the new system

  These utilities are described in Chapter 12, "System Administration", Appendix I, "Migrating MBCS Data to UTF-8 Data", and the installation guides (Migrate pre-Dimensions CM 12.1 deployment areas).

  See "Invoking the dmdba Tool" on page 136 for details of invoking this tool.

NOTE You must be logged in to the Dimensions CM server to invoke this tool.

- **Dimensions CM Server Pooling**

  Dimensions CM server pooling enables a number of Dimensions CM application servers to be pre-started prior to clients connecting to the Dimensions CM server. See "Invoking Dimensions Server Pooling" on page 138 for details.

Manually starting and configuring server pooling using the `dmlsnr` tool is described in "Starting and Configuring Dimensions Server Pooling" on page 165.

You can also display server pooling status information by either running the `getpoolstats` command or its Administration Console equivalent.

NOTE You must be logged in to the Dimensions CMserver to invoke `dmlsnr` or `getpoolstats`.

- **dmcli – Dimensions CM Command-Line Interface**

  The standard Dimensions CM `dmcli` tool is used, on a per Dimensions CM base database basis, to access the following administration utilities:

  - Register a new Dimensions CM user (`UREG`).
  - Delete (unregister) an existing Dimensions CM user (`XREG`).
  - Reassign an existing Dimensions CM user to a new operating system user account (`RREG`).
  - Create a credential set (CCS).
  - Delete a credential set (DCS).
  - Update a credential set (UCD).
  - List credential sets (LCS).

  These functions are described in Chapter 12, "System Administration".

  See "Invoking dmcli" on page 141 for details of invoking this tool.

NOTE You can be logged on to either the Dimensions CM server or a Dimensions CM client to invoke this tool.

- **Dimensions CM Administration Console**

  The Dimensions CM Administration Console may be used, on a per Dimensions CM base database basis, to access the following administration utilities:

  - Register a new Dimensions CM user.
  - Enable proxy log in for any user.
  - Delete (unregister) an existing Dimensions CM user.
  - Disable proxy log in for any user.
  - Promote an existing Dimensions CM user to a base database manager.
  - Demote an existing base database manager back to a normal Dimensions CM user.
  - Display Connection Pooling Statistics.

  These functions are described in Chapter 12, "System Administration".

  See "Invoking the Administration Console" on page 142 for details of invoking this tool.

NOTE You can be logged onto either the server or a client to invoke this tool.

Invoking the dm_control Tool

To invoke the UNIX-only `dm_control` tool, log in to the Dimensions CM server as `root`, set up your access to Dimensions CM as explained in "General Prerequisites" on page 132 (but as user `root` in this instance), and type the following command at the operating system prompt:

```
dm_control
```

This displays the command usage and a summary of the supported parameters.

Running the dm_control tools is described in detail in "Starting, Stopping, and Restarting Dimensions" on page 151.

Invoking the dmdba Tool

Various Methods of Invoking dmdba

To invoke `dmdba`, log in to the Dimensions CM server as `dmsys`, set up your access to Dimensions CM as explained in "General Prerequisites" on page 132 (especially take note of the additional privilege requirements if you are using the Serena-Supplied Runtime RDBMS or Oracle Enterprise), and type the following command at the operating system prompt:

```
dmdba
```

This displays the prompt:

```
Enter user-name:
```

What is being requested here is the Serena-Supplied Runtime RDBMS, Oracle Enterprise, or Microsoft SQL Server base database name.

Type

```
<basedb>/<password>@<connect_string>
```

for the Serena-Supplied Runtime RDBMS or Oracle Enterprise, or

```
<basedb>@<connect_string>
```

for Microsoft SQL Server Enterprise

(these parameters are described on page 138)

This displays a banner confirming establishment of connection and the prompt:

```
BASEDB>
```

for example,

```
CM_TYPICAL>
```

As alternatives to the above, you can also connect from the operating system prompt as in the following examples:

- Serena-Supplied Runtime RDBMS or Oracle Enterprise Windows: interactive

```
set TWO_TASK=<connect_string>
dmdba
Enter user-name: <basedb>
```

```
Password: <password>
```

- Serena-Supplied Runtime RDBMS or Oracle Enterprise: non-interactive

```
dmdba <basedb>/<password>@<connect_string>
```

- SQL Server Enterprise: non-interactive

```
dmdba <basedb>@<connect_string>
```

Serena-Supplied Runtime RDBMS or Oracle Enterprise Specific

If you are using the Serena-Supplied Runtime RDBMS or Oracle Enterprise, in addition to needing to be connected with Oracle accounts with DBA privilege as explained in "General Prerequisites" on page 132, two of the Serena-Supplied Runtime RDBMS or Oracle Enterprise `dmdba` operations—Export Process Model Data (`EXPM`) and Reset Sequence Value for Current Base Database (`SSEQ`)—also require additional parameters to be passed to `dmdba` to identify the database that is to be the subject of these operations. These additional parameters are:

```
dmdba -db <DBA_Priv_Account_Name>/
    <password>@<connect_string> <basedb>
```

An example of exporting a process model to a file `acme1.sql` could be:

```
dmdba -db system/manager@dim10 acme1 expm acme1.sql
```

Once connected, you can type

```
BASEDB> help
```

to list the commonly used `dmdba` utilities available, namely:

```
LSDB - List Dimensions Base Databases
Expm - Export Process Model Data
Crdb - Create New Dimensions Base Database
DlDB - Delete Dimensions Base Database
SSEQ - Reset Sequence value for Current Base Database
CPAS - Changing passwords for databases
SPAC - Report on tablespace usage
INCR - Increase database space available to a tablespace
STATISTICS - Manage statistics for the database
MIGRATE - Migrating MBCS Data to UTF-8 Data
INSV - Install report Views
DELV - Delete report views
GRTV - Grant views to user
REKV - Revoke views from a user
LSGV - List users granted view access
ITEMLIBMODE - Enforce a base database-wide item library access policy
```

NOTE `CPAS`, `SPAC`, and `INCR` are only present when connected to Serena-Supplied Runtime RDBMS or an Oracle database. `STATISTICS` is only present when connected to an Oracle database.

The specialized dmdba operation `UPGRADEDEPLOY` (for manually upgrading databases and pre-Dimensions CM 12.1 deployment areas) are not included in the dmdba help system. See the installation guides for details.

Individual help on each utility can be obtained by typing

```
BASEDB> help <utility>
```

for example

```
BASEDB> help crdb
```

For details about the parameters of the above utilities, and their use, see Chapter 12, "System Administration" and "Migrating MBCS Data to UTF-8 Data" on page 565.

To exit the `dmdba` utilities, repeatedly enter

```
exit
```

until you return to the operating system prompt.

Full Syntax for dmdba

```
dmdba [[<option>] [<logon>] [<start>]]
```

<option> Specify one the following options:

Option	Result
-	Displays the usage syntax.
-?	Displays the `dmdba` version banner.
-S	Uses silent mode.

<logon> Specify one or more of the following logon parameters:

Parameter	Description
<basedb>	Name of base database.
/<password>	If the RDBMS you are attempting to connect to is Oracle, you must also supply the `<basedb>` password. An example <basedb>/<password> is `acme/acme`.
@<connect_string>	The data source name (DSN) identifying the database instance.

<start> Can be for running a sequence of `dmdba` commands in a batch file, for example:

```
dmdba system/manager@dim10 lsdb
```

Invoking Dimensions Server Pooling

The Dimensions CM server offers an advanced pooling feature for Dimensions CM application servers when servicing connection requests from Dimensions CM clients. Dimensions CM server pooling is automatically invoked upon Dimensions CM startup. These processes are also automatically stopped upon Dimensions CM shutdown. The

server pooling is invoked with various default parameters, which for many users is perfectly adequate.

NOTE In Dimensions CM for z/OS, a certificate mechanism is used to connect back to Dimensions CM from the mainframe. These certificates are also managed in memory by the pooling feature on the server machine. For further information on this aspect of the server pooling feature and timeouts that can be set, please see "Configuring Dimensions Server Pooling" on page 165.

You can, however, configure the pool manager to meet your own specific needs by specifying parameters such as:

1. The service name (or the TCP/IP port number) to be used for listening to client requests.
2. The number of application servers to be created when the pool manager is first started up.
3. The minimum and maximum number of application servers allowed in the pool.
4. The timeout period for an idle application server to be returned back to the pool.

Please refer to "Starting and Configuring Dimensions Server Pooling" on page 165 for details.

The server utility `refreshpoolconfig` is also available to re-read any new changes to the configuration parameters. The utility `getpoolstats` is available to obtain pool status information.

On a Dimensions UNIX server, automatic server pooling is started and stopped as part of standard running of the following Dimensions server startup and shutdown scripts:

```
$DM_ROOT/prog/dmstartup
$DM_ROOT/prog/dmshutdown
```

or

```
$DM_ROOT/prog/dm_control cm_start
$DM_ROOT/prog/dm_control cm_stop
$DM_ROOT/prog/dm_control cm_restart
```

On a Dimensions Windows server, the automatic server pooling is started and stopped by starting and stopping the following Dimensions Windows service:

```
Dimensions CM Listener Service
```

To manually start/stop server pooling, dynamically reconfigure the parameters with which it is run, or display server pooling statistics, you need to invoke the `dmlsnr`, `refreshpoolconfig`, or `getpoolstats` tool.

On a Dimensions UNIX server:

1. Log in to the Dimensions server as user `root` if you want to run `dmlsnr` (you can log in as any Dimensions user to run `getpoolstats`).
2. Set up your access to Dimensions as explained in "General Prerequisites" on page 132.
3. Type the following command at the operating system prompt

```
dmlsnr -param $DM_ROOT/dfs/listener.dat
```

where `listener.dat` is a configuration file.

NOTE Once started, every application process created in the pool is owned by the operating system account specified by the `-user` parameter in the `$DM_ROOT/dfs/listener.dat` file. By default this account is `dmsys`.

4 If you want to update the pool configuration while Dimensions is running, edit `listener.dat` entries appropriately and type either of the following commands at the operating system prompt (this can be done as either user `root` or `dmsys`)

```
refreshpoolconfig -host machine:port
```

if you are not using the default TCP port of 671 for the Dimensions CM Listener Service, in which case you must specify the appropriate port; or simply

```
refreshpoolconfig -host machine
```

if you are using the default TCP port of 671.

5 If you want to display server pooling statistics, type the following command at the operating system prompt (this can be done as either user `root` or `dmsys`)

```
getpoolstats
```

The statistics output are similar to those described in "Displaying Connection Pooling Statistics" on page 170.

The format of the configuration file `listener.dat` is detailed in "System Administration" on page 149.

On a Dimensions CM Windows server:

1 Log in to the Dimensions CM server as a Windows administrator.

2 Set up your access to Dimensions CM as explained in "General Prerequisites" on page 132.

3 Start the `Dimensions CM Listener Service` Windows service.

4 To update the pool configuration while Dimensions CM is running, edit `listener.dat` entries appropriately and enter either of the following commands at the operating system prompt (this can be done as either user `root` or `dmsys`):

```
refreshpoolconfig -host machine:port
```

if you are not using the default TCP port of 671 for the Dimensions CM Listener Service, in which case you must specify the appropriate port; or simply

```
refreshpoolconfig -host machine
```

if you are using the default TCP port of 671.

5 To display server pooling statistics, enter the following command at the operating system prompt:

```
getpoolstats
```

The format of the configuration file `listener.dat` is detailed in "System Administration" on page 149.

NOTE On Windows hosts, make sure that `registry.dat` contains the credentials of the pool owner user. For example, run the following command:

```
dmpasswd <pool_owner> -add -pwd <password
```

Invoking dmcli

To invoke `dmcli` for the User Administration Utilities:

1. Log in to a Dimensions CM UNIX or Windows client as a registered Dimensions CM base database manager (`dmsys` if no others have yet been created) .
2. Set up your access to Dimensions CM as explained in "General Prerequisites" on page 132.
3. Enter the following command at the operating system prompt

   ```
   dmcli
   ```

NOTE If your Dimensions CM system utilizes a smart Common Access Card (CAC) for log in, you need to specify `dmcli -card`. Please see "Configuring Centralized Network Authentication" on page 435 for details of CAC logging in with dmcli.

4. Fill in the login dialog box operating system and Dimensions CM connection details.

NOTE See the *Command-Line Reference* and the online help for general details about the login dialog box.

Upon successful connection, Dimensions CM displays a banner and the prompt:

```
Dimensions>
```

For details on how to run Dimensions CM `dmcli` commands, see the *Command-Line Reference*.

NOTE If you have installed the server on a Widows 64-bit machine and subsequently installed the client on that machine, you may need to set the path to access the correct 64-bit version of dmcli, as the client install is referencing the 32-bit version. To do this, for example, run the following command:

```
C:\>set PATH=C:\Program Files\OpenText\Dimensions
    <version>\CM\prog;%PATH%
```

NOTE For the purposes of running `dmcli`, a Dimensions UNIX server can be thought of as a superset of a Dimensions UNIX client, that is, you can run `dmcli` on it.

Invoking the Administration Console

IMPORTANT! Only users with requisite administrative privileges (for example, dmsys) can make changes using the Administration Console. All other users can only use it in a "view-only" mode.

The Administration Console is a web application (which requires a Java runtime to be pre-installed).

To open the Administration Console on a Dimensions CM for Windows server:

```
Start | Dimensions CM <version> | Administration Console
```

To open it on a Dimensions CM for Windows agent or client, or a Dimensions CM for UNIX agent or client with X11 Window:

1. Launch your web browser.
2. Specify the URL required to invoke the Administration Console, for example:

```
http(s)://<DimensionsServer_host_name>:8080/adminconsole/
```

See the Dimensions CM online help for details.

Accessing the Serena-Supplied Runtime RDBMS DBA Utilities

Dimensions CM provides the following tools to backup or restore one or all of the schemas in a local Dimensions CM Serena-Supplied Runtime RDBMS:

- dm_backup
- dm_restore

To invoke the `dm_backup` or `dm_restore`, log in to the Dimensions CM server where the local Serena-Runtime RDBMS is located and enter the following command at the operating system prompt:

```
dm_backup
```

or

```
dm_restore
```

This displays the command usage and a summary of the supported parameters.

For details about running the dm_backup or dm_restore tools, see "Serena-Supplied Runtime RDBMS Database Administration" on page 187.

Accessing Oracle Enterprise or SQL Server Enterprise DBA Utilities

Maintenance and general database operations for Oracle Enterprise or SQL Server Enterprise that may be required in addition to those supported by Dimensions CM (for example, to perform a full system export), necessitate the use of tools provided by Oracle or Microsoft. Please consult your RDBMS DBA or vendor documentation.

Database Connectivity Mechanisms

IMPORTANT! The Open Database Connectivity (ODBC) connectivity mechanism is *not* supported by Dimensions CM 12.1 with respect to interacting with Serena-Supplied Runtime RDBMS or Oracle Enterprise databases. For such databases, only the Oracle Open Call Interface (OCI) is supported—this is the official API that Oracle uses to interact with Oracle databases.

The above *does not apply* to using ODBC drivers when using a third-party reporting tool against Dimensions CM Published Views.

Connecting to a Serena-Supplied Runtime RDBMS or Oracle Enterprise

Dimensions CM uses OCI for accessing the Serena-Supplied Runtime RDBMS or the Oracle Enterprise database as it affords the highest performance. If you are using your own Oracle Enterprise, you must use a version of OCI that is supported by Dimensions CM, see the platform matrix.

Connecting to SQL Server Enterprise on a Remote Host

NOTE The following instructions assume that the remote SQL Server Enterprise database has been populated by the Dimensions CM server.

To connect a Dimensions CM server to a SQL Server database running on a remote host, you must perform the following steps:

1. Create a system DSN on the Dimensions CM server host for the remote Dimensions CM database.
2. Ensure that the remote Dimensions CM database is owned by a Windows domain account and not a local user account. The Dimensions CM server host must be a member of such a Windows domain.
3. Edit `%DM_ROOT%\dfs\listener.dat` to ensure that:
   a. The `-dsn` parameter refers to the system DSN created at step 1.
   b. The -user parameter refers to either:

- The full domain name of the remote database owner. If using the full domain name, you also need to ensure that the DM_LOGON_DOMAIN symbol is unset in the %DM_ROOT%\dm.cfg configuration file.
- The (short) username of the remote database owner. If using the short username, you need to set the DM_LOGON_DOMAIN symbol to the corresponding Windows domain name in the %DM_ROOT%\dm.cfg configuration file.

4 Execute the following command:

```
dmpasswd <user> -add -pwd <pwd>
```

where:

<user>	is the value of the -user parameter in %DM_ROOT%\dfs\listener.dat
<pwd>	is the corresponding user password

5 Restart the Dimensions CM listener.

Database-Dependent Schema Limits

Schema Limit	Serena-Supplied Runtime RDBMS, Oracle and SQL Server
Maximum number of user-defined attributes.	996
Maximum length of an attribute value. The DBCS and UTF-8 limits depend on the exact type of data stored, see the note below.	4000 ASCII characters, or 2000 DBCS characters, or 1000 MBCS UTF-8 characters
Maximum length of a valid set value.	1978

IMPORTANT! The maximum DBCS and UTF-8 attribute lengths quoted above are the "safe minimum" maximum lengths, assuming the "worst" case. You can always get at least that many characters of each of the types in the field, but for most types of data you get a lot more. For example, for Oracle UTF-8 data:

- If it is comprised entirely of low-ASCII, the UTF-8 limit is 4000.
- If it is comprised entirely of high-ASCII, Cyrillic, Greek, or similar, the limit is 2000 characters.
- If it is comprised entirely of common CJK ideographs, then the limit is 1333 characters.
- Only if the data is comprised entirely of rare ideographs, mathematical symbols, characters from dead or extinct languages, or similar, the limit falls to the "safe minimum" of 1000 characters.

Chapter 11

PostgreSQL Backup and Recovery

Overview	146
SQL Dump	146
File System Level Backup	146
Continuous Archiving	147

Overview

There are multiple ways to back up and recover PostgreSQL data:

- SQL dump
- File system level backup
- Continuous archiving

SQL Dump

SQL dump generates a text file with SQL commands that recreates the database in the same state as it was at the time of the dump. Use the PostgreSQL utility program `pg_dump` to create a dump, its basic usage is:

```
pg_dump dbname > outfile
```

NOTE

- The utility can also create files in other formats that allows for parallelism and more fine-grained control of object restoration.
- You can perform this backup procedure from any remote host that has access to the database.
- `pg_dump` does not operate with special permissions but it must have read access to all tables that you want to back up. To back up an entire database you typically run it as a database superuser.
- Only use SQL dump when no users are accessing Dimensions CM.

For more information, see the PostgreSQL documentation:

https://www.postgresql.org/docs/current/static/backup-dump.html

File System Level Backup

You can directly copy the files that PostgreSQL uses to store the data in the database. However, there are two restrictions that make this method inferior to the SQL dump method:

- Only enables 'cold backups'. Dimensions CM and the PostgreSQL database server must be shut down to get a usable backup. You also need to shut down the server before restoring the data.
- File system backups only work for a complete backup and restoration of an entire database cluster. You cannot back up or restore specific tables or databases from their respective files or directories.

Continuous Archiving

PostgreSQL maintains a Write Ahead Log (WAL) in the `pg_wal/` subdirectory of a cluster's data directory. The log records every change made to the data files and exists primarily to help recover from system crashes. However, you can combine a file system level backup with WAL file backup. If recovery is needed, restore the file system backup and then replay from the backed-up WAL files to bring the system to a current state. This approach is more complex to administer than either of the previous approaches, but has some significant benefits:

- Can provide 'hot backups' while Dimensions CM is running.
- You do not need a perfectly consistent file system backup as the starting point. Any internal inconsistency in the backup is corrected by log replay.
- Continuous backup can be achieved by continuing to archive the WAL files. This is particularly valuable for large databases, where it might not be convenient to take a full backup frequently.
- It is not necessary to replay the WAL entries all the way to the end. You can stop the replay at any point and have a consistent snapshot of the database as it was at that time.
- If you continuously feed the series of WAL files to another machine that has been loaded with the same base backup file, you can have a standby system with an almost identical copy of the database.

NOTE

- Only supports the restoration of an entire database cluster, not a subset.
- Requires a lot of archival storage. The base backup might be bulky and a busy system generates many megabytes of WAL traffic that have to be archived.
- To recover successfully using continuous archiving you need a continuous sequence of archived WAL files that extends back at least as far as the start time of your backup.

Chapter 12

System Administration

Introduction	150
Starting, Stopping, and Restarting Dimensions	151
Administering Dimensions Schemas	155
Configuring Dimensions Server Pooling	165
Displaying Connection Pooling Statistics	170
Managing Remote Jobs	171
Administering Dimensions Users	174
Backing Up Item Libraries	184
Configuring the dmlcasrv Server	185
Encrypting Usernames and Passwords	185

Introduction

The schema utilities are operative across one or more Dimensions base databases and their use is restricted to the Dimensions System Administrator (normally user `dmsys`). They are all accessed through the `dmdba` tool (unless indicated otherwise), and they allow `dmsys` to:

- Start, stop, and restart Dimensions through the `dm_control` standalone tool,.
- Create a new base database.
- Delete an existing Dimensions base databases.
- List all Dimensions base databases.
- Export the Dimensions process model for a database to a file (this fie can then be used when creating a new database to base the new database on the same process model).
- Reset the sequence value for a currently connected base database.
- Change the passwords for databases.
- Report the space usage of Dimensions tablespaces.
- Increase database space available to a tablespace.
- Manage statistics for the database.
- Install report views into a base database.
- Delete report views from a base database.
- Grant to a user the right to access the report views in a base database.
- Revoke from a user the right to access the report views in a base database.
- List all users who have been granted the right to access the report views in a base database.
- Grant to a user the right to access the report views in a base database.
- Revoke from a user the right to access the report views in a base database.
- Migrate a Serena-Supplied Runtime RDBMS or Oracle Enterprise MBCS encoded database to UTF-8.
- Configure connection pooling,. These actions are executed using Dimensions standalone tools.
- Displaying connection pooling statistics.These actions are executed using Dimensions standalone tools and the Administration Console.
- Managing remote jobs.These actions are executed using the Administration Console.
- Administering Dimensions users. These actions are executed using Dimensions `dmcli` and the Administration Console.
- Backing up item libraries.

Launching the `dmdba` tool and connecting to a particular Dimensions base database are described in "Invoking the dmdba Tool" on page 136. In the remainder of this chapter,

when discussing dmdba commands, it is assumed that the dmdba tool has been launched and connection has been made to the relevant base database.

NOTES

- For the system administration schema utilities described in this chapter, when dmdba is used, the connection must be, by default, to the SYSTEM base database unless stated otherwise. However, if you are using a "locked down" Oracle Enterprise where you cannot use the Oracle account SYSTEM for DMDBA commands, you can set up another Oracle account and grant it the requisite privileges, see "General Prerequisites" on page 132.
- Do not attempt to use RDBMS-specific commands to perform functions similar to CRDB, DLDB, LSDB, or EXPM.
- The individual dmdba tool commands accept hyphen (-) as a continuation character if you want to type parameters on more than one line. Each continuation line begins with a right angle bracket (>).
- The dmdba tool also supports the limited processing of SQL statements for querying and update purposes. The use of dmdba in this mode is not generally discussed in this guide (the ability to process SQL statements is primarily intended to enable experienced RDBMS DBAs to troubleshoot database problems).
- If you specify a UNIX path in a Dimensions command in the dmdba tool, you must enclose it in double quotes. Otherwise, the command does not recognize the path. For example:

```
SYSTEM> expm "/tmp/payroll_01.exp" payroll /basedb=intermediate
```

Starting, Stopping, and Restarting Dimensions

Starting Dimensions for Windows

For Dimensions for Windows to be successfully started, the following need to be running:

- The appropriate Serena-Supplied Runtime RDBMS, Oracle Enterprise, or SQL Server Enterprise Windows service for the instance used by Dimensions.
- The Window service for the License Manager (unless you are running Dimensions with an evaluation license).
- The Windows service for the OpenText Common Tomcat, if you want to run Dimensions Web tools.

If the Dimensions server, database server (Serena-Supplied Runtime RDBMS, Oracle Enterprise, or SQL Server Enterprise), License Manager, and Common Tomcat are all on the same PC, then all the appropriate Windows services should start up automatically when the PC is started. However, if some of the above mentioned components fail to automatically start or are elsewhere on the network, proceed as follows:

- To start the Serena-Supplied Runtime RDBMS.
- To start the other types of database server, consult your DBA or vendor documentation.
- To start the License Manager service.

- To start the Dimensions server (assuming the other components are already started):
  - **a** Log in as a user with local Windows administrative rights.
  - **b** Access Windows services by:

    `Start | Control Panel | Services`

    or

    `Start | Control Panel | Administrative Tools | Services`
  - **c** Right-click `Dimensions CM Listener Service`.
  - **d** Select Start.
  - **e** If Windows service still fails to start, consult the Troubleshooting appendix in the *Installation Guide for Windows*.
- To start the Common Tomcat service.

Starting Dimensions for UNIX

For Dimensions for UNIX to be successfully started, the following need to be running:

- The appropriate Serena-Supplied Runtime RDBMs or Oracle Enterprise UNIX processes for the instance used by Dimensions.
- The UNIX processes for the License Manager (unless you are running Dimensions with an evaluation license).
- The UNIX processes for the Common Tomcat, if you want to run Dimensions Web tools.

Proceed as follows to start the above mentioned components:

- To start the Serena-Supplied Runtime RD.
- To start the Oracle Enterprise database server, consult your DBA or vendor documentation.
- To start the Dimensions server (assuming the other components are already started):
  - **a** Log in as user `root`.
  - **b** Give yourself the Dimensions environment, as explained in "General Prerequisites" on page 132.
  - **c** Run the following command:

    ```
    $ dm_control cm_start
    ```
  - **d** Run the following command:

    ```
    $ ps -eaf | grep dm[pal]
    ```
  - **e** If the `dmappsrv.x` UNIX processes are not running, consult the Troubleshooting appendix in the *Installation Guide for UNIX*.
- To start the Common Tomcat service, see "Starting Tomcat" on page 27.

Stopping Dimensions for Windows

If the Dimensions server, database server (Serena-Supplied Runtime RDBMs, Oracle Enterprise, or SQL Server Enterprise), License Manager, and Common Tomcat are all on the same PC, then all the appropriate Windows services stop automatically when the PC is shut down. However, if you want to stop some of the above mentioned components without shutting down Windows, proceed as follows:

- To stop the Dimensions server:
  - **a** Log in as a user with local Windows administrative rights.
  - **b** Access Windows services by:

    `Start | Control Panel | Services`

    or

    `Start | Control Panel | Administrative Tools | Services`
  - **c** Right-click `Dimensions CM Listener Service`.
  - **d** Select Stop.
- To stop the Common Tomcat service, see "Stopping Tomcat" on page 29.
- To stop the Serena-Supplied Runtime RDBMS, see "Stopping the Serena-Supplied Runtime RDBMS" on page 189.
- To stop the other types of database server, consult your DBA or vendor documentation.

Stopping Dimensions for UNIX

Proceed as follows to stop the Dimensions server, database server (Serena-Supplied Runtime RDBMs or Oracle Enterprise), License Manager, and Common Tomcat:

- To stop the Dimensions server:
  - **a** Log in as user `root`.
  - **b** Give yourself the Dimensions environment, as explained in "General Prerequisites" on page 132.
  - **c** Run the following command:

    ```
    $ dm_control cm_stop
    ```
- To stop the Common Tomcat service, see "Stopping Tomcat" on page 29.
- To stop the Serena-Supplied Runtime RDBMS, see "Stopping the Serena-Supplied Runtime RDBMS" on page 189.
- To stop the Oracle Enterprise database server, consult your DBA or vendor documentation.

Restarting Dimensions for Windows

If the Dimensions server, database server (Serena-Supplied Runtime RDBMs, Oracle Enterprise, or SQL Server Enterprise), License Manager, and Common Tomcat are all on the same PC, then all the appropriate Windows services should restart automatically when

the PC is restarted. However, if you want to restart some of the above mentioned components without restarting Windows, proceed as follows:

- To restart the Serena-Supplied Runtime RDBMS, see "Restarting the Serena-Supplied Runtime RDBMS" on page 190.
- To restart the other types of database server, consult your DBA or vendor documentation.
- To restart the Dimensions server (assuming the other components are already started):
  - **a** Log in as a user with local Windows administrative rights.
  - **b** Access Windows services by:

    `Start | Control Panel | Services`

    or

    `Start | Control Panel | Administrative Tools | Services`
  - **c** Right-click `Dimensions CM Listener Service`.
  - **d** Select Restart.
  - **e** If Windows service still fails to restart, consult the Troubleshooting appendix in the *Installation Guide for Windows*.
- To restart the Common Tomcat service, see "Starting Tomcat" on page 27.

Restarting Dimensions for UNIX

Proceed as follows to restart the Dimensions server, database server (Serena-Supplied Runtime RDBMS or Oracle Enterprise), License Manager, and Common Tomcat:

- To restart the Dimensions server:
  - **a** Log in as user `root`.
  - **b** Give yourself the Dimensions environment, as explained in "General Prerequisites" on page 132.
  - **c** Run the following command:

    ```
    $ dm_control cm_restart
    ```
- To restart the Common Tomcat service, see "Starting Tomcat" on page 27.
- To restart the Serena-Supplied Runtime RDBMS, see "Restarting the Serena-Supplied Runtime RDBMS" on page 190.
- To restart the Oracle Enterprise database server, consult your DBA or vendor documentation.

Administering Dimensions Schemas

CRDB – Create Dimensions Base Database

dmdba Command (in connect mode to SYSTEM base database)

This command enables you to create a Dimensions base database.

NOTE When you create a new base database, the default grant rules for the Update Files from Project/Stream and Deliver Files into Project/Stream privileges include the "User holds any role on the product owning the object" rule. As a result, there is a security issue where certain users are able to download and upload files from any project in the product including those to which they should not have access. To correct this, you must remove the "User holds any role on the product owning the object" rule from the grant rules for "Update Files from Project/Stream" and "Deliver Files into Project/Stream" privileges.

The syntax is as follows:

```
crdb <newbasedb> /toolman=<username> /pass[word]=<pcms_sys_passwd>
   [[/basedon=<basedb> /product=<product-id>] |
      /import=<exportfile>]
   [/def_tab=<default_tablespace>
      /temptab=<temporaray_tablespace>]
   [/indextab=<index_tablespace>] [/installviews]
```

NOTE Either /basedon or /import needs to be specified. Their use is mutually exclusive.

where

`<newbasedb>`	The database name for the new base database (schema) that is to be created, e.g. `PROJ_A`. Note that the name of the new base database should be alphanumeric and that the first character *must* be alphabetic. For example, `PROJ1` is valid but *not* `1PROJ`. **NOTE** For Oracle systems the database name is `<NEWBASEDDB>` (uppercase) and a password of `<NEWBASEDB>` (uppercase) is also assigned to the new base database. For instructions on how to change the password, see "CPAS – Changing Passwords for Databases" on page 161.
`<username>`	The login-id of the user who acts as base database manager for this Dimensions base database.
`<pcms_sys_passwd>`	The password for the `PCMS_SYS` schema. For the Serena-supplied runtime, this is `PCMS_SYS` by default (if this default password has *not* been changed then the `/PASS[WORD]` parameter need not be specified).

<basedb>	The name of an existing base database from which to copy a process model during creation of the new base database. Note that the process model data is imported into the product $GENERIC of the new base database.
<product_id>	If /basedon is used, the name of a product defined in the existing base database given by <generic_db> whose process model is to be used to set up the Generic Process Model in the new Dimensions base database. If this is not specified, then the process model for the $GENERIC product is used by default.
<exportfile>	Specifies the process model export file that is to be imported when creating the new base database.
<default_tablespace>	Serena-*Supplied Runtime RDBMS or Oracle Enterprise only:* Specifies the optional default tablespace for this user.
<temporary_tablespace>	Serena-*Supplied Runtime RDBMS or Oracle Enterprise only:* Specifies the optional temporary tablespace for this user.
<index_tablespace>	Serena-*Supplied Runtime RDBMS or Oracle Enterprise only:* Specifies the optional index tablespace for this user.

/installviews	Install Published Views (Report Views) for the Report User that gets created.
	
	An example invocation using SQL Server:
	
	`PCMS_SYS> crdb fred /basedon=qlarius_cm /`
	`  toolman=dmsys /installviews`
	`Creating the new base database FRED. Please`
	`  wait...`
	`Base Database has been successfully created`
	`Installing report views in FRED@dim10,`
	`  please wait....`
	`Report views have been successfully`
	`  installed.`

Experienced Oracle Enterprise users may note that Dimensions DBA commands by default support the use of RDBMS tablespaces other than the SYSTEM tablespace when an Oracle user is created. By default, the PCMS_DATA tablespace is used.

IMPORTANT!

- In order to connect to the newly created database, the Dimensions service may require stopping and restarting.
- The length of each item of information regarding the new base database is limited, and any excess is truncated. Base database names and passwords are limited to 25 characters and logins IDs to 64 characters.

DLDB – Delete Dimensions Base Database

dmdba Command (in connect mode to SYSTEM base database)

This command enables you to delete a base database.

IMPORTANT! To be able to use this command, you must issue it while connected to *another* base database; that is, it is *not possible* to delete a base database while any Dimensions users are connected to it.

CAUTION! You (`dmsys`) must first check that the contents of the base database are no longer needed as the deletion operation cannot be 'undone' later on (except by resorting to a backup copy of the database files or a full export of the RDBMS database taken before deletion). The `DLDB` operation *must only* be run when no other users are accessing the base database to be deleted.

All information contained in the base database is deleted, and all the Dimensions users who were registered to use it are also dropped. But the product item libraries (which are operating system directories) associated with the base database are not deleted.

The syntax is as follows:

```
dldb <basedb>
   [/confirm=yes]
   [/droplogin]
```

where

`<basedb>`	Is the name of the base database to be deleted.
`/confirm=yes`	By default, `DLDB` prompts you to confirm the deletion of `<basedb>`. Specifying `/confirm=yes` disables this prompt, and `DLDB` begins immediately.
`/droplogin`	(SQL Server only). If there is an SQL Server login account with the same name as the Dimensions database that is being deleted, the login ID is also deleted.

NOTES

- DLDB deletes the base database and any database objects owned by that base database.
- DLDB can take several minutes to complete.
- If DLDB fails because users are still connected to the schema, you can—in an emergency—disconnect such users by shutting down the Dimensions CM Listener Service and then resubmit the command. To shut down the Dimensions CM Listener Service:
  - Confirm, first, that the "-dsn <base_db>@dsn" entry in the listener.dat file is pointing to the base database that you want to delete. The listener.dat file is located in the dfs subdirectory of the Dimensions CM root directory.
  - To shut down the Dimensions CM Listener service Windows:
    - Start | Control Panel | Administrative Tools Services
    - (Right-click) Dimensions CM Listener | Stop
  - To shut down the Dimensions CM Listener service on UNIX:
    - Log in as user root.
    - Change directory to $DM_ROOT/prog
  - Run the command ./dmshutdown

LSDB – List Dimensions Base Databases

dmdba Command (in connect mode to SYSTEM base database)

This command enables you to obtain a listing of all the Dimensions base databases registered in the RDBMS database.

The syntax is as follows:

```
lsdb /log=<reportName>
```

where

<reportName>	Is the name of the output file. If not specified, a temporary filename is used.

The following example demonstrates the output generated by this command:

```
==========================================
Details for Base database QLARIUS_CM
==========================================

Dimensions Base Database Registered Users                          Status
------------------------ ---------------------------------------- ----------

QLARIUS_CM               TED (Ted Lewis)                          [active]
QLARIUS_CM               PHIL (Phil Cheng)                        [active]
QLARIUS_CM               DON (Don Evens)                          [active]
QLARIUS_CM               PD (DP Yung)                             [disabled]
QLARIUS_CM               DP (DP Yung)                             [active]
QLARIUS_CM               PETA (Peta Miata)                        [active]
QLARIUS_CM               DMSYS (DMSYS)                            [active]
```

```
QLARIUS_CM                DAN (Dan Price)                          [active]
QLARIUS_CM                DANIEL (Daniel Vincent)                  [active]
QLARIUS_CM                DIETER (Dieter Milliar)                  [active]
QLARIUS_CM                DAWN (Dawn Vincent)                      [active]
QLARIUS_CM                BOB (Bob Marlow)                         [active]
QLARIUS_CM                JORDIE (Jordie Davis)                    [active]
QLARIUS_CM                PENNY (Penny Simpson)                    [active]
QLARIUS_CM                TIM (Tim Thomas)                         [active]


Dimensions Base Database Registered Products
------------------------ ------------------

QLARIUS_CM               $GENERIC
QLARIUS_CM               QLARIUS
```

EXPM – Exporting a Process Model to a File

dmdba Command (in connect mode to SYSTEM base database)

This command enables a process model, *defined for a particular product*, to be exported into an (operating system) export file.

The export file thus generated can then be specified as an input when creating a Dimensions base database, the process data in the export file being imported into the product `$GENERIC` of the newly created base database. See "System Administration" on page 149.

Refer to the description of the `CRDB` command (see "EXPM – Exporting a Process Model to a File" on page 159) for more detailed information.

To execute the `expm` command, you must either:

- First switch connection to the relevant base database once you have initially connected as `SYSTEM`, for example, to switch to the base database `mydatabase`:

  ```
  SYSTEM> connect mydatabase
  ```

  and then execute expm without the /basedb parameter at the `MYDATABASE>` prompt; or

- Remain connected as `SYSTEM` and execute `expm` with the `/basedb` parameter (which is no longer optional in that case).

The full syntax is as follows:

```
expm "<exportfilename>" [<productid>] [/basedb=<basedb>]
```

where:

<exportfilename>	Is the name of the file to contain the exported process model data defined for a particular product.
	
	On UNIX, if you want to specify a full pathname, for example, /tmp/test.exp, the <exportfilename> needs to be enclosed in double quotes (") if run from the database prompt, for example:
	
	`MYDATABASE> expm "/tmp/test.exp" qlarius`
	
	If you want to run a similar UNIX command directly from the operating system prompt, to ensure that the dmdba command gets parsed correctly, the <exportfilename> needs to be enclosed first in single quotes (') and then in double quotes ("), for example:
	
	`$ dmdba -s system/manager@dim10 expm "'/tmp/test.exp'" qlarius /basedb=qlarius_cm`

<productid>	Specifies the particular product from which the process model is to be exported.
<basedb>	Specifies the optional base database to export, else the current base database is used.

NOTE The new database only has a $GENERIC product containing the process model inherited from the particular product you based the export on. It does not have any other products until you subsequently create them using the Dimensions DNP command.

To clone the entire database content, you need to use RDBMS export/import utilities such as dm_backup/dm_restore for use with the Serena-Supplied RDBMS or Oracle EXP/IMP for use with Oracle Enterprise 10g or 11g.

SSEQ – Resetting the Sequence Value for a Current Base Database

NOTE You only need to run this command if requested by Support.

dmdba Command (in connect mode to SYSTEM base database)

This command enables you to reset the RDBMS sequence value for the currently connected base database. See "SSEQ – Resetting the Sequence Value for a Current Base Database" on page 160 to see how to switch from the initial SYSTEM base database to your desired base database.

The syntax is as follows:

```
sseq
```

CPAS – Changing Passwords for Databases

NOTE Only available, and displayed, when connected to a Serena-Supplied Runtime RDBMS or an Oracle Enterprise database.

dmdba Command (in connect mode to SYSTEM base database)

This command enables you to change the password associated with a Dimensions CM base database Oracle account.

The syntax is as follows:

```
cpas <basedb> /new_password=<new_password>
```

where

`<basedb>`	Is the name of the base database for which the password is to be changed.
`<new_password`	Specifies the new password.

SPAC– Reporting on Tablespace Usage

NOTE Only available, and displayed, when connected to a Serena-Supplied Runtime RDBMS or an Oracle Enterprise database.

dmdba Command (in connect mode to SYSTEM base database)

This command enables you to report on the space usage for Dimensions tablespaces.

The syntax is as follows:

```
spac [/list=<filename>] [/print]
```

where

`/list=<filename>`	Specifies an optional filename for the report. By default, the report generated is called `dm_space_usage.out`.
`/print`	Optionally specifies if the report is to be printed.

INCR – Increasing Database Space Available to a Tablespace

NOTE Only available, and displayed, when connected to a Serena-Supplied Runtime RDBMS or an Oracle Enterprise database.

dmdba Command (in connect mode to SYSTEM base database)

This command enables you to increase the space available for Dimensions tablespaces.

The syntax is as follows:

```
incr <tablespacename> /filename=filename
  /filesize=<size_in_MB>
```

where

`<tablespacename>`	Specifies the tablespace that is to be increased.
`/filename`	Specifies the name of the data file to be created or modified. If the file does not exist, it is created.
`/filesize`	Specifies the size (in MB) to be added.

STATISTICS – Manage Statistics for the Database

dmdba Command (in connect mode to SYSTEM base database)

This command enables you to manage statistics for a Dimensions CM database. See "STATISTICS – Manage Statistics for the Database" on page 162 to see how to switch from the initial `SYSTEM` base database to your desired base database.

Syntax:

```
statistics estimate | compute | delete | sample | instance | table
```

where

`estimate`	Estimates statistics for this database.
`compute`	Computes statistics for this database.
`delete`	Drops statistics on this database
`sample`	Generates sample statistics for this database.
`instance`	Estimate statistics for all databases in this instance.
`schema [verbose]`	Estimates statistics for the current schema using auto-sampling.
`table <table name>`	Generate statistics for the specified table in the current schema.

Examples:

```
statistics table TABLE_NAME [estimate|compute|...]
```

```
statistics schema [TABLE_NAME_FILTER [VERBOSE]]
```

INSV – Installing Report Views

dmdba Command (in connect mode to SYSTEM base database)

This command installs Dimensions Published Views into a nominated base database, after first checking that they are not already installed.

NOTE The Dimensions Published Views mechanism and the individual views available are described in the related document called the *Reports Guide*. In the remainder of this section they are referred to as 'report views' for short.

The syntax is as follows:

```
insv <basedb>
```

where

<basedb>	Is the name of the base database into which the report views are to be installed.

DELV – Deleting Report Views

dmdba Command (in connect mode to SYSTEM base database)

This command deletes report views from a nominated base database, after first checking that they are currently installed.

The syntax is as follows:

```
delv <basedb>
```

where

<basedb>	Is the name of the base database from which the report views are to be deleted.

GRTV – Granting Report Views to a Dimensions User

dmdba Command (in connect mode to SYSTEM base database)

This command grants report views access to nominated users to a nominated base database, after first checking that they are not currently granted such access.

The syntax is as follows:

```
grtv <basedb> <user>
```

where

<basedb>	Is the name of the base database to which the user is to be granted report views access.
<user>	Is the name of the database user to whom report views access is to be granted. When you create a new base database in Dimensions you get, by default, a <basedb>_rept report user. Subsequently, when you install views using INSV, you can then use GRTV to grant views to the database user <basedb>_rept, or any other database user that you have created manually. For Microsoft SQL Server Enterprise only, if you specify the special user PUBLIC, then access to the report views is granted to all Dimensions users.

REKV – Revoking Report Views from a Dimensions User

dmdba Command (in connect mode to SYSTEM base database)

This command revokes report views access from nominated users from a nominated base database, after first checking that they currently are granted such access.

The syntax is as follows:

```
rekv <basedb> <user>
```

where

<basedb>	Is the name of the base database from which the user's access to the report views are revoked.
<user>	Is the name of the database user from whom report views access is revoked. Microsoft SQL Server Enterprise only: If you specify the special user PUBLIC, then access to the report views is removed from all Dimensions users.

LSGV – Listing Dimensions Users Granted Report Views Access

dmdba Command (in connect mode to SYSTEM base database)

This command list all Dimensions users in a nominated base database to whom report views access has been granted, after first checking that report views are currently installed in the base database.

The syntax is as follows:

```
lsgv <basedb> <user>
```

where

<basedb>	Is the name of the base database from which to list users currently granted report views access.

Recreating and Resizing Indexes in a Specified Base Database

The `recreateIndexes.sql` script enables you to recreate and resize the indexes in a specified base database. By default all the indexes in a base database are recreated, but optional parameters allow you to recreate all the indexes on a specified table or just a single specified index. By default the recreated indexes are placed in the `PCMS_IDX` tablespace, but this can be overridden by an optional parameter.

The `recreateIndexes.sql` script is run under the `dmdba` utility, and is invoked by the command:

```
recreateIndexes <basedb> [<tablespaceName>]
```

or

```
recreateIndexes <basedb> [<tablespaceName>] [/t[able]=<tableName>]
```

or

```
recreateIndexes <basedb> [<tablespaceName>] [/i[ndex]=<indexName>]
```

where `<basedb>` is the schema whose index(es) are to be recreated and `<tablespaceName>` is the tablespace in which the indexes are created.

The `recreateIndexes.sql` script rebuilds and resizes:

- all the indexes in the specified schema, or
- if the `/table` qualifier is given, all the indexes on the table specified by `<tableName>`, or
- if the `/index` qualifier is given, the single index `<indexName>`.

NOTE When the `/table` or `/index` qualifier is used, there must be no spaces before or after the following `'='` sign.

MIGRATE– Migrate an MBCS Encoded Database to UTF-8

NOTE Only available, and displayed, when connected to a Serena-Supplied Runtime RDBMS or an Oracle Enterprise database.

dmdba Command (in connect mode to SYSTEM base database)

This command enables you to migrate a Serena-Supplied Runtime RDBMS or Oracle Enterprise MBCS encoded database to UTF-8, see "Migrating MBCS Data to UTF-8 Data" on page 565.

ITEMLIBMODE - Item Library Access Policy

Enforces an item library access policy base across a base database.

```
ITEMLIBMODE default|secure
```

where:

- `default`

  14.3.3 and earlier: item libraries are not required to have an associated credential set.

- `secure`

  14.3.3 and later: must be explicitly activated. Requires item libraries to have an associated credential set to access files.

See also, "Item Library Security" on page 60.

This command enables you to migrate a Serena-Supplied Runtime RDBMS or Oracle Enterprise MBCS encoded database to UTF-8, see "Migrating MBCS Data to UTF-8 Data" on page 565.

Configuring Dimensions Server Pooling

Starting and Configuring Dimensions Server Pooling

Standalone tools

The `dmlsnr` and `refreshpoolconfig` tools must normally (unless mentioned otherwise) be run by the following user accounts:

- `dmlsnr`: User `root` on UNIX and user `dmsys` on Windows.

- dmlsnr: User root on UNIX and user dmsys on Windows.
- refreshpoolconfig: User root or dmsys on UNIX and user dmsys on Windows.

Dimensions server pooling is automatically invoked upon Dimensions startup. These processes are also automatically stopped upon Dimensions shuts down. The server pooling is invoked with various default parameters in the file listener.dat in the dfs subdirectory of the Dimensions installation directory. In many circumstances, these default parameters are perfectly adequate for most users.

NOTE In Dimensions CM for z/OS, a certificate mechanism is used to connect back to Dimensions CM from the mainframe. These certificates are also managed in memory by the pooling feature on the server machine. Timeouts for these certificates (-cert_timeout and -cert_purge_timeout) can be set up in the listener.dat file as described later in this section).

On a Dimensions UNIX server, automatic server pooling is started and stopped as part of the standard running of the Dimensions server startup and shutdown scripts introduced at Dimensions release 2009 R1:

```
$DM_ROOT/prog/dm_control cm_start
$DM_ROOT/prog/dm_control cm_stop
```

or by the legacy scripts

```
$DM_ROOT/prog/dmstartup
$DM_ROOT/prog/dmshutdown
```

On a Dimensions Windows server, the automatic server pooling is started and stopped by starting and stopping the following Dimensions Windows service:

```
Dimensions CM Listener Service
```

If you want to manually start/stop server pooling or dynamically reconfigure the parameters with which it is run, then you need to invoke the dmlsnr or refreshpoolconfig tools:

- dmlsnr enables a number of Dimensions application servers to be pre-started prior to Dimensions clients connecting to a Dimensions server. This enables faster connection time for clients as the application servers are already initialized and connected to the Dimensions database. The pool manager server also creates extra application servers, if necessary, up to a configured maximum after which client requests are refused. The pool is also shrunk by destroying unallocated application servers as clients disconnect, depending on the configuration of the pool, as determined by the listener.dat file.

  An application server that is serving an idle client is de-allocated from that client after a configurable length of time and is returned back to the pool ready for allocation to other clients. Inactive clients are seamlessly reconnected when they try to access Dimensions.

  Once a client's application server is returned back to the pool, the client's state is still maintained by the Dimensions pool manager server for a period of time. If they remain inactive during this time, their state is discarded. When they are reconnected, they start with a new session.

- refreshpooconfig enables the pool configuration to be updated while Dimensions is still running, by dynamically updating the listener.dat file.

Manually invoking the dmlsnr or refreshpoolconfig commands is explained in "Invoking Dimensions Server Pooling" on page 138. To recap, the specific commands are

```
dmlsnr -param $DM_ROOT/dfs/listener.dat
```

and

```
refreshpoolconfig -host machine:port
```

If you are not using the default TCP port of 671 for the Dimensions CM Listener Service, you must specify the appropriate port number.

NOTE To display server pooling statistics, use either the `getpoolstats` command or the Administration Console. See "Displaying Connection Pooling Statistics" on page 170 for details.

The `listener.dat` file is a text file containing a series of lines of the following format:

```
-<param><white-space><param-value>
```

The parameters supported are as follows:

`-service`	Default `pcms_sdp` Specifies the port to listen to for client connections. You may, therefore, run more than one pool manage using different ports.
`-user`	Specifies the operating system account name to own every application process in the pool. For the Serena-Supplied Runtime RDBMS or Oracle Enterprise, this can be any arbitrary valid operating system account. For Microsoft SQL Server Enterprise, it must be the operating system account of the database owner. **NOTE** See the `-restricted` parameter concerning area/remote node authentication credentials with respect to running a server or agent in restricted mode.
`-dsn`	Default `intermediate@dmudb` Specifies the Dimensions base database name and the ODBC data source name (DSN) to use when starting up the initial applications servers at the time of pool manager startup.
`-min`	Default `5` Minimum number of application servers to shrink the pool to.
`-max`	Default `120` Maximum number of application servers to expand the pool to.
`-initial`	Default `5` Initial number of application servers to be created in the pool at startup time. These use the database connection parameter `-dsn` for identifying the base database to connect to.
`-free`	Default `20` The maximum number of application servers that may be left idle. When this limit is reached, the pool starts shrinking as necessary by destroying idle application servers.

-idle_timeout	Default 300 at installation Minimum: none, but 60 is recommended The amount of time, in seconds, that a client can be idle before it is disconnected.
-session_timeout	Default 86400 (24 hours) at installation Minimum 3600 (1 hour) The amount of time, in seconds, that a client's session state is maintained by the pool manager before being discarded. The application server and the Dimensions license token are then returned to the pool. **NOTE** This value overrides the session-timeout parameter used in web-inf\web.xml files.
-cert_timeout	Default 3600 seconds at installation. Maximum: none. In Dimensions CM for z/OS, a certificate mechanism is used to connect back to Dimensions CM from the mainframe. This parameter specifies the interval that must pass before the certificate expires.
-cert_purge_timeout	Default 3600 seconds at installation. Maximum: 3600 (i hour) In Dimensions CM for z/OS, a certificate mechanism is used to connect back to Dimensions CM from the mainframe. This parameter specifies how often the pool removes expired certificates.
-max_auth_attempts	Default 3 Minimum 0. Specifying zero disables the functioning of this parameter, allowing an unlimited number of attempts to be made to pass an authentication point. To assist customers working in an environment where verification of a user's identity is required as part of the process, Dimensions offers authentication facilities for "sensitive" changes to an object's lifecycle state and attributes, improved audit trail generation, and new reporting facilities. The -max_auth_attempts parameter specifies the maximum number of attempts that can be made to pass an authentication point. When the limit is reached, the user's session is terminated and they have to re-log in to the client.
-dont_use_proxy	External executables called from event triggers run as the Dimensions CM proxy user unless you specify -dont_use_proxy in the dmlsnr configuration file or the dmlsnr command line. Specifying -dont_use_proxy causes dmappsrv processes to run as the authenticated user instead of the Dimensions CM proxy user. External executables then run as the authenticated user. However, the dmappsrv is locked to the user session, which disables dmappsrv pooling.

`-restricted_mode`

Used to start a Dimensions server or agent in restricted mode, that is, as a user other than the default UNIX user `root` or a Windows local administrator (for example `dmsys`). To do this, you also have to set the `-user` parameter to the appropriate user.

NOTE When running a server or agent in restricted mode, area/remote node authentication credentials are *not* used— that is, in restricted mode files in a remote area are owned by the user running the `dmpool` process (by default `dmsys`), regardless of which user-id is set for the area or userid specified in Remote Node Authentication.

NOTE LDAP authentication is required for UNIX servers.

NOTE You must additionally ensure that the service that is specified by `-service` (which is `pcms_sdp` by default) utilizes a port number of `1025` or higher rather than the default of `671`. You do this as follows:

- On Windows systems:

  Edit the symbol `DM_SERVICE_PCMS_SDP_TCP` (see "Setting Dimensions Server and Client UNIX Symbols" on page 39) to specify the port number to be used. Note, that this port number must also be used on the server node.

- On UNIX systems:

  a Edit the `/etc/services` file to specify the port number to be used (for example, '`pcms_sdp<tab>1025/tcp`').

  b Delete the "local connect pipe" (typically '`/tmp/dimensions_local_connect`')—it is automatically recreated when the listener is restarted.

  c Change the ownership and/or permissions on `$DM_ROOT/prog/dmstartup` and `$DM_ROOT/prog/dmshutdown` so that they are executable by the user specified by the `-user` flag.

`-ssl`
`-ssl_password`

Used for Secure Sockets Layer (SSL) connections.

`-trace`
`-tracedir`

Commented out by default.

There may be occasions when you need to diagnose possible issues with the Dimensions CM Listener. Dimensions provides special initialization parameters for starting the listener in a mode that traces internal progress and status information to a log file for debugging purposes.

To enable tracing, uncomment and edit appropriately the following lines in the listener.dat file:

-trace

-tracedir <directory_name>

where `<directory_name>` is the name of a directory where the trace files are to be created.

Displaying Connection Pooling Statistics

Standalone Utility

```
getpoolstats [-host machine:port]
```

Administration Console

```
Database Management | Connection pooling statistics
```

This function displays connection pooling statistics.

If you are not using the default TCP port of 671 for the Dimensions CM Listener Service, you must specify the appropriate port number.

The statistics output by `getpoolstats` and the Administration Console are similar and include:

- General Information
  - Number of sessions
  - Number of active sessions
  - Number of application server processes
- Detailed Information
  - User ID
  - Client Node
  - DB Name
  - DB Connection
  - Application Name
  - Time Connected (in sec)
  - Time Allocated (in sec)
  - Session ID
  - Process ID

Connection Pooling Statistics Main Window

The Connection Pooling Statistics main window of the Administration Console enables you to view information about the current sessions in use with the Dimensions server.

Field or Button	Description
Home	Returns to the Administration Console main window.
Help	Displays this help.
Refresh	Manually refreshes the display of pooling statistics. The display refreshes automatically every five minutes.
Number of sessions	Total number of login sessions (active plus inactive). Inactive sessions entries are colored blue; active sessions are colored black.
Number of active sessions	The number of login sessions that are presently active (colored black).

Field or Button	Description
Number of server processes	The number of server processes that have been started and are currently in use by a session.
User ID	The user ID for the login session.
Client Node	The machine the user logged in from. For the web client or Administration Console, this is the machine that is running the web application server (Tomcat).
DB Name	The base database name for the login session.
DB Connection	The database connection for the login session.
Application Name	The client program that is being accessed by the user. For example, PCCLIENT represents the desktop client.
Time Connected (in sec)	The amount of time the user has been logged in using this session.
Time Allocated (in sec)	The total amount of time a server process has been allocated to this session.
Session ID	The unique ID for the session.
Process ID	The unique process ID of the server process allocated to this session. If this value is 0, then there is no server process allocated and the session is idle.

How to View Connection Pooling Statistics

Purpose

Follow this procedure to display information about which users are connected to the Dimensions CM server and what resources they are using. You can use this information to monitor the server pool and make adjustments as necessary in the configuration file, listener.dat. For example, you might adjust the minimum and maximum parameters in order to control the number of server processes in the pool.

To view pooling statistics:

Under Database Management, click the **Connection pooling statistics** link. The Connection Pooling Statistics Main Window appears.

Managing Remote Jobs

Administration Console

```
Distributed Development | Remote jobs
```

Remote job execution enables a client to initiate a remote operation to be run on a tertiary node on which a Dimensions listener is running. For details on how to set up remote jobs, see "Remote Job Execution Templates" in the *Developer's Reference* and "Executing Operations on Remote Nodes" on page 551.

The Administration Console Remote Jobs function enables you to list the jobs in the job queue and to view their logs, and to edit and delete jobs in the queue.

About Remote Jobs

Remote job execution enables you to invoke processing on a remote node on which a Dimensions CM Listener is running, and to integrate Dimensions with your own systems and tools. For further details, see "Remote Job Execution" in the *Developer's Reference*.

The Remote Jobs option in the Administration Console enables you to:

- List and view remote jobs.
- Change the status or description of remote jobs.
- Delete remote jobs.

Remote Jobs Main Window

The Remote Jobs main window consists of the following:

- **Menu area:** Displays the toolbar buttons to help you carry out various tasks. See "Remote Jobs Menu Area" on page 172.
- **Navigation area:** Enables you to view and select from a list of remote jobs. See "Remote Jobs Navigation Area" on page 172.
- **Content area:** Displays a list of remote jobs, or the details on a selected remote job. See "Remote Jobs Content Area" on page 173.

Remote Jobs Menu Area

The Remote Jobs tab enables you to view the details on remote jobs. You can also change the status of a job and change its description. You can also delete one or more remote jobs. See the table below for a description of the toolbar buttons that appear on the Remote Jobs tab:

Remote Jobs Navigation Area

Button	Function
	Changes the status of selected remote jobs, if at status 'SUBMITTED', using the Change Status of Remote Job dialog box.
	Changes the description of selected remote jobs, if at status 'SUBMITTED', using the Change Description of Remote Job dialog box.
	Deletes one or more selected jobs.

When you select the Remote Jobs tab, the navigation area lists all the remote jobs for the base database. Selecting the top-level icon displays all the jobs in the content area. Selecting a job in the navigation area displays its details in the content area.

Remote Jobs Content Area

What appears in the content area depends on what you have selected in the navigation area.

If you select...	Then the content area displays...
The top-level base database object	■ A list of remote jobs. ■ : Select one or more objects. ■ : Select all objects. ■ **Filter**: Accesses the Remote Job Filter dialog box. ■ : Display a summary of the objects in an HTML page (suitable for printing). ■ : Display a summary of the objects as comma-separated values in a text file.
A remote job object	A **General** section: displays general details about the remote job. Click the View/Show Logs button adjacent to **General** to display the logs for the job.

How to List and View Remote Jobs

Purpose Follow this procedure when you want to see the status and result of remote jobs or build commands that have been initiated on Dimensions tertiary network nodes, or to view their logs.

For a build command, the log contains a brief build summary and details on which targets were built and successfully preserved in Dimensions. It also shows any messages/ warnings/errors generated during build outputs collection.

To list remote jobs:

1 In the Remote Jobs main window, click the Remote Jobs tab.

To filter the list of remote jobs:

1 Click the **Filter** link at the top of the content area.

2 Complete the fields in the Remote Job Filter dialog box. To clear the filter criteria, click the **Reset Filter** button.

To view the log for a remote job:

1 Select the job in the navigation area.

2 In the content area, click the View Logs button.

How to Edit and Delete Remote Jobs

Purpose Follow this procedure when you want to change the status, result code, or description of remote jobs or build commands in the job queue, or to delete them from the job queue.

To change the status of a remote job:

1 Select the job(s) whose status you want to change in the content area.

NOTE You can only change the status if the current status is SUBMITTED.

2 Click the Change Status button.

3 In the Change Status of Remote Job dialog box, select SUCCEEDED or FAILED.

4 Enter a Result Code.

5 Click OK to confirm the change.

To change the description of a remote job:

1 Select the job(s) whose description you want to change in the content area.

NOTE You can only change the description if the current status of the job is SUBMITTED.

2 Click the Change Description button.

3 Enter a new Description.

4 Click OK to confirm the change.

To delete remote jobs:

1 Select the job(s) you want to delete in the content area.

2 Click the Delete button to delete the selected job(s).

3 Click Yes to confirm that you want to delete the job(s).

NOTE You can only delete a job if the status is SUCCEEDED or FAILED.

Administering Dimensions Users

Introduction

The Dimensions user administration utilities (apart from "Displaying Connection Pooling Statistics" on page 170) are operative with respect to a particular Dimensions base database, and their use is restricted to the Dimensions base database managers for the base database concerned or—for credential set administration—the user who created a particular credential set. They are accessed through one or more of:

- The Dimensions `dmcli` command-line interface.

The command names are used when referring generically to the associated utilities. The following utilities are provided.

- **UREG - Register Users**

dmcli Command	`UREG <qualifiers>`
Administration Console	`Users and Roles | User & group registration | Users | Users and Groups tab | New button`

- **XREG - Un-Register User**

dmcli Command	`XREG <qualifiers>`
Administration Console	`Users and Roles | User & group registration | Users | Users and Groups tab | Delete button`

- **UUA - Update User Attributes**

dmcli Command	`UUA <qualifiers>`
Administration Console	`Users and Roles | User & group registration | Users | Users and Groups tab | Manage Attributes button`

- **RREG - Reassign User Registration**

dmcli Command	`RREG <qualifiers>`
Administration Console	No equivalent function available.

- **CCS - Create Credential Set**

dmcli Command	`CCS <qualifiers>`
Administration Console	No equivalent function available.

- **DCS - Delete Credential Set**

dmcli Command	`DCS <qualifiers>`
Administration Console	No equivalent function available.

- **UCS- Update Credential Set**

dmcli Command	`UCS <qualifiers>`
Administration Console	No equivalent function available.

- **LCS - List Credential Sets for Current User**

dmcli Command	`LCS`
Administration Console	No equivalent function available.

- **Assign base database manager**

dmcli Command	No equivalent function available.
Administration Console	`Users and Roles | Privileged Assignments | Administration Privileges` (from drop down box; shows Product Level Privileges by default).

- **Displaying Connection Pooling Statistics**

Standalone Utility	`getpoolstats -host machine:port`
Administration Console	`Database Management Tool | Connection Pooling Statistics`

Invoking the various tools is described in:

- `dmcli` commands: "Invoking dmcli" on page 141.
- Administration Console: "Invoking the Administration Console" on page 142.
- `getpoolstats` standalone utility: "Invoking Dimensions Server Pooling" on page 138.

UREG – Register Users

dmcli Command

Registers a Dimensions CM user. A base database manager may only register users in base databases for which they have such privileges (see also "System Administration" on page 149).

The syntax is as follows:

```
UREG <UserId> -
   [/WORKSET=<project_spec>]
   [/[NO]PASSWORD_SAVE] -
   [/LOCALE=<locale>] -
   [/ATTRIBUTES=(site=<site>,group_id=<group_id>, -
   Dept=<dept>,full_name=<full_name>,phone=<Phone>, -
   <attribute_id>=<value>,email_addr=<email_addr>)]
   [/[NO]PROXY]
```

`<UserId>`	Is an existing operating system login id for the new Dimensions user, for example, `usera`. To register a user as a "proxy" user, specify * as the user-id. See "Types of Dimensions User" on page 125 for a discussion of "proxy" users. **Note:** When proxy log in is enabled, any user can access the database without registering.
`<project_spec>`	Is the specification (in the form `<product_id>:<project_name>`) of the user's initial default project/stream. If this is not specified, no initial default project/stream is defined for the user.
`<locale>`	Is the UNIX-style local string, for example, `en_US`.
`<site>`	Is the location or site of the new Dimensions user, for example, `St. Albans`. **Note:** You can specify as many or as few of the attributes—this and below—as you want.
`<group_id>`	Is the group_id of the new Dimensions user, for example, `Testing`.
`<dept>`	Is the department of the new Dimensions user, for example, `Finance`.
`<full_name>`	Is the full User name of the new Dimensions user, for example, `A. N. User`

`<Phone>`	Is the new Dimensions User's telephone number, for example, `01727 812812`
`<attribute_id>`	Is a pre-defined user-specified attribute and associated value, for example, `country_orig=UK` **NOTE** If you subsequently want to update a user's attributes, you can either do that through the Administration Console (see the *online help*) or through the `dmcli` "Update User Attributes" command `UUA` (see *Command-Line Reference*).
`<email_addr>`	Is a free text value for your email address. As the character `'@'` is the default `dmcli` escape character, you need to specify your email address in a form such as `'myemail@@mycomapny.com'`. All your Dimensions email is sent to this address.
`/PASSWORD_SAVE` or `/NOPASSWORD_Save`	Certain Dimensions client tools have a "remote automatic login" facility (for details, see the *online help*). 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 create a Dimensions user: ■ This facility is permitted by specifying the `/PASSWORD_SAVE` qualifier. Subsequently, the first time a client tool is invoked, the user has the option in the Login dialog box 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 are automatic unless the `CTRL` or `Shift` key is kept pressed 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 receive a message informing them that their stored remote login password is incorrect. ■ The Login dialog box is displayed, prompting them for their (new) remote login password. ■ After connection has been established, the new remote login password is stored as described above. ■ From then on, subsequent remote logins are again automatic.
`/[NO]PROXY`	■ /PROXY enables a user to login as a proxy user, which restricts the actions they can perform. ■ /NOPROXY upgrades a non-proxy user to a full user.

IMPORTANT! The length of each item of information about the new user is limited, and any excess is truncated. Base database names and passwords are limited to 25 characters and logins IDs to 64 characters.

An advisory email 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.

Administration Console

The function is accessed as follows:

```
Users and Roles | User & group registration | Users | New button
```

See the description of the UREG command and the *online help* for more information.

XREG – Unregister Users

dmcli Command

Users should be deleted (unregistered) whenever they no longer need to use Dimensions. The XREG command enables the base database managers for a base database to delete users from it.

The syntax is as follows:

```
XREG <UserId> [/[NO]KEEP]
```

where:

<UserId>	Is the login id of the of the Dimensions user to be deleted, for example, USER_A. To disable "proxy" user log in, specify * as the user-id. See "Types of Dimensions User" on page 125 for a discussion of "proxy" users. **Note:** After proxy login is disabled, no user can access the database without requiring to be registered.
/[NO]KEEP	Determines whether to keep or delete associated delegations and user role assignments.

CAUTION! For this function to run successfully, ensure that no other Dimensions or RDBMS activity affecting this user occurs while the function is running. The operation must also not be interrupted or terminated prematurely.

Before the XREG operation is started, the command checks if any users are currently connected to the database. The operation does not run if the user is active. However, the XREG operation does not prevent users attempting to connect while the operation is in progress.

Once the XREG operation has completed, an advisory email message is sent to the user.

Although XREG retains the corresponding user profile information, unless /KEEP is specified associated delegations and user role assignments (specified by the Dimensions Define User Roles (DUR) command or the Administration Console) are also deleted.

Upon completion of the delete operation, the deleted user's account profile information is retained within the base database, and such a user becomes *Dormant*.

Administration Console

The function is accessed as follows:

```
Users and Roles | User & group registration | Users | Delete button
```

See the description of the XREG command and the *online help* for more information.

Credential Sets Considerations

XREG checks the credential set table for any matches. If a match is found:

- A warning stating credential set records exist is issued to the user.
- All corresponding credential set records are deleted.
- Any area definitions referencing the above credential sets have the credential set uid field (AREA_CATALOGUE.AREA_CS_UID) reset as well as the user name field (AREA_CATALOGUE.AREA_USERNAME).

UUA– Update User Attributes

dmcli Command

This command is used to update the attributes for a Dimensions user.

The syntax is as follows:

```
UUA <UserId> [/ATTRIBUTES=(<attribute-id>=<value>,
    <attribute-id>=<value>,…)] [/[NO]PASSWORD_SAVE]
```

where

`<UserId>`	is the login id of the Dimensions user.
`<attribute-id>`	Specifies an attribute ID.
`<value>`	is the attribute value.

/[NO]PASSWORD_SAVE

Certain Dimensions client tools have a "remote automatic login" facility (for details, see the online help). 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 has the option in the Login dialog box 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 are automatic unless the CTRL or Shift key is kept pressed during client launch.
- 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 receive a message informing them that their stored remote login password is incorrect.
- The Login dialog box is displayed prompting them for their (new) remote login password.
- After connection has been established, the new remote login password is stored as described above.
- From then on, subsequent remote logins are again automatic.

The length of each item of information about the new user is limited, and any excess is truncated. Base database names and passwords are limited to 25 characters and logins IDs to 64 characters.

An advisory email 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.

For example, if user `peter` is registered as a Dimensions user and his site and passport number has changed, the command is:

```
UUA peter /ATTRIBUTES=(site=Michigan, passport_id=1243)
```

Administration Console

The function is accessed as follows:

```
Users and Roles | User & group registration | Users | Manage
    Attributes button
```

RREG – Reassign User Registration

dmcli Command

This command is used to change the operating system user account that is associated with a Dimensions user.

The syntax is as follows:

```
RREG <ExistingUserId> /USER=<NewUserId>
```

where

`<ExistingUserId>`	is the old operating system login id currently associated to a Dimensions user.
`<NewUserId>`	is the new operating system login to be associated to a Dimensions user.

For example, if user `peter` is registered as a Dimensions user and his operating system account name has changed to `peter_rd`, the command is:

```
RREG peter /USER=peter_rd
```

NOTES

- If you change the operating system user account for a database to which you are currently connected, we recommend that you quit your client immediately after running the `RREG` command.
- `RREG` completely renames the user account and so affects the change history. For example, audit trail records that refer to the old operating system account are updated to the newly specified account.

Administration Console

No equivalent function available.

CCS – Create Credential Set

dmcli Command

Creates a credential set with the specified name, user-id, password, and owner. Each user creates their own credential set.

The syntax is as follows:

```
CCS <credential-spec> -
   /USER =<userid> -
   /PASSWORD=<password>
   [/OWNER=<user or group>]
```

where:

`<credential-spec>`	Specifies the name of the credential set.
`<user-id>`	Specifies the operating system user id.
`<password>`	Specifies the operating system password for the specified user-id.
`<owner>`	Specifies a user or group that owns the credential set.

Administration Console

No equivalent function available.

Overview

Prior to Dimensions CM 10.1.2, each Dimensions CM work or deployment area required a set of credentials (operating system user id and password) stored against it, which are

then used for any operations performed against that area. When a password expires or changes with respect to a given user-id, the user had to search out the areas that use that set of credentials and update the credentials appropriately.

To facilitate managing credential sets, from Dimensions CM 10.1.2 onwards, user-id and password information is externalized into a user-owned table; this table can then be referenced by the various parts of Dimensions CM that require this type of information, for example, work or deployment areas. By externalizing this data into a user-owned table, the user can update a given set of credentials in one location and have the effects seen across the system.

Features

- The credential set name is unique to each user. There are no global credential sets.
- Credential set names are unique within each base database. A base database upgrade automatically detects and renames duplicate credential set names.
- Credential sets can be owned by an explicitly named user or group. Provided with the appropriate privileges, the owner of a credential set can update credential set details, delete the credential set, and associate it with an area or an item library.
- A user may only view, update, delete, or use credential sets that belongs to them.
- A user invokes a credential set by specifying the name wrapped in <> in any field that accepts a user-id. for example:

  ```
  <my-credential-set>
  ```

- AES128 encryption is used on all credential set passwords.

Related Commands

DCS - Delete Credential Set. See "DCS – Delete Credential Set" on page 183.

UCS - Update Credential Set. See "UCS – Update Credential Set" on page 183

LCS - List Credential Sets for Current User. See "LCS – List Credential Sets for Current User" on page 183.

XREG - Un-Register Users. See "XREG – Unregister Users" on page 178 and DUSR in *Command-Line Reference.*

AUTH - Authorize Access to Node. See *Command-Line Reference*.

CA - Create Area. See *Command-Line Reference*.

DUSR - Un-register Users. See *Command-Line Reference and* "XREG – Unregister Users" on page 178.

REXEC - Execute a Job on a Network Node. See *Command-Line Reference* and "Executing Operations on Remote Nodes" on page 551.

UA - Update Area. See *Command-Line Reference*.

DCS – Delete Credential Set

dmcli Command

This command deletes the specified credential set. Only the user who created a credential set can delete it.

The syntax is as follows:

```
DCS <credential-spec>
```

where:

`<credential-spec>`	Is the name of the credential set.

Administration Console

No equivalent function available.

UCS – Update Credential Set

dmcli Command

This command updates an existing credential set. Only the user who created a credential set can update it. If the credential set does not exist, it is added.

The syntax is as follows:

```
UCS <credential-spec> -
   /NEW_NAME=<new credential set name>
   /USER =<new user name>
   /PASSWORD=<new password>
   /OWNER=<new owner>
```

where:

`<credential-spec>`	Specifies the current name of the credential set.
`<new_name>`	Specifies the new name of the credential set.
<user-id>	Specifies a new user ID.
<password>	Specifies a new password.
<owner>	Specifies a new user or group that owns the credential set.

Administration Console

No equivalent function available.

LCS – List Credential Sets for Current User

dmcli Command

This command lists the current credential sets defined for the current user. Only the set name and user-id are displayed, not the password.

The syntax is as follows:

```
LCS
<credential set name filter>
/OWNER=<user or group>
```

where:

`<credential set name filter>`	Lists all credential sets that match a pattern (suitable for `SQL LIKE`).

/OWNER=<user or group>	Lists all credential sets owned by the specified user or group. Use '%' to list all credential sets.

The listing is similar to the following example:

Name	User ID
D3	joe-user
N1	juser
S1	joeuser

Administration Console: No equivalent function available.

Assigning General Administrative Privileges

dmcli Command: Not supported.

Administration Console:

```
Users and Roles | Privileges | Administration Privileges (select from
    the list)
```

This function enables an existing base database manager to add a Dimensions user to the list of base database managers for the base database (the base database concerned is the one that was selected when the Administration Console was first invoked).

Base database managers are also deleted (to be precise, demoted back to normal Dimensions users) by use of the Administration Console.

See the *online help* for details.

Backing Up Item Libraries

To guard against the possibility of hardware failure that may result in the RDBMS database or Dimensions libraries becoming unusable, it is vital that back ups take place on a regular basis. The information stored under Dimensions that should regularly be backed up comprises:

- Information stored within base databases.
- Product items stored within the Dimensions item libraries. The item libraries are operating system directories.

To preserve consistency between the database and the item libraries after recovery, it is necessary to take backups of both of these simultaneously: that is, export or back up all database files and their associated product item libraries at the same time. This ensures physical synchronization among all the components, representing a valid snapshot of the Dimensions data at a specific point in time.

To determine the nodes and directory names for each of the item libraries:

1 List the base database names with Dimensions `LSDB`.

2 Connect to the first base database <basedb> with the appropriate DSN connect string <connect_string> by typing

```
dmdba <basedb>@<connect_string>
```

See "Running the Dimensions DBA Tools" on page 133 regarding the dmdba command line interface.

3 Execute the following SQL commands with respect to the connected base database <basedb>

```
BASEDB> spool "<listing.out>"
BASEDB> select distinct node_name, library from obj_types where
    type_flag = 'I';
BASEDB> exit
```

Replace listing.out with your own filename in which to store the information displayed to the screen.

4 Repeat steps 2 and 3 for all the remaining base databases.

Configuring the dmlcasrv Server

The *dmlcasrv* server monitors all projects and streams for new change sets and sends them to their respective library caches using the DLCA command with a list of changes.

dmlcasrv_config.dat is the configuration file for *dmlcasrv* and is located in:

%DM_ROOT%\dfs\

The file contains descriptions of all the server options.

Encrypting Usernames and Passwords

To connect to a database or start a process as a specific operating system user, the Dimensions CM server requires the credentials of an operating system or database user account, typically a user ID and password. Credentials are stored in the registry.dat file in encrypted format.

By default, Dimensions CM uses AES128 encryption (cipher type aes-128-cbc). If you need a stronger encryption policy, Dimensions CM also supports the AES256 cipher (aes-256-cbc).

NOTE To use aes-256-cbc cipher type, the Java Runtime Environment that runs OpenText Common Tools must be patched with *JCE Unlimited Strength Jurisdiction Policy Files*.

To disable AES256 encryption and use only the AES128 cipher, add the following flag to the dm.cfg configuration file on the server and agents:

```
DM_ALLOW_AES_STRENGTH_UPGRADE N
```

To allow connections from clients and agents using generated logon certificates encrypted with the legacy Blowfish cipher, add the following flag to the `dm.cfg` configuration file on the server:

```
DM_ALLOW_LEGACY_LOGON_CERT Y
```

To change the default encryption type, add the flag `DM_CIPHER_TYPE` to the Dimensions CM `dm.cfg` configuration file on the server, and specify one of the supported types:

- `aes-256-cbc`
- `bf-ecb` (Blowfish)

If `registry.dat` was encrypted with a cipher that is different to the current one, Dimensions CM automatically:

- Re-encrypts `registry.dat` with the current cipher.
- Saves a copy of the previous version as `registry.dat.prev.`*n*.

Re-encryption occurs when:

- A Dimensions CM server initially accesses `registry.dat`, for example, after a restart or an upgrade.
- The default cipher is changed using the `dmpasswd` utility, and the server is not restarted.

After every re-encryption, *n* is incremented by 1, for example:

```
registry.dat.prev.1
registry.dat.prev.2
```

Chapter 13

Serena-Supplied Runtime RDBMS Database Administration

Chapter Overview	188
Disclaimer	188
Starting the Serena-Supplied Runtime RDBMS	188
Stopping the Serena-Supplied Runtime RDBMS	189
Restarting the Serena-Supplied Runtime RDBMS	190
Backing Up Serena-Supplied Runtime RDBMS Dimensions Databases	191
Restoring Serena-Supplied Runtime RDBMS Dimensions Databases	192
Reducing Database Fragmentation	195
Additional Serena-Supplied Runtime RDBMS Control Files	195
Use of Serena-Supplied Runtime RDBMS Tablespaces	196
Recovery After Media Failure	197

Chapter Overview

IMPORTANT!

- The utilities described in this chapter should only be run by a database administrator or a user with detailed technical knowledge.
- If you are using an Oracle Enterprise or Microsoft SQL Server Enterprise RDBMS, please refer to your DBA or vendor documentation concerning similar utilities to those described in this chapter for the Serena-Supplied Runtime RDBMS.

The starting, stopping, and restarting of Windows services and the use of various Serena-Supplied Runtime RDBMS database maintenance utilities enable:

- The starting of the Serena-Supplied Runtime RDBMS (see "Starting the Serena-Supplied Runtime RDBMS" on page 188).
- The stopping of the Serena-Supplied Runtime RDBMS (see "Stopping the Serena-Supplied Runtime RDBMS" on page 189).
- The restarting of the Serena-Supplied Runtime RDBMS (see "Restarting the Serena-Supplied Runtime RDBMS" on page 190).
- The backing up of Dimensions databases in the Serena-Supplied Runtime RDBMS to a backup file (see "Backing Up Serena-Supplied Runtime RDBMS Dimensions Databases" on page 191).
- The restoring of Dimensions databases into a Serena-Supplied Runtime RDBMS from a backup file created earlier (see "Restoring Serena-Supplied Runtime RDBMS Dimensions Databases" on page 192).

Disclaimer

Serena makes no warranty of any kind in regard to the contents of this chapter, including but not limited to implied warranties of merchantable quality or fitness for any particular purpose. Serena shall not be liable for errors contained in it or for incidental or consequential damages in connection with the furnishing, performance, or use of this chapter or associated software. The information in this chapter is subject to change without notice.

Starting the Serena-Supplied Runtime RDBMS

Starting the Serena-Supplied Runtime RDBMS for Windows

The appropriate Windows services for the Serena-Supplied Runtime RDBMS should start up automatically when the PC is started. However, if they fail to automatically start, proceed as follows:

1 Log in as a user with local Windows administrative rights.

2 Access Windows services by:

```
Start | Control Panel | Services
```

or

```
Start | Control Panel | Administrative Tools | Services
```

3 Right-click `Oracle<oracle_service_name>TNSListener` (by default, this is `OracleDimensionsTNSListener`).

4 Select Start.

5 Right-click `OracleService<oracle_service>` (by default, this is `OracleServiceDIM10`).

6 Select Start.

Starting the Serena-Supplied Runtime RDBMS for UNIX

Proceed as follows to start the Serena-Supplied Runtime RDBMS:

1 Log in as user `root`.

2 Give yourself the Dimensions environment, as explained in "General Prerequisites" on page 132.

3 Run the following command:

```
$ dm_control rdbms_start
```

4 Run the following command:

```
$ ps -eaf | grep ora
```

5 If various `oracle` UNIX processes are not running, consult the Troubleshooting appendix in the *Installation Guide for UNIX*.

Stopping the Serena-Supplied Runtime RDBMS

Stopping the Serena-Supplied Runtime RDBMS for Windows

The appropriate Windows services for the Serena-Supplied Runtime RDBMS should stop automatically when the PC is shut down. However, if you want to stop them without shutting down Windows, proceed as follows:

1 Log in as a user with local Windows administrative rights.

2 Access Windows services by:

```
Start | Control Panel | Services
```

or

```
Start | Control Panel | Administrative Tools | Services
```

3 Right-click `Oracle<oracle_service_name>TNSListener` (by default, this is `OracleDimensionsTNSListener`).

4 Select Stop.

5 Right-click `OracleService<oracle_service>` (by default, this is `OracleServiceDIM10`).

6 Select Stop.

Stopping the Serena-Supplied Runtime RDBMS for UNIX

Proceed as follows to stop the Serena-Supplied Runtime RDBMS:

1 Log in as user `root`.

2 Give yourself the Dimensions environment, as explained in "General Prerequisites" on page 132.

3 Run the following command:

```
$ dm_control rdbms_stop
```

Restarting the Serena-Supplied Runtime RDBMS

Restarting the Serena-Supplied Runtime RDBMS for Windows

The appropriate Windows services for the Serena-Supplied Runtime RDBMS should restart automatically when the PC is restarted. However, if you want to restart them for some reason without restarting Windows, proceed as follows:

1 Log in as a user with local Windows administrative rights.

2 Access Windows services by:

```
Start | Control Panel | Services
```

or

```
Start | Control Panel | Administrative Tools | Services
```

3 Right-click `Oracle<oracle_service_name>TNSListener` (by default, this is `OracleDimensionsTNSListener`).

4 Select Restart.

5 Right-click `OracleService<oracle_service>` (by default, this is `OracleServiceDIM10`).

6 Select Restart.

Restarting the Serena-Supplied Runtime RDBMS for UNIX

Proceed as follows to restart the Serena-Supplied Runtime RDBMS:

1. Log in as user `root`.
2. Give yourself the Dimensions environment, as explained in "General Prerequisites" on page 132.
3. Run the following command:

```
$ dm_control rdbms_restart
```

4. Run the following command:

```
$ ps -eaf | grep ora
```

5. If various `oracle` UNIX processes are not running, consult the Troubleshooting appendix in the *Installation Guide for UNIX*.

Backing Up Serena-Supplied Runtime RDBMS Dimensions Databases

Overview

The Dimensions CM `dm_backup` script enables you to back up (for archiving or restoration using the `dm_restore` script):

- Specific Serena-Supplied Runtime RDBMS database schemas.
- The full database instance.

NOTE

- All CM servers accessing the database must be shutdown for the duration of the backup.
- The host Oracle must be running.

Before running the backup script check that no character set conversion occurs during database export:

1. Check the current database character set:

```
$ sqlplus system/XXXXX@DIM14
SQL> select * from nls_database_parameters where parameter =
    'NLS_CHARACTERSET';

PARAMETER             VALUE
--------------------  ------------------
NLS_CHARACTERSET      AL32UTF8
```

2. Check the current (Oracle) client environment:

```
$ echo $NLS_LANG
ENGLISH_UNITED KINGDOM.AL32UTF8
```

If the NLS_LANG setting is wrong and does not match the database character set you must set it to the correct value.

3 When the script runs the output or log file should contain a line, similar to the following, that references the database character set:

```
Export done in AL32UTF8 character set and AL16UTF16 NCHAR character
    set
```

The dm_backup Syntax

The dm_backup syntax is as follows:

```
dm_backup /u <schemaName> /f <expFile> [/a] [/l <logFile>]
```

For example:

```
dm_backup /u system/<password>@dim10 /f export.exp /a
```

where

`/u <schemaName>`	Specifies the full database connection details to access the schema you want to backup: ■ If you want to backup a specific schema, then you must specify that schema. ■ If you are backing up your entire Dimensions CM instance, then you must specify the SYSTEM account details.
`/f <expFile>`	Specifies the name of the backup file to be created.
`/a`	If specified, results in all schemas being backed up. This is only meaningful when performing a SYSTEM backup.
`/l <logFile>`	If specified, logs operating system messages output by `dm_backup` to the file <logFile>.

Restoring Serena-Supplied Runtime RDBMS Dimensions Databases

Overview

The Dimensions CM `dm_restore` script enables the DBA to restore from a backup file (created from the complementary `dm_backup` script) either:

- Selected specific Serena-Supplied Runtime RDBMS database schemas.
- The full database instance itself.

CAUTION!

- The `dm_restore` script reinitializes the database that is being restored. Check that no one is accessing the databases before running the script. Otherwise, the restoration process may fail.
- The restoration process deletes the database schemas that you are trying to restore. It is essential that you have a trusted database instance restore point that you can go back to if an issue occurs during the restoration process. Otherwise, you may lose data.

The dm_restore Syntax

The dm_restore syntax is as follows:

```
dm_restore <systemDatabase@DSN> /u <schemaName> /f <expFile> [/a]
    [/l <logFile>] [/z]
```

For example:

```
dm_restore system/<password>@dim10 /f export.exp /a
```

where

<systemDatabase@DSN>	Specifies the full connection details of the `SYSTEM` account you are wanting to restore databases for.
`/u <schemaName>`	Specifies the Dimensions CM database schema that you want to restore: ■ If you want to restore a specific schema, then you must specify that schema. ■ If you want to restore your entire Dimensions CM instance, then you must specify `SYSTEM`. **NOTE** See the additional notes below.
`/f <expFile>`	Specifies the name of the specific backup file (created by the `dm_backup` script) to be used for restoration.
`/a`	If specified, results in all schemas being restored. This is only meaningful when performing a SYSTEM restoration.
`/l <logFile>`	If specified, logs operating system messages output by `dm_restore` to the file <logFile>.
`/z`	If specified, forces restoration to occur. No confirmation is required.

NOTES

- If you are restoring a specific Dimensions CM database schema, that schema must already exist in the database instance that you are restoring the schema into.
- If you are restoring your entire database instance, the restoration process only restores the database schemas that exist both in your backup file and your recipient instance. If this condition is not satisfied, then you either need to:
  - restore specific database schemas one at a time, or
  - use the DMDBA `crdb` utility to create empty databases first.

  Failure to do this may result in the restoration process failing to restore the database you were expecting.

Preparing the Restored Database Schemas for Use

Restoring Database Schema Grants

If you restored database schemas by repeatedly restoring single database schemas (rather than restoring the entire database instance by specifying `SYSTEM`), then for each specific schema restored, you need to restore the appropriate grants to use the database schema.

To do this, run the following commands for each schema:

```
dmdba pcms_sys/<pcms_sys_password>@<dnsName> grant_pcms_sys
    <DimensionsDatabaseName>
dmdba system/<system_password>@<dnsName> cpas
    <DimensionsDatabaseName> /new=<DimensionsDatabaseName>
```

Once the grant commands have been run for all the restored schemas, you then need to shut down and restart the Dimensions CM installation.

Reinstalling Published Views into the Restored Schemas

If you deleted published (report) views prior to creating a backup (which is a prerequisite if such views existed), then you now need to re-install and re-grant these views to the report users. For *each database*:

1. Log in to `dmdba` as the Dimensions CM RDBMS Administrator (for the Serena-Supplied Runtime RDBMS this is `system`), see "Running the Dimensions DBA Tools" on page 133.
2. Enter the following command for database schema `<basedb>`:

   ```
   insv <basedb>
   ```

   (see "INSV – Installing Report Views" on page 162).
3. Enter the following command for report user `<basedb_report_username>` in database schema `<basedb>`:

   ```
   grtv <basedb> <basedb_report_username>
   ```

   (see "GRTV – Granting Report Views to a Dimensions User" on page 163).

The following message should return:

```
Report views have been successfully granted.
```

The initial invocation of `grtv` sometimes results in an error stream starting with:

```
SQL-1E36-40(00B0FE60) ORA-00955: name is already used by an existing object
```

You can safely ignore these errors.

4 Repeat Step 3 on page 194 for each report user in database schema <basedb>.

5 Repeat Step 1 on page 194 to Step 4 on page 195 for each restored database schema.

Reducing Database Fragmentation

The Serena-Supplied Runtime RDBMS database may over time become increasingly fragmented as Dimensions users are created and dropped and Dimensions operations are performed. This fragmentation may begin to have a harmful effect on the performance of Dimensions if left unchecked. The DBA can check on the amount of fragmentation that is present in the database by running the `dmdba SPAC` report (see "SPAC– Reporting on Tablespace Usage" on page 161). This report details the number of fragments that are present in a database. If the number under the column "Number of Fragments" reads at or above 50 then the DBA should consider performing a full SYSTEM dm_restore.

Additional Serena-Supplied Runtime RDBMS Control Files

Every Serena-Supplied Runtime RDBMS instance uses a small binary file called a *Control File*. This file contains the information required to startup the Serena-Supplied Runtime RDBMS Instance. This file is resident:

- UNIX

  In the directory `$ORACLE_HOME/oradata/<ora_sid>` – it is given the name `ctl1pcms.ora`.

- Windows

  In the directory `%ORACLE_HOME%\oradata\<ORA_SID>` – it is given the name `CTL1PCMS.ORA`.

We recommend that you keep multiple copies of this file on separate disks to reduce the risk of a media failure removing the control files. The UNIX and Windows installations of the Serena-Supplied Runtime RDBMS, however, create only one. To create another Control File (second one for UNIX or Windows) and/or to relocate the existing or created additional copies of the Control File (UNIX or Windows) use the following procedure:

1 Shut the database down.

  See "Stopping the Serena-Supplied Runtime RDBMS" on page 189.

2 Copy the Control File to another disk.

- For UNIX

  Modify the file `init<ora_sid>.ora` by *adding* or *modifying* the parameter `CONTROL_FILES` to include the name of the new Control File. A symbolic link to the file `init<ora_sid>.ora` is located in the `$ORACLE_HOME/dbs` directory (the physical file is located in the `$ORACLE_HOME/admin/<ora_sid>/pfile` directory).

  For example, suppose that the Oracle SID is dim10, the existing Control File is:

  ```
  /opt/oracle/10.2.0.2/oradata/dim10/ctl1pcms.ora
  ```

  and this file is copied into a new file called:

  ```
  /disk2/oracle/data/cntrl2pcms.ora
  ```

  residing on a different disk.

  To take into account the new Control File, add the following line to the file `initdim10.ora`:

  ```
  control_files =
  (/opt/oracle/10.2.0.2/oradata/dim10/ctl1pcms.ora,
  /disk2/oracle/data/cntrl2pcms.ora)
  ```

- For Windows:

  Modify the file `init.ora` by *adding* or *modifying* the parameter `control_files` to include the name of the new Control File. The file `init.ora` is located in the `%ORACLE_HOME%\srvm\admin` directory.

  The editing process in analogous to that described for UNIX.

3 Restart the database.

See "Restarting the Serena-Supplied Runtime RDBMS" on page 190.

Use of Serena-Supplied Runtime RDBMS Tablespaces

The Dimensions CM database administration functions and installation script allow the use of database tablespaces other than the `SYSTEM` tablespace. This has many advantages, for example:

- Separation of database control information from the Dimensions CM data.
- More effective backup strategies.
- Allocation of database files on different devices to improve performance.
- Minimization of database fragmentation.

When Dimensions CM is installed, tablespaces for the following are created:

SYSTEM	PCMS_TEMP
PCMS_RBS	PCMS_DATA
PCMS_IDX	USERS
SYSAUX01	UNDOTBS01

Unless non-default directories were specified during the installation, these tablespaces are placed in the following directories:

Oracle Type	Directory
UNIX	$ORACLE_HOME/oradata/<ora_sid>
Windows	%ORACLE_HOME%\oradata\<ORA_SID>

The SYSTEM tablespace is used to hold the data dictionary and the tables for the PCMS_SYS database user. The PCMS_RBS tablespace is used for rollback information for all database users.

Initially, all the Dimensions CM users are set up with their database tables in the PCMS_DATA tablespace, and with PCMS_TEMP as their temporary tablespace. A major advantage of using a separate temporary tablespace is that this prevents large queries and similar operations from fragmenting the space used for allocation to tables.

We recommend that all the Dimensions CM users (except for the special PCMS_SYS user) are set up to use non-SYSTEM tablespaces both as their default tablespace and as their temporary tablespace. There are many advantages to be had from using separate tablespaces for different base databases (particularly when these are large).

Recovery After Media Failure

IMPORTANT! The Dimensions CM administrator is strongly advised, in association with the Serena-Supplied Runtime RDBMS DBA, to back up your CM database and item libraries on a regular basis (preferably daily). You must ensure that the database and the item libraries are backed up as a consistent set (see "Backing Up Item Libraries" on page 184), representing a valid snapshot of your data at a specific point in time. In the unlikely event of a database failure, or disk failure, where a full recovery to a consistent Dimensions CM state is not possible, contact Support to resolve database problems after you have restored all relevant data files from your latest backup. In accordance with the company's standard maintenance terms, the company reserves the right to charge for such assistance and do so if a comprehensive backup is not available, is inconsistent, or is considered out of date.

Disk hardware failure can cause the RDBMS database or Dimensions CM libraries to become unusable. Recovery from this situation depends on the frequency and type of backups, and the type of failure. The Serena-Supplied Runtime RDBMS DBA should determine precisely what backups are being taken and with what frequency. There should be a tried and tested recovery plan in place before the media failure occurs. The plan should identify that there is always a possibility of loss of data, which requires users to re-input their work. The maximum time length of the loss should be quantified as a risk. It is important that your management and Dimensions CM Product Managers are made aware and accept the risk *before* the media failure occurs.

The extent of media failure needs to be accurately determined so that recovery is planned to achieve minimum data loss. As an extra precaution you can backup the disks immediately before any restore operations are started. A minor media failure can develop into a major loss of data if the wrong restore is carried out or if there is a second media failure.

It is important to recover to a consistent set of the RDBMS database and the Dimensions CM item libraries:

- If the database and libraries are co-located on the same disk, restoring a complete disk image results in a consistent rollback. However, all work since the last backup has to be re-input.
- If more than one disk is used the situation is more complex. Individual files may require restoring to achieve consistency and re-input of work may be required.

CAUTION! If database or media recovery is required, contact Support before proceeding.

Part 3

Data Migration

Introducing Data Migration	201
Dimensions Migration Standalone Utilities	205
Data Interchange File Format (PDIFF)	215
Using PDIFF to Import/Export Data	235
PDIFF Syntax	243

Chapter 14

Introducing Data Migration

Migrating Operating System or Existing Configuration Management Data	202
Web Client Data Migration Facility	202
Desktop Client Data Migration Facility	202
Exporting a User's Settings to Another Desktop Client	203
Dimensions dmcli Commands	203
Standalone Utilities	203
Data Interchange File Format (PDIFF) Import/Export Utility	204

Migrating Operating System or Existing Configuration Management Data

Dimensions® CM addresses the issues of uploading or downloading operating system file data to or from the Dimensions CM repository or of migrating existing Dimensions CM configuration management data to the Dimensions CM repository, as described below.

Web Client Data Migration Facility

The Dimensions CM web client provide a one-time or incremental migration (deliver) of a list of specified files in a specified operating system directory structure into revisioned files (items) under Dimensions CM control (see the online help for details). This facility is available to all users who have the privilege to create new items or revise existing item revisions in Dimensions CM.

NOTE Downloading/updating of data from Dimensions CM to an operating system directory structure of files is not available from the Dimensions CM web client.

Desktop Client Data Migration Facility

The Dimensions CM desktop client provides (*Windows only*) a one-time or incremental:

- Migration (deliver) of a list of specified files in a specified operating system directory structure into revisioned files (items) under Dimensions CM control.
- Migration (update) of revisioned files (items) under Dimensions CM control into a specified operating system directory structure.

IMPORTANT! Synchronization is not available for Dimensions CM streams.

- Synchronization of updated specified files in a specified operating system directory structure with respect to revisioned files (items) under Dimensions CM control or vice versa.

The Dimensions CM wizard provides these facilities. It can be invoked either:

- Directly from the Dimensions CM desktop client:
  - Right-click a Dimensions CM project folder | Update
  - Right-click a Dimensions CM project folder | Deliver
  - Right-click a Dimensions CM project folder | Synchronize
- From a Windows explorer window (provided the Dimensions CM explorer plugin has been installed):
  - Right-click a Dimensions Synchronize Wizard configured Windows folder | Dimensions | Synchronize

- Right-click a Dimensions Synchronize Wizard configured Windows folder | Dimensions | Update
- Right-click a Dimensions Synchronize Wizard configured Windows folder | Dimensions | Deliver

Before the wizard can be used, various configuration set up steps have to be performed. These configuration steps and a detailed description of the operation of the Dimensions Synchronize Wizard are documented in the Dimensions CM online help.

This facility is available to all users who have the privilege to create new items or revise existing item revisions in Dimensions CM.

Exporting a User's Settings to Another Desktop Client

To migrate settings from one desktop client to another, the user must edit the registry using regedit.

To export the keys, run the following command:

```
regedit /e Dimensions.reg
    "HKEY_CURRENT_USER\Software\Serena\Dimensions\14.6\PC Client"
```

To import the keys, run this command:

```
regedit /s Dimensions.reg
```

Dimensions dmcli Commands

Dimensions CM provides the `dmcli` commands:

- `upload`
- `download`

These commands provide a superset of all the facilities formerly provided by the legacy standalone utilities described in the next section, without the need to install the executables provided by Dimensions Make. We strongly recommend to use the `upload` and `download` `dmcli` commands in preference to the legacy standalone utilities of the same name. The legacy standalone utilities continue to be supported for backward compatibility until further notice, but they will eventually be "sunsetted".

Standalone Utilities

The standalone utilities (see Chapter 15, "Dimensions Migration Standalone Utilities") comprise:

- **Download** (see "Dimensions Download" on page 208)

  A one-time or incremental download of revisioned items from Dimensions control to a target operating system directory. This is equivalent to a Dimensions `get` operation

with no item header substitution. This facility is available to all users who have the privilege to get items from Dimensions.

- **Upload** (see "Dimensions Upload" on page 210)

  A one-time or incremental upload of a list of specified files in a specified operating system directory structure into revisioned files (items) under Dimensions control.

- **PRCS** (see "RCS-Like Front End to Dimensions: prcs" on page 212)

  A RCS-like front end to the version control commands of Dimensions CM.

- **PSCCS** (see "SCCS-Like Front End to Dimensions: psccs" on page 213)

  A SCCS-like front end to the version control commands of Dimensions CM.

NOTE The standalone functions are all available on the UNIX or Windows versions of Dimensions, with the exception of `prcs` and `psccs` which are only supported under UNIX.

Data Interchange File Format (PDIFF) Import/Export Utility

The PDIFF format and `pdiff` Tool (see Chapter 16, "Data Interchange File Format (PDIFF)" and Chapter 17, "Using PDIFF to Import/Export Data") are used to import/export the following Dimensions CM data to and from Dimensions CM:

- Issues
- Design Parts
- Certain process model data, namely:
  - Valid sets.
  - Delegation candidates.
  - User role assignments.
  - Change Document browse templates.

Chapter 15

Dimensions Migration Standalone Utilities

Foreword	206
General Information	207
Dimensions Download	208
Dimensions Upload	210
RCS-Like Front End to Dimensions: prcs	212
SCCS-Like Front End to Dimensions: psccs	213

Foreword

CAUTION! If you have enabled Change Management (CM) rules, this option must be disabled while performing data migration utilizing the standalone `upload` utility (as that utility is unable to specify a request to which to relate the uploaded data). Otherwise, errors may occur. See the *online help* for details about CM rules.

The `dmcli upload` command has a `/CHANGE_DOC_IDS` qualifier, so it is not affected by this limitation. These `dmcli` commands provide a superset of all the facilities formerly provided by the legacy standalone utilities described here, without the need to install the executables provided by Dimensions Make. We strongly recommend that you use the `upload` and `download dmcli` commands instead of the standalone utilities of the same name. The standalone utilities continue to be supported for backward compatibility until further notice, but they will eventually be "sunsetted".

This section describes the Dimensions CM standalone utilities used for data migration. Some of these utilities are intended for use by change-managers, product-managers, or part-managers only. Some utilities are available only for certain operating systems.

- `download` This utility performs a download of a list of specified files under Dimensions into a target directory. The new `download dmcli` command offers a superset of all the features found in the standalone `download` utility and should be used in preference. See the *Command-Line Reference* for details.
- `upload` This utility performs an upload of a list of specified files in a user directory into Dimensions. The new `upload dmcli` command offers a superset of all the features found in the standalone `upload` utility and should be used in preference. See the *Command-Line Reference* for details.
- `prcs` *[UNIX servers clients, and agents only]* This utility provides a RCS-like front end to the version control commands of Dimensions.
- `psccs` *[UNIX servers clients, and agents only]* This utility provides an SCCS-like front end to the version control commands of Dimensions.
- `pdiff` This utility is discussed in Chapter 17, "Using PDIFF to Import/Export Data" following an introduction in Chapter 16, "Data Interchange File Format (PDIFF)".

Legal considerations with respect to standalone download and upload

The `download` and `upload` standalone utilities require that various Dimensions Make and ADG executables and associated libraries files be installed. Some of these executables and associated libraries, namely:

File	UNIX and Windows	UNIX Only	Windows Only
adg	?		
dm_make	?		
dm_nmake			?
libmcx.so		?	
mcx.dll			?

are derived from source code covered by the GNU GENERAL PUBLIC LICENSE and the GNU LIBRARY GENERAL PUBLIC LICENSE.

These files, the *GNU Make User Guide*, and the *Dimensions CM Make Guide* are available from a download area on the Support website (see details below). To have a working Dimensions Make and ADG system (which also enables you to use the legacy `download` and `upload` standalone utilities), you first need to download the executables, libraries, "mini" installation scripts, and documentation from the Support page.

NOTE For conciseness, in the remainder of this manual the files referred to above are simply referred to as Dimensions Make.

As a condition of the GNU GENERAL PUBLIC LICENSE and the GNU LIBRARY GENERAL PUBLIC LICENSE, source code for the above discussed executable and library files is also available:

- The source code of the `make client` executable is a derivative of the source code of GNU Make, and is covered by the GNU GENERAL PUBLIC LICENSE (Version 2, June 1991).
- The source code of the `MCX` library, which is used for communication between `dm_make` and the slave process, is covered by the GNU LIBRARY GENERAL PUBLIC LICENSE (Version 2, June 1991).

To download Dimensions Make, contact Support.

Dimensions Make requires a Dimensions CM client or server installation. For installation instructions for Dimensions Make, see the *Installation Guide for Windows* and *Installation Guide for UNIX*.

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 login, which sets up the environment required for all Dimensions CM processing. The utilities reside in the directory specified by the environment variable `DM_PROG`, which the standard login includes 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

Lowercase letters can be included in the values of parameters and are automatically interpreted as the equivalent in uppercase. This applies to all parameters, except those that are specifically stated to be case-sensitive.

Wildcard 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 wildcard % may be used.

In other parameters a *null string* can be used to imply all possible values. A null string is specified as "" (two double-quote characters). This applies only to parameters for which it is stated that a null string may be used.

Execution Authority

If the parameters are specified so that a utility is required to process the issues of several different products in the same execution, then the user must either have Dimensions CM request privilege for every product concerned, or else have general administration privileges for the database.

Dimensions Download

CAUTION! The legacy standalone `download` utility does not support Dimensions streams. It fails ungracefully if you attempt to use it with streams.

This utility is available to all users who have the privileges to get items from Dimensions. It performs a download of a list of specified files under Dimensions into a user directory. A download of a Dimensions file is equivalent to a `get` operation with no item header substitution, that is, `FI /NOEXPAND`.

Syntax

```
download [ --help ] [ --version ] [ -n [ -s script-file ] ] [ -r ]
   [ -C start-dir ] [ -W project-spec | -B baseline-spec ]
   [ -D root-dir ] [ -f rule-file ] [ -p database-specifier]
   [-l log-file] [-t] [ -N ] [ -E ] [ input-file ] [-v] [-S]
```

<input-file>, -, -r

`download` performs a download of the list of files held in Dimensions specified in `<input-file>`. If no `<input-file>` is specified, `download` searches the project default directory unless "–" is specified, in which case it reads a list of filenames from the standard input. If the `-r` option is specified, `download` performs a recursive search of the project working location corresponding to the current working location for items to download.

IMPORTANT! To ensure correct operation of `download`, files specified in `<input-file>` must be separated with `<line-feed><carriage-return>`, this includes the last file in the list. If this is not done, you may under certain circumstances get a 'Path too long' error message.

Pcmsfile, -f

If no `<rule-file>` is specified, `download` looks for `Pcmsfile` in the current working location and the project default location. `download` restricts searches to the specified project. It does not preserve anything (as the whole point is to export items from Dimensions). As a result, the only sensible rule directive to put in a download rule file is `nofetch`. This directive instructs download to ignore the matching files. The syntax for these directives is exactly the same as that used for Dimensions Make—please refer to the *Dimensions CM Make Guide* for details.

Example

To download the entire contents of a project except the `'.doc'` file and the contents of the project location `'test_data'`, use the command:

```
download -W "PROD:PROJECT" -D /your/root -r -f your-rules
```

The file 'your-rules' contains:

```
test_data/% %.doc:
  nofetch
```

-W, -D — The `-W` option specifies the project from which to download the files, and the `-D` location option maps the project (or baseline) root to the location specified.

If no `-W` option is specified, the current project is used, as determined from the current working location. If no `-D` option is specified, the current working location is used as the project default location.

If a -D option is specified it must be an ancestor of the starting location, see -C option.

NOTE If you do not use the -D option, download assumes the default project root for your default project (if -B is specified) or the project specified with -W (default project if -W is not specified). If you start a download from a location that is not a proper descendant of that location and do not use the -C option, as explained above, the paths are not mapped correctly.

-n, -s — Specifying `-n` causes a command file to be generated, rather than the file being directly downloaded. The name of the script can be specified using the `-s` option. If no name is specified, the name `download.cmd` is used and the file is placed in the current working location.

CAUTION! In Windows, `".cmd"` is reserved as the filename extension for an executable. You must make sure that the specified filename does not end in `".cmd"`.

-p — Enables you to specify the database from which the files are to be downloaded. If not specified, the current database is used.

--version — Enables you to check the version of the download you are running. It merely displays the version number and then exits.

--help — Prints the syntax usage.

-l — Enables you to specify a file to contain error messages from the download.

-B — Specifies that downloads should be made from a specified baseline rather than a project.

-N — Causes dual-mode length and checksum matching to be disabled. The dual-mode feature matches files using the length and checksum of both the binary and text interpretation of a file where the native text file format is not the same as UNIX text file format.

-E — Causes files to be expanded when they are downloaded. The default is to download files in unexpanded form.

-C — Enables you to specify the starting location for the download process by `-C start-dir`. Any files created by the download are created in this location. If not specified, the current directory is used as the starting location. This is useful only if the location is the one specified by the -D option, or a sublocation thereof.

-t — Specifies that the system date/time shall be applied to the gotten file.

-v — Specifies verbose mode.

-S — Suppresses all non-fatal messages from `stdout`.

Dimensions Upload

CAUTION! The legacy standalone `upload` utility does not support Dimensions streams. It fails ungracefully if you attempt to use it with streams.

This utility is available to all users who have the privileges to create new items or revise existing item revisions in Dimensions. It performs an upload of a list of specified files from a user directory into the Dimensions repository.

Syntax

```
upload [ --help ] [ --version ] [ -n [ -s script-file ] ] [ -r ]
   [-C start-dir ] [ -W project-spec ] [ -D project-dir ] [ -U ]
   [ -R ] [ -f rule-file ] [ -p database-specifier ] [ -l log-file ]
   [ -N ] [ -F ] [ -E ] [ -k | -z ] [ input-file ]
```

<input-file>, -r

`upload` performs an upload of the list of files specified in `<input-file>`. If no `<input-file>` is specified, `upload` attempts to upload every file in the current working directory. If the `-r` option is specified, `upload` attempts to upload every file found recursively below the current working directory.

-U

Specifies that an unrestricted upload should be performed. In this mode, `upload` revises items for changed files it finds in the target project and additionally creates items for any files it cannot find there. In the default restricted mode, `upload` revises items if the corresponding file both exists and has changed.

-F

Specifies that a forced upload should be performed. In this mode, `upload` revises all items that it finds in the target project.

-f, <rule-file>, Pcmsfile

If no `<rule-file>` is specified, upload looks for `Pcmsfile` in the current working directory and the project default directory. The `Pcmsfile` contains filename pattern matching rules that tell the upload program which files are to be to considered for uploading and which should be excluded from consideration. The `preserve` and `nopreserve` directives are respectively used in these rules to nominate files for, or exclude files from, consideration when performing an upload; they are also used to name design parts to own the Dimensions items, types for the items and various other properties. The syntax for these directives is exactly the same as that used for Dimensions Make—please refer to the *Dimensions CM Make Guide* for details.

NOTE The `nopreserve` directive overrides the `preserve` directive if there is a filename pattern match, and that the order is important – more general pattern matches should be specified before more restrictive ones.

Because `upload` has to map files to Dimensions items, it needs more complicated rules than `download`. The minimum requirement is a rule-file that specifies an item type to use, for example:

Example

```
%:
   preserve $TYPE = SRC
```

The upload facility can work out defaults for just about everything else, except for the format to use for files without a suffix. You can use two approaches to resolve this. One is to provide a default format and override it where appropriate, for example:

Example

```
%:
   $FORMAT = TEXT
%.c %.h:
   $FORMAT = C
```

```
%.cpp CC/include/%.h:
   $FORMAT = C++
```

This example gives all files the format TEXT, except for `'.c'` and `'.h'` files which use format `C`, unless it is a `'.h'` file in the directory `'CC/include'` (or a descendent), in which case format C++ is used. Files with a `'.cpp'` suffix use format C++ too.

The other approach is to explicitly specify the format for each file without a suffix, for example:

Example

```
scripts/%:
   $FORMAT = SH
README:
   $FORMAT = TEXT
```

In this example, all files in scripts get the format `SH`, while all files called `README` use the format `TEXT`.

Configuration search paths can be used to allow the files on disk to be compared against the contents of more than one project. This is most useful when using upload to merge several products with some common code into a single Dimensions product.

Example

For example, say you have `PRODUCT1`, `PROdUCT2`, and `PRODUCT3`. First, `PRODUCT1` is loaded into `PROD:ROJECT1`. Then, when you load `PRODUCT2`, you could use the same rule-file but with the following additional directive:

```
%:
   configuration PROD:PROJECT1
```

Files that exist in both `PRODUCT1` and `PRODUCT2` are not loaded from `PRODUCT2` (unless the version in `PRODUCT2` is different). This works best if you use item types with edit at initial lifecycle state disabled and make the projects for the second and subsequent products branch projects. You can then use the project merge facilities to resolve any conflicts between the products.

-W, -D

The `-W` option specifies the project to upload into and the `-D` option specifies the directory to treat as the project working location.

If no `-W` option is specified, the current project is used, as determined from the current working location. If no `-D` option is specified, the current working location is used as the project default location.

If a -D option is specified it must be an ancestor of the starting directory, see -C option.

NOTE The current working directory must be the specified project default directory or a descendent thereof.

-n, -s

Specifying `-n` causes a command file to be generated, rather than the file being directly uploaded. The name of the script can be specified using the `-s` option. If no name is specified, the name `upload.cmd` is used and the file is placed in the current working directory.

CAUTION! In Windows, `".cmd"` is reserved as the filename extension for an executable. You must make sure that the specified filename does not end in `".cmd"`.

-p

Enables you to specify the database into which the files are to be uploaded. If not specified, the current database is used.

--version Enables you to check the version of the `upload` you are running. It merely displays the version number and then exits.

--help Prints the syntax usage.

-l Enables you to specify a file to contain error messages from the upload.

-N Causes dual-mode length and checksum matching to be disabled. The dual-mode feature matches files using the length and checksum of both the binary and text interpretation of a file where the native text file format is not the same as UNIX text file format.

Gotten Files By default (unless `-z` is specified as described below), files are gotten back to disk after they are uploaded. These gotten files are also by default unexpanded unless the addition `-E` option is specified to cause them to be expanded.

-R Causes upload to check in locally checked out files in addition to unchecked out files that have been modified locally.

-k Causes user files to be retained when they are uploaded.

-z Stops files being gotten back to disk after they have been uploaded. It overrides the normal default action which is to get these files.

-C Enables you to specify the starting directory for the upload process by `-C start-dir`. Any files found in this directory or constituent subdirectories are processed by `upload`. If not specified, the current directory is used as the starting directory.

-E Expand gotten files (default is unexpanded).

RCS-Like Front End to Dimensions: prcs

IMPORTANT! prcs is *only* supported on Dimensions for UNIX servers, clients, or agents.

This utility is available to all users who have the roles to do the associated Dimensions CM operations. It provides an RCS-like front end to the version control commands of Dimensions.

Syntax
```
prcs subcommand [ option ... ] [ filename ... ]
    [Dimensions option ...]
```

filename `prcs` applies an RCS-like subcommand to the Dimensions CM item associated with each specified `filename`. Additionally, Dimensions CM-specific options may be specified.

filename, PCMS, PCMS/<wildcard> The mapping between the filenames specified to the `prcs` command and the associated Dimensions CM item is derived from the `filename` by searching for a matching project filename in the current project. The `prcs` command expects these items to reside in a project location that, when appended to the current project working location, equates to the current working location. If the `filename` is given as `PCMS` or `PCMS/<wildcard>` then the command is applied to all Dimensions CM items which are in the same relative sublocation of the project and which match the wildcard. The wildcard is in standard Dimensions CM format.

If `prcs` is unable to determine a project, then a diagnostic message is issued. Please refer to the *Command-Line Reference* for full details on `prcs`.

SCCS-Like Front End to Dimensions: psccs

IMPORTANT! psccs is *only* supported on Dimensions for UNIX servers, clients, and agents.

This utility is available to all users who have the roles to do the associated Dimensions CM operations. It provides an SCCS-like front end to the version control commands of Dimensions CM.

Syntax

```
psccs subcommand [ option ... ] [ filename ... ] [Dimensions option ...]
```

filename

`psccs` applies an SCCS-like subcommand to the Dimensions CM item associated with each specified `filename`. Additionally, Dimensions CM-specific options may be specified.

filename, PCMS, PCMS/<wildcard>

The mapping between the filenames specified to the `psccs` command and the associated Dimensions CM item is derived from the `filename` by searching for a matching project filename in the current project. The `psccs` command expects these items to reside in a project location that, when appended to the current project working location, equates to the current working location. If the `filename` is given as `PCMS` or `PCMS/<wildcard>` then the command is applied to all Dimensions CM items which are in the same relative sublocation of the project and which match the wildcard. The wildcard is in standard Dimensions CM format.

If `psccs` is unable to determine a project, then a diagnostic message is issued. Please refer to the *Command-Line Reference* for full details on `psccs`.

Chapter 16

Data Interchange File Format (PDIFF)

Foreword	216
PDIFF Specification	216
Change Document (Dimensions CM Request) POID	217
Design Part POID	222
Control Plan (Process Model) POIDs	224
Replicator Changes to POID Definitions	226
Example PDIFF Files	227

Foreword

This section describes PDIFF, the Dimensions CM Data Interchange File Format, that is used to import or export data to or from Dimensions CM. Chapter 17, "Using PDIFF to Import/Export Data" describes the associated tool `pdiff`.

NOTE In this section, PDIFF is used when referring to the Dimensions CM Data Interchange File Format itself, and `pdiff` is used when referring to the associated tool.

PDIFF supports the following Dimensions CM data:

- Dimensions CM requests
- Design Parts
- Process Model (Control Plan)
  - Valid Sets
  - Delegation Candidates
  - User Role Assignments
  - Request Browse Templates.

PDIFF Specification

PDIFF is a text file format composed of lines no longer than 255 characters. PDIFF files may only contain blank lines, comments, and Dimensions CM Object Instance Definitions (POIDs):

Comments	These start with a # character and are terminated by an end-of-line character.
POIDs	Dimensions CM deals with a number of object types, for example, `ITEM`, `PART`, `CHDOC` (Dimensions CM request), and `CONTROL PLAN` (process model), but the scope of this manual only covers CHDOC, `PART`, and some `CONTROL PLAN` POIDs.

A PDIFF file begins with a 10-line header followed by one or more POIDs. The PDIFF header contains the PDIFF version, the creation date and time, and the name of the creator of the file. The rest of the lines in the header are reserved for possible future use.

```
PDIFF :==
   <PDIFF Header>
   <POIDS>
```

The following rules apply to the PDIFF contents:

1. The number of characters in each line of the file must not exceed 255.
2. `KEYWORD` may either be a keyword specially recognized by Dimensions CM (see below) or refer to a non-system request field.
3. `<string>` is normally a maximum 1978-character value either on the same line or spanning several lines with each line terminated with the back-slash character (`\`).

When determining a value spanning several lines, the character \ at the end of each line and the subsequent newline character are ignored. For the request descriptions, ($DETAILED_DESC, $ACTION_DESC, $AD_HISTORY), values may be longer than 1978 characters.

4 Any "\n" found within <text-value> is interpreted as a new line character and, consequently, a new line is inserted at that point within the text.

5 A "\F" found for the first two characters of values detailed in (Step 3) indicates that the following string is to be interpreted as a filename from which to get the attribute or description value.

The PDIFF header has the following format:

```
<PDIFF Header> :==
  #PDIFF<version number>
       #DATE <DD-MMM-YYYY> <HH:MM:SS>
       #CREATED BY <name>
       #RESERVED
```

Change Document (Dimensions CM Request) POID

A change document (Dimensions CM request) definition begins with the OBJECT CHDOC statement on a single line and ends with a END_OBJECT statement. The body of the CHDOC POID consists of system attributes, user attributes, actions, updates and relations.

```
<CHDOC POID> :==
  OBJECT CHDOC
         <attributes>
         <composite request attributes>
       END_OBJECT
<attributes> :==
  <system request attribute> | <user attribute> | <attributes>
<composite request attributes> :==
  <action>                                  |
  <update>                                  |
  <relation>                                |
  <delegate>                                |
  <attribute history>                       |
  <attribute update history>                |
  <composite request attributes>
<system request attributes> :==
  $DOC_SEQ <number>|
  $IDENTIFIER           <string>            |
  $ARCHIVED             {Y|N}               |
  $TYPE                 <string>            |
  $CREATE_DATE          <date>              |
  $TIMEZONE             <timezone>          |
  $CURRENT_STATUS       <string>            |
  $ORIGINATOR           <string>            |
  $LIFECYLE_ID          <string>            |
  $LC_SEQ               <number>            |
  $DETAILED_DESC        <long string>       |
  $ACTION_DESC
```

```
   .CURRENT      <long string>
   .ALL_PREV     <long string>           |
$AD_HISTORY
   .ACTION_NO    <number>
   .FILE         \F<filename>            |
$ATTACHMENT
   .FILE         <filename>
   .USER_FILE    <filename>
   .USE          <string>
   .COMMENT      <string>
   .DATE         <date>                  |
$UPDATE_DATE          <date>             |
$OWNER_SITE           <string>           |
$REFERENCE_ONLY       <string>           |
$LOCKED               <string>           |
$DELEGATED_OWNERSITE  <string>
```

System Change Document (Request) Attributes

System attributes for a Dimensions CM change document (request) can be one of:

- Request Sequence – (**mandatory**)

  This is a unique sequence number given to each request within a product.

- Request Identifier – (**optional**)

  The request identifier is composed of the product identifier, the request type, and the Request Sequence number:

  ```
  <product id>_<type>_<doc_seq>
  ```

- Archive flag – (**optional**)

  A flag to indicate if the request is to be archived or not, the default is not to archive the request.

- Type of the Request – (**mandatory**)
- Date the Request was Created – (**mandatory**)
- Timezone – (**optional**)

  This indicates the timezone in which the PDIFF file was generated. Currently, this field is for information purposes only, and is ignored on load.

- Current Status of the Request – (**mandatory**)
- The Originator of the Request – (**mandatory**)
- Lifecycle Id – (**optional**)

  The lifecycle used by the request.

- Lifecycle Sequence – (**optional**)

  The lifecycle sequence is a number indicating the sequence of the current status within the lifecycle. The lifecycle and the lifecycle sequence must be both specified together or not at all for a given request.

- The Detailed Description of the Request – (**optional**)
- Action Description (this has two parts) – (**optional**)

The current action description and concatenation of all previous action descriptions.

- Action Description History (this has two parts) – (**optional**)

  The action description in a file together with an action number. There may be many of these statements, each for a different action number.

- Request Attachment – (**optional**)
- The Update Date of the Request – (**optional**)

  The format is DD-MM-YYYY HH:MM:SS.

- The replication site that currently owns a request – (**optional**)
- Whether or not the request is reference only with respect to replication – (**optional**)

  The format is N or Y.

- Whether or not the request is locked with respect to replication – (**optional**)

  The format is N or Y.

- The replication site identity that has been delegated the ownership of a request at the time of replication – (**optional**)

User Attribute – (Optional)

Dimensions user attributes are defined by the user with Dimensions product privileges as part of the process model (control plan) definition for the Dimensions product. Each user-defined attribute has a variable name, and this is used in the PDIFF to refer to this attribute. In PDIFF, user attributes are specified as:

```
<user attribute> :==
   <attribute variable name>   <attribute value>
<attribute variable name> :==
   <string>
<attribute value> :==
   <string>
```

NOTE `<attribute variable name>` cannot contain white space characters. When importing a request, the loader does not validate the values of the request attributes against their associated valid sets, if any.

Action - (Optional)

An action (that is, a state transition such as a sign-off or rejection) performed by a user. There may be several of these for the different request action numbers.

NOTE Each state transition performed by a user on a request is represented in Dimensions as a separate action number. The action number starts from 0 when the request is created in the user's held list (that is, private area). When the request is entered (saved) into the system it is given the action number 1. The action number is then incremented each time the request is actioned to a new state.

```
<action> :==
   $ACTION
   .ACTION_NO       <number>
```

```
.DATE           <date>
.USER           <string>
.STATE          <string>
.PHASE          <string>
.TYPE           A
.COMMENT        <max 80 string>
```

NOTE The .COMMENT value must be one of the following: "Document created" or "Actioned document from <STATE1> to <STATE2>".

Only the first 80 characters of the .COMMENT value are read; the rest are ignored. The .TYPE specifier is optional.

Update – (Optional)

This is a record of the update history for the request at specific action numbers. There is an optional .TYPE specifier to indicate Update (U) or Delegate (D) history—there may be several such entries. The update statement consists of the action number at which the update was performed, user who performed the update, the state in which the update was made, date of update, and an update comment if any.

```
<update> :==
   $UPDATE
      .ACTION_NO     <number>
      .DATE          <date>
      .USER          <string>
      .STATE         <string>
      .PHASE         <string>
      .TYPE          {U|D}
      .COMMENT       <max 80 string>
```

NOTE The .COMMENT value must be one of the following: "Change document attribute(s) updated", "Added Delegation for...", "Related Design Part...", or "Added action description".

Only the first 80 characters of the .COMMENT value are read; the rest are ignored. The .TYPE specifier is optional.

Relate – (Mandatory: at Least One Design Part Relate Required)

This statement defines a single relationship which the request has with the specified affected object. The relation statement consists of the action number at which the relationship was created, type of the relationship (for example, DEPENDENT, AFFECTED, INFO, IN RESPONSE TO, or "user defined type"), the class of the related object (for example, CHDOC, ITEM, or PART), the date the relationship was created and the user who created the relationship.

NOTE A user-defined relationship type must be defined using the Process Modeling Scripting Interface. See the *online help*.

For requests, the destination object may be a child or a parent. In the .OBJECT part of the $RELATE statement a CHDOC: implies a child and a PARENT_CHDOC: implies a parent.

```
<relation> :==
   $RELATE
      .ACTION_NO     <number>
      .TYPE          <string>
      .OBJECT        CHDOC:<request-id>            |
                     PARENT_CHDOC:<request-id>     |
                     PART:<part spec>              |
                     ITEM:<item spec>              |
                     BASELINE:<baseline spec>
      .DATE          <date>
      .USER          <string>
```

Delegate – (Optional)

This statement enables the delegation of the request to a specified *user* having the specified *role*. There are two optional specifiers: .CAPABILITY and .OPERATION. The .CAPABILITY specifier may be one of Secondary (S), Primary (P), or Leader (L) with the default being Primary (P) and .OPERATION may be one of ADD or REPLACE.

The delegate statement has the following syntax:

```
<delegate> :==
   $DELEGATE
      .USER          <string>
      .ROLE          <string>
      .CAPABILITY    { S | P | L }
      .OPERATION     { ADD | REPLACE }
```

Attribute History – (Optional)

This statement enables attribute history to be added for each action number of a request. The specifiers are .ACTION_NO for the action number of the history and .ATTRIBUTE consisting of the attribute name, the sequence, and the attribute value. The sequence starts at 1 and indicates if the attribute had multi values. All single valued attributes must have a sequence number of 1. The attribute number is in the range 1 to 240. The .ATTRIBUTE specifier is repeated for each attribute name and attribute sequence number.

```
<attribute history> :==
   $ATTRIBUTE_HISTORY
      .ACTION_NO   <number>
      <attribute info list>
<attribute info list> :==
   <attribute info> [<attribute info list>]
<attribute info> :==
   .ATTRIBUTE      <attr_name>:<attr seq>:<string>
```

Attribute Update History – (Optional)

This statement enables attribute update history to be added for each update to an attribute. The specifiers .ACTION_NO, .DATE, .USER, .STATUS must correspond to a $UPDATE statement in the request POID.

```
<attribute update history> :==
   $ATTRIBUTE_UPDATE_HISTORY
      .ACTION_NO     <number>
      .ATTR_NO       <number>
      .SEQ_NO        <number>
      .DATE          <date>
      .USER          <string>
      .STATUS        <string>
      .PREV_VALUE    <string>
      .NEW_VALUE     <string>
```

Request Attachments – (Optional)

This statement supports the dumping and loading of request attachments from and to base databases. The specifiers are `.FILE` for the internal database "filename" identity for the attachment, `.USER_FILE` for the operating system filename to load attachment from, `.USER` for the user who created the attachment, `.COMMENT` for a textual comment regarding the attachment, and `.DATE` for the date the attachment was created.

```
<attachment> :==
   $ATTACHMENT
      .FILE         <filename>
      .USER_FILE    <filename>
      .USE          <string>
      .COMMENT      <string>
      .DATE         <date>
```

Design Part POID

A design part definition begins with the `OBJECT PART` statement on a single line and ends with a `END_OBJECT` statement. The body of the `PART POID` consists of system attributes, user attributes, relations, and relation histories.

```
<PART POID> :==
   OBJECT PART
      <attributes>
      <relate_history>
   END_OBJECT
<attributes> :==
   <system part attribute>       |
   <user attribute>              |
   <attributes>
<composite part attributes> :==
   <part relate>                 |
   <part relate history>         |
   <composite part attributes>

<system part attributes> :==
   $PRODUCT_ID       <string>    |
   $PART_ID          <string>    |
   $VARIANT          <string>    |
   $CURRENT_PCS      <string>    |
```

```
$PCS              <string>    |
$STATUS           <string>    |
$DESCRIPTION      <string>    |
$CREATE_DATE      <string>    |
$CATEGORY         <string>    |
$USER             <string>    |
$PREV_PART        <part spec>
```

System Design Part Attributes

System Attributes for a Dimensions CM design parts can be one of:

- Product Identifier – (**mandatory**)

  This is the owning Dimensions CM Product of the PART POID.

- Design Part Identifier – (**mandatory**)

  This is the identity of the PART POID.

- Design Part Variant – (**mandatory**)

  This is the variant of the PART POID.

- Part Control Status (PCS or CURRENT_PCS) – (**mandatory**)

  This is the revision of the PART POID. $PCS or $CURRENT_PCS may be used as the specifier. $CURRENT_PCS is used for design parts that are the most current revisions and that are OPEN. $PCS is used for DESIGN PARTS that are CLOSED or REJECTED.

- Status – (**mandatory**)

  This is the lifecycle status of the PART POID and has one of the following values: OPEN, CLOSED, or SUSPENDED.

- Description – (**optional**)

  This is the description of the PART POID.

- Create Date – (**mandatory**)

  This is the creation date of the PART POID.

- Category – (**mandatory**)

  This is the design part type category of the PART POID.

- User – (**mandatory**)

  This is the owning user of the PART POID.

- Previous Design Part – (**optional**)

  This is a reference to a design part specification from which this PART POID was derived when the design part revision (PCS) was changed.

User Attribute – (Optional)

Dimensions CM user attributes are defined by the user with product management privileges as part of the process model definition for the Dimensions CM product. Each user-defined attribute has a variable name, and this is used in the PDIFF to refer to this attribute. In PDIFF user attributes are specified as:

```
<user attribute> :==
   <attribute variable name> <attribute value>
<attribute variable name> :==
   <string>
<attribute value> :==
   <string>
```

NOTE <attribute variable name> cannot contain whitespace characters. When importing a request, the loader does not validate the values of the request attributes against their associated valid sets, if any.

Relate – (Optional)

This statement defines a single breakdown or usage relationship to another design part. PARENT_PART in the .OBJECT specifier indicates the relationship is to be reversed. All design parts other than the root ones must have parent BREAKDOWN relationships.

```
<part relate> :==
   $RELATE
      .TYPE          {BREAKDOWN|USAGE}
      .OBJECT        PARENT_PART:<part spec>   |
                     PART:<part spec>
      .DATE          <date>
      [.USER         <string>]
```

Relate History – (Optional)

This statement defines the history for the changes made to the design part BREAKDOWN and USAGE relationships.

```
<part relate history> :==
   $RELATE_HISTORY
      .TYPE          {BREAKDOWN|USAGE}
      .OBJECT        <part spec>
      .DATE          <date>
      [.USER         <string>]
```

Control Plan (Process Model) POIDs

There are three control plan (process model) POIDs currently supported by PDIFF:

1. VALID_SET, DELEGATEES
2. CHDOC_BROWSE_TEMPLATES
3. ITEM_BROWSE_TEMPLATES

Process Model Attribute Valid Sets

The valid set POID defines multiple choice for attribute values for the given Dimensions CM product process model (control plan). The valid set POID is defined as follows:

```
<VALID SET POID> :==
  OBJECT VALID_SET
     $PRODUCT_ID      <string>
     $VALID_SET_NAME  <string>
     $NO_COLUMNS      <number>
     $ERROR_MESSAGE   <string>
     $DESCRIPTION     <string>
     <valid set values>
  END_OBJECT
<valid set values> :==
  <value set value> [<valid set values>]
<valid set value> :==
     $VALUE
        .DISP_ORDER   <number>
        .COL1         <string>
        .COL2         <string>
        .COL3         <string>
        .COL4         <string>
        .COL5         <string>
        .COL6         <string>
        .COL7         <string>
        .COL8         <string>
```

Process Model Delegatees

The delegatee POID defines user role assignments and delegation candidates for a given Dimensions CM product process model (control plan).

```
<VALID SET POID> :==
  OBJECT DELEGATEE
     $PRODUCT_ID    <string>
     <delegatees user_role_assignments>
  END_OBJECT
<delegatees user_role_assignments> :==
  <delegatee user_role assignment>
     [<delegatees user_role_assignments>]
<delegatee user_role_assignment> :==
  <delegatee> | <user_role_assignment>
<delegatee> :==
  $DELEGATEE
     .ROLE          <string>
     .USER          <string>
     .CAPABILITY    [P|S|L]
     .PART_ID       <string>
     .PART_VARIANT  <string>
     .WORKSET_ID    <string>
<user_role_assignment> :==
  $USER_ROLE_ASSIGNMENT
     .ROLE          <string>
```

```
   .USER          <string>
   .CAPABILITY    [P|S|L]
   .PART_ID       <string>
   .PART_VARIANT  <string>
   .WORKSET_ID    <string>
```

The specifiers .PART_VARIANT and .WORKSET_ID are optional.

Process Model Change Request and Item Browse Templates

The request and item browse templates POID define the browse template files in the given Dimensions product for the browse template identifiers and revisions.

```
<CHDOC_BROWSE_TEMPLATES POID> :==
  OBJECT CHDOC_BROWSE_TEMPLATES
     $PRODUCT_ID   <string>
        <browse template definitions>
  END_OBJECT
<ITEM_BROWSE_TEMPLATES POID> :==
  OBJECT ITEM_BROWSE_TEMPLATES
     $PRODUCT_ID   <string>
        <browse template definitions>
  END_OBJECT
<browse template definitions> :==
  <browse template definition>
     [<browse template definitions>]
<browse template definition> :==
  $TEMPLATE_DEF
     .IDENTIFIER   <string>
     .REVISION     <string>
     .FILE         \F<filename>
```

Replicator Changes to POID Definitions

The POID definition has been extended to allow the replication of both relate and unrelate operations for items:

- RII, XII, RICD, and XICD ("affected" relationships only).
- RIP and XIP ("usage" relationships only).

For example:

- POID data representing RII:

```
$RELATE
   .TYPE          SRC-SRC
   .OBJECT        ITEM:REPX:ESG_SRC_ITEM1_H.A-SRC;host1\#1
   .DATE          09-FEB-2009 15:30:36
   .USER          DMSYS
```

- POID data representing RICD:

```
$RELATE
   .TYPE          Affected
   .OBJECT        CHDOC:REPX_CR_105
   .DATE          09-FEB-2009 15:39:37
   .USER          DMSYS
```

- POID date representing RIP:

```
$RELATE
.TYPE             Usage
.OBJECT           PART:REPX:SAILOR_MARS.A;1
.DATE             09-FEB-2009 15:42:24
.USER             DMSYS
```

- POID data representing XII:

```
$UNRELATE
.TYPE             SRC-SRC
.OBJECT           ITEM:REPX:ESG_SRC_ITEM1_H.A-SRC;host1\#1
.DATE             09-FEB-2009 15:30:36
.USER             DMSYS
```

- POID data representing XICD:

```
$UNRELATE
.TYPE             Affected
.OBJECT           CHDOC:REPX_CR_105
.DATE             09-FEB-2009 15:39:37
.USER             DMSYS
```

- POID date representing XIP:

```
$UNRELATE
.TYPE             Usage
.OBJECT           PART:REPX:SAILOR_MARS.A;1
.DATE             09-FEB-2009 15:42:24
.USER             DMSYS
```

Example PDIFF Files

Change Document (Request) Example

```
#PDIFF3
#DATE 09-FEB-2006 12:37:0
#CREATED BY TJ
#RESERVED
#RESERVED
#RESERVED
OBJECT CHDOC
   $DOC_SEQ             1
   $IDENTIFIER          VB1_NETSTAT_1
   $ARCHIVED            N
   $TYPE                NETSTAT
   $CREATE_DATE         04-JAN-2006 13:18:18
   $CURRENT_STATUS      LOGGED
```

```
$ORIGINATOR          TJ
$LIFECYCLE_ID        NETSTAT_LC
$LC_SEQ              3
$DETAILED_DESC       \F.//chdoc/transfer_netstat_1.desc
$DESC_CHECKSUM       46082
$ACTION_DESC
   .CURRENT
   .ALL_PREV
$AD_HISTORY
   .ACTION_NO        1
   .FILE             \F.//chdoc/transfer_netstat_1.act1
   .AD_HIST_CHECKSUM 0
$AD_HISTORY
   .ACTION_NO        2
   .FILE             \F.//chdoc/transfer_netstat_1.act2
TITLE                Mail - Network
SEVERITY             Default
NETWORK_NODE         mail@internet.loc.com
NETWORK_NODE         guest@darpa.us.com
NETWORK_TYPE         DARPA*NET
NETWORK_TYPE         Internet
$ACTION
   .ACTION_NO        1
   .DATE             04-JAN-2006 13:18:18
   .USER             TJ
   .STATE            PENDING
   .PHASE            AN+WORK
   .TYPE             A
   .COMMENT          Document created
$ACTION
   .ACTION_NO        2
   .DATE             05-JAN-2006 10:28:12
   .USER             TJ
   .STATE            LOGGED
   .PHASE            CLOSED
   .TYPE             A
   .COMMENT          Actioned document from PENDING to
                     LOGGED
$UPDATE
   .ACTION_NO        0
   .DATE             04-JAN-2006 13:18:18
   .USER             TJ
   .STATE            $TO_BE_DEFINED
   .PHASE            HELD
   .TYPE             U
   .COMMENT          Related Design Part VB1:NETWORK MANAGEMENT.GN;1
$UPDATE
   .ACTION_NO        1
   .DATE             04-JAN-2006 13:32:00
   .USER             TJ
   .STATE            PENDING
   .PHASE            AN+WORK
   .TYPE             U
   .COMMENT          Added action description

$UPDATE
```

```
      .ACTION_NO        1
      .DATE              04-JAN-2006 13:32:00
      .USER             TJ
      .STATE            PENDING
      .PHASE            AN+WORK
      .TYPE             U
      .COMMENT          Added action description
   $UPDATE
      .ACTION_NO        1
      .DATE             04-JAN-2006 13:37:06
      .USER             TJ
      .STATE            PENDING
      .PHASE            AN+WORK
      .TYPE             U
      .COMMENT          Change document attribute(s)updated
   $RELATE
      .ACTION_NO        0
      .TYPE             AFFECTED
      .OBJECT           PART:VB1:NETWORK MANAGEMENT.GN;1
      .DATE             04-JAN-2006 13:18:03
      .USER             TJ
   $DELEGATE
      .USER             VB
      .ROLE             SUPPORT ENGINEER
      .CAPABILITY       S
      .OPERATION        REPLACE
   $ATTRIBUTE_HISTORY
      .ACTION_NO        1
      .ATTRIBUTE           TITLE: 1:Mail - Network
      .ATTRIBUTE           OPTION: 1:Default
      .ATTRIBUTE         ADDRESS1: 1:mail @internet.loc.com
      .ATTRIBUTE         ADDRESS1: 2:guest@darpa.us.com
      .ATTRIBUTE         ALIAS: 1:DARPA*NET
      .ATTRIBUTE         ALIAS: 2:Internet
   $ATTRIBUTE_UPDATE_HISTORY
      .ACTION_NO         1
      .ATTR_NO           ADDRESS1
      .SEQ_NO            1
      .DATE              04-JAN-2006  13:37:06
      .USER              TJ
      .STATUS            PENDING
      .PREV_VALUE        baggins@shire.middleearth.equ
      .NEW_VALUE
END_OBJECT
```

Design Part Example

```
#PDIFF3
#DATE 09-FEB-2006 17:41:37
#CREATED BY
#RESERVED
OBJECT PART
   $PRODUCT_ID          VB1
   $PART_ID             VB1
   $VARIANT             AAAA
```

```
   $CURRENT_PCS         1
   $STATUS              OPEN
   $DESCRIPTION         VB1
   $CREATE_DATE         13-JAN-2006 18:19:46
   $CATEGORY            PRODUCT
   $USER                VB
   # Start of User Defined Design Part Attributes
   EXPORT_LINK          Not Defined
   SECURITY             Level 1
   # End of User Defined Design Part Attributes
   $RELATE
      .TYPE             BREAKDOWN
      .OBJECT           PART:VB1:TEST DATA.AAAA;2
      .DATE             14-JAN-2006 17:33:15
      .USER             VB
   $RELATE
      .TYPE             BREAKDOWN
      .OBJECT           PART:VB1:TEST DATA.UNIX;2
      .DATE             14-JAN-2006 17:38:14
      .USER             VB
   $RELATE_HISTORY
      .TYPE             BREAKDOWN
      .OBJECT           VB1:TEST DATA.AAAA;1
      .DATE             14-JAN-2006 17:34:04
      .USER             VB
END_OBJECT
OBJECT PART
   $PRODUCT_ID          VB1
   $PART_ID             TEST DATA
   $VARIANT             AAAA
   $PCS                 1
   $STATUS              CLOSED
   $DESCRIPTION         Design Part to Hold Test Data
   $CREATE_DATE         14-JAN-2006 17:33:15
   $CATEGORY            MODULE
   $USER                VB
   # Start of User Defined Design Part Attributes
   # End of User Defined Design Part Attributes
END_OBJECT
OBJECT PART
   $PRODUCT_ID          VB1
   $PART_ID             TEST DATA
   $VARIANT             AAAA
   $CURRENT_PCS         2
   $STATUS              OPEN
   $DESCRIPTION         Design Part to Hold Test Data
                        with new PCS
   $CREATE_DATE         14-JAN-2006 17:34:07
   $CATEGORY            MODULE
   $USER                VB
   $PREV_PART           VB1:TEST DATA.AAAA;1
   # Start of User Defined Design Part Attributes
   EXPORT_LINK          Not Defined
   SECURITY             Level 2
   # End of User Defined Design Part Attributes
   $RELATE
```

```
      .TYPE             BREAKDOWN
      .OBJECT           PART:VB1:DOCUMENTS.UNIX;2
      .DATE             14-JAN-2006 17:39:57
      .USER             VB
   $RELATE
      .TYPE             BREAKDOWN
      .OBJECT           PARENT_PART:VB1:VB1.AAAA;1
      .DATE             14-JAN-2006 17:33:15
      .USER             VB
END_OBJECT
OBJECT PART
   $PRODUCT_ID          VB1
   $PART_ID             TEST DATA
   $VARIANT             UNIX
   $CURRENT_PCS         2
   $STATUS              OPEN
   $DESCRIPTION         Design Part to Hold Test Data
                        with new PCS
   $CREATE_DATE         14-JAN-2006 17:38:13
   $CATEGORY            MODULE
   $USER                VB
   $PREV_PART           VB1:TEST DATA.AAAA;2
   # Start of User Defined Design Part Attributes
   EXPORT_LINK          Not Defined
   SECURITY             Level 2
   # End of User Defined Design Part Attributes
   $RELATE
      .TYPE             USAGE
      .OBJECT           PART:VB1:DOCUMENTS.UNIX;2
      .DATE             14-JAN-2006 17:40:54
      .USER             VB
   $RELATE
      .TYPE             BREAKDOWN
      .OBJECT           PARENT_PART:VB1:VB1.AAAA;1
      .DATE             14-JAN-2006 17:38:14
      .USER             VB
END_OBJECT
OBJECT PART
   $PRODUCT_ID          VB1
   $PART_ID             DOCUMENTS
   $VARIANT             UNIX
   $CURRENT_PCS         2
   $STATUS              OPEN
   $DESCRIPTION         Part to hold TEST data Documents
   $CREATE_DATE         14-JAN-2006 17:39:16
   $CATEGORY            MODULE
   $USER                VB
   # Start of User Defined Design Part Attributes
   EXPORT_LINK          Not Defined
   SECURITY             Level 4
   # End of User Defined Design Part Attributes
   $RELATE
      .TYPE             BREAKDOWN
      .OBJECT           PARENT_PART:VB1:TEST DATA.AAAA;2
      .DATE             14-JAN-2006 17:39:57
      .USER             VB
```

```
END_OBJECT
```

Process Model Example

```
#PDIFF3
#DATE 09-FEB-2006 15:59:7
#CREATED BY
#RESERVED
#RESERVED
#RESERVED
#RESERVED
#RESERVED
#RESERVED
#RESERVED
OBJECT VALID_SET
   $PRODUCT_ID          VB1
   $VALID_SET_NAME      ACT_REL_PLAT_OS
   $NO_COLUMNS          3
   $VALUE
      .DISP_ORDER       0
      .COL1             1.0.0
      .COL2             Sun Sparc
      .COL3             SunOS 4.1.3
   $VALUE
      .DISP_ORDER       0
      .COL1             1.0.0
      .COL2             VAX
      .COL3             VMS 5.5
   $VALUE
      .DISP_ORDER       0
      .COL1             1.0.1
      .COL2             Sun Sparc
      .COL3             SunOS 4.1.3
   $VALUE
      .DISP_ORDER       0
      .COL1             1.2.0
      .COL2             IBM RS6000
      .COL3             AIX 3.2.5
   $VALUE
      .DISP_ORDER       0
      .COL1             1.2.0
      .COL2             Sequent
      .COL3             Dynix/Ptx 2.1
   $VALUE
      .DISP_ORDER       0
      .COL1             2.0.0
      .COL2             Alpha AXP
      .COL3             OSF1 V2.0
   $VALUE
      .DISP_ORDER       0
      .COL1             2.0.0
      .COL2             Alpha AXP
      .COL3             OpenVMS 6.1
   $VALUE
      .DISP_ORDER       0
```

```
      .COL1             2.0.0
      .COL2             HP 9000 700/800
      .COL3             HP UX A09.03/A09.04
   $VALUE
      .DISP_ORDER       0
      .COL1             2.0.0
      .COL2             Sun Sparc
      .COL3             Solaris 2.3
   $VALUE
      .DISP_ORDER       0
      .COL1             2.0.0
      .COL2             Sun Sparc
      .COL3             SunOS 5.3
END_OBJECT
OBJECT DELEGATEES
   $PRODUCT_ID          VB1
   $USER_ROLE_ASSIGNMENT
      .ROLE             $CHANGE-MANAGER
      .USER             VB
      .CAPABILITY       S
      .PART_ID          VB1
   $USER_ROLE_ASSIGNMENT
      .ROLE             $PRODUCT-MANAGER
      .USER             TIM
      .CAPABILITY       S
      .PART_ID          VB1
      .PART_VARIANT     AAAA
      .WORKSET_ID       WS_V2.1
   $DELEGATEE
      .ROLE             AUTHOR
      .USER             WILLIAM
      .CAPABILITY       P
      .PART_ID          DOCUMENTATION
      .WORKSET_ID       WS_V2.1
END_OBJECT
OBJECT CHDOC_BROWSE_TEMPLATES
   $PRODUCT_ID          VB1
   $TEMPLATE_DEF
      .IDENTIFIER       PR_TEMPLATE
      .REVISION         1
      .FILE             \F.//vb1_template_1.bt
END_OBJECT
OBJECT ITEM_BROWSE_TEMPLATES
   $PRODUCT_ID          VB1
   $TEMPLATE_DEF
      .IDENTIFIER       SRC_TEMPLATE
      .REVISION         1
      .FILE             \F.//vb1_I_template_1.bt
   $TEMPLATE_DEF
      .IDENTIFIER       TXT_TEMPLATE
      .REVISION         1
      .FILE             \F//usr1/tmp/vb1_I_template_2.bt
END_OBJECT
```

Chapter 17

Using PDIFF to Import/Export Data

General Description	236
Process Model Export	236
Process Model Import	237
Design Part Export	238
Design Part Import	238
Request Export	239
Request Import	241
Submitting PDIFF Commands from a Parameter File	242
Limitations	242

General Description

NOTE In this section, PDIFF is used when referring to the Dimensions CM Data Interchange File Format itself, and `pdiff` is used when referring to the associated tool.

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

```
<base_database_id>@<db_connection>
```

For example, in Dimensions CM for Windows:

```
set DMDB=cm_typical@dim10
```

`pdiff` is a tool for importing or exporting data through a PDIFF file to or from a specified Dimensions CM product. PDIFF files are ASCII text files, see Chapter 16, "Data Interchange File Format (PDIFF)".

`pdiff` runs within the Dimensions CM environment and supports the importing or exporting of Dimensions CM requests, design parts, and a partial process model (control plan).

The `pdiff` tool in import mode reads Dimensions CM Object Instance Definitions (POIDs) sequentially from the input PDIFF files and creates the equivalent object instance in the Dimensions CM base database; in export mode the database is read and a PDIFF file together with any POID related files are created.

pdiff principal parameter	Function
-dump_chdocs	Dump requests from a product into a PDIFF file.
-load_chdocs	Load requests from PDIFF files into a product.
-dump_parts	Dump design parts from a product into a PDIFF file.
-load_parts	Load design parts from PDIFF files into a Dimensions CM product.
-dump_cpl	Dump a partial process model (control plan) from a product into a PDIFF file.
-load_cpl	Load a partial process model (control plan) from PDIFF files into a product
-params	Submit the pdiff commands from a parameter file rather than interactively.

Process Model Export

The PDIFF process model (control plan) export supports the following PDIFF POIDs:

```
VALID_SET
   Attribute Validation Sets
DELEGATEES
   Delegation Candidates and User Role Assignments
CHDOC_BROWSE_TEMPLATES
   Request Browse Template Definitions
ITEM_BROWSE_TEMPLATES
   Item Browse Template Definitions
```

Syntax

```
pdiff <product_id> -dump_cpl
   [-rename_prod_id <product_id>]
   [-f <dump_file>]
   [-working_dir <dump directory>]
```

<product_id> Specify the product from which to export the process model (control plan).

-dump_cpl Specify the `pdiff` command option.

-rename_prod_id `-rename_prod_id <product_id>`

Optionally specify that all references to the original Product Id are to be replaced by `<product_id>` in the PDIFF file.

-f `-f <dump_file>`

Optionally specify that the process model (control plan) is to be exported into the PDIFF file `<dump_file>`. Defaults to `<product_id>.dump_cntrl`.

-working_dir `-working_dir <dump directory>`

Optionally specify the directory used for storing any request browse template files.

Process Model Import

The PDIFF process model (control plan) import supports the following PDIFF POIDs:

```
VALID_SET
   Attribute Validation Sets
DELEGATEES
   Delegation Candidates and User Role Assignments
CHDOC_BROWSE_TEMPLATES
   Request Browse Template Definitions
ITEM_BROWSE_TEMPLATES
   Item Browse Template Definitions
```

Syntax

```
pdiff <product_id> -load_cpl
   [-d]
   [-l <log file>]
   [-f <pdiff_file>]
```

<product_id> Specify the product into which to import the process model (control plan).

-load_cpl Specify the `pdiff` command option.

-d Optionally specify that any existing process model (control plan) objects with identical identifiers are to be deleted before importing the PDIFF POIDs. If this option is not specified, any PDIFF POIDs that exist within the target product are not imported.

-l `-l <log file>`

Optionally specify that any verbose output from `pdiff` is to be placed in the log file `<log file>`.

-f `-f <pdiff_file>`

Optionally specify that the process model (control plan) is to be imported from the PDIFF file `<pdiff_file>`. Defaults to reading from the operating system standard input.

Design Part Export

The design part export exports the entire design part structure from a given product. The exported PDIFF file contains all design part variants, PCSs, breakdown and usage relationships as well as the user-defined attributes for each design part.

Syntax
```
pdiff <product_id> -dump_parts
   [-rename_prod_id <product_id>]
   [-f <dump_file>]
```

<product_id> Specify the product from which to export the design parts.

-dump_parts Specify the `pdiff` command option.

-rename_prod_id `-rename_prod_id <product_id>`

Optionally specify that all references to the original Product Id are to be replaced by `<product id>` in the PDIFF file.

-f `-f <dump_file>`

Optionally specify that the design parts are to be exported into the PDIFF file `<dump_file>`. Defaults to `<product_id>.dump_parts`.

Design Part Import

Syntax
```
pdiff <product_id> -load_parts
   [-d]
   [-l <log file>]
   [-f <pdiff_file> <pdiff_file> ...]
```

<product_id> Specify the product into which to import the design parts.

-load_parts Specify the `pdiff` command option.

-d Optionally specify that any existing design part objects with identical identifiers are to be deleted before importing the PDIFF POIDs. If this option is not specified, any PDIFF POIDs that exist within the target product are not imported.

-l `-l <log file>`

Optionally specify that any verbose output from `pdiff` is to be placed in the log file `<log file>`.

-f `-f <pdiff_file> <pdiff_file> ...`

Optionally specify that the design parts are to be imported from the PDIFF file(s) `<pdiff_file>`. Defaults to reading from the operating system standard input.

CAUTION! A design part can only be loaded into an empty product, that is, one that does not contain any items.

Request Export

Syntax

```
pdiff <product_id> -dump_chdocs
   [-chdoc <request_id>]
   [-remap_file <remap_attr_filename>]
   [-user <user_id>]
   [-type <request_type>]
   [-f <dump_file>]
   [-status <request_state>]
   [-working_dir <dump directory>]
   [-rename_prod_id <product_id>]
   [-gmt]
   [-dump_secondary]
   [-dump_parent_chdoc_rels]
   [-dont_dump_part_rels]
   [-dont_dump_ws_rels]
   [-dont_dump_item_rels]
   [-dont_dump_chdoc_rels]
   [-dont_dump_revised_baseline]
   [-dont_dump_extracted items]
   [-dont_dump_attr_names]
```

<product_id> Specify the product from which to export the Dimensions CM requests.

-dump_chdocs Specify the `pdiff` command option.

-chdoc `-chdoc <request_id>`

Optionally specify a filter to be applied to the request(s) selected for export. This filter is applied on the request identifier. Wildcard filters are to be applied as per ANSI-SQL.

-remap_file `-remap_file <remap_attr_filename>`

File to allow attribute names to be remapped. The format of the entries in the file should be:

```
<old_attribute_name><TAB_character><new_attribute_name>
```

-user `-user <user_id>`

Optionally specify a filter to be applied to the request(s) selected for export. This filter is applied on the request originator. Wildcard filters are to be applied as per ANSI-SQL.

-type	`-type <request_doc_type>` Optionally specify a filter to be applied to the request(s) selected for export. This filter is applied on the request type. Wildcard filters are to be applied as per ANSI-SQL.
-status	`-status <request_state>` Optionally specify a filter to be applied to the request(s) selected for export. This filter is applied on the request status. Wildcard filters are to be applied as per ANSI-SQL.
-working_dir	`-working_dir <dump directory>` Optionally specify the directory used for storing any request description files. Defaults to the current working directory.
-rename_prod_id	`-rename_prod_id <product_id>` Optionally specify that all references to the original Product Id are to be replaced by `<product id>` in the PDIFF file.
-gmt	Optionally specify that all request dates are to be converted to GMT format.
-dump_secondary	Optionally specify that the export applies to requests in the secondary catalog. Defaults to requests from the primary catalog.
-dump_parent_chdoc_rels	Dump the objects related as the parent to the request. This dumps the relationship not the object itself.
-dont_dump_part_rels	Optionally specify that design part relationships to requests are not placed in the PDIFF POIDs.
-dont_dump_ws_rels	Dump the related projects. This dumps the relationship not the object itself.

CAUTION! This option is only supported with respect to projects—streams are not supported.

-dont_dump_item_rels	Optionally specify that item relationships to requests are not placed in the PDIFF POIDs.
-dont_dump_chdoc_rels	Optionally specify that request relationships to requests are not placed in the PDIFF POIDs.
-dont_dump_revised_baseline	Optionally specify that revised baseline relationships to requests are not placed in the PDIFF POIDs.
-f	`-f <dump_file>` Optionally specify that the requests are to be exported into the PDIFF file `<dump_file>`. Defaults to `<product_id>.dump_chdocs`.

CAUTION! Currently, if you dump requests using a filter, such as the request identity, there is no guarantee that related parent requests that do not match that filter are also dumped.

-dont_dump_extracted_items	Optionally specify that extracted items that are related to requests are not to be dumped.

-dont_dump_attr_names

By default, attributes values are dumped in name format, for example:

```
.ACTION_NO 1
.ATTRIBUTE TITLE; 1:REPX-chan request
.ATTRIBUTE SEVERITY: 1:1_critical
.ATTRIBUTE RELEASE: 1:1.0.0
```

If the -dont_dump_attr_names parameter is set, then the attribute values are dumped in number format, for example:

```
.ACTION_NO 1
.ATTRIBUTE 1; 1:REPX-chan request
.ATTRIBUTE 10: 1:1_critical
.ATTRIBUTE 16: 1:1.0.0
```

Request Import

Syntax

```
pdiff <product_id> -load_chdocs
   [-d][-dont_delegate]
   [-default_part <part spec>]
   [-dont_validate_user]
   [-gmt]
   [-dont_cm_pend]
   [-l <log file>]
   [-g <proxy_user>]
   [-f <pdiff_file> <pdiff_file> ...]
```

<product_id> Specifies the product into which to import the Dimensions CM requests.

-load_chdocs Specify the `pdiff` command option.

-d Optionally specify that any existing request objects with identical identifiers are to be deleted before Importing the PDIFF POIDs. If this option is not specified, any PDIFF POIDs that exist within the target product are not imported.

-dont_delegate Optionally specify that the request user role delegations are not to be imported.

-default_part `-default_part <part spec>`

Optionally specify the design part to relate to if a POID is encountered with no design part relationships. Requests must *Affect* at least one design part. Defaults to failing the request POID import if no design part relationships are specified.

-dont_validate_user Optionally specify that users encountered in the request POID are not to be validated. Defaults to validating the users and failing the POID import if users do not exist. If the `-g` option is given, it is used.

-gmt Optionally specify that all requests are to be converted from GMT format.

-dont_cm_pend Optionally specify that at the *end* of the import, request User Pending Lists are not to be recalculated.

-l `-l <log file>`

Optionally specify that any verbose output from `pdiff` is to be placed in the log file `<log file>`.

-g `-g <proxy_user>`

Creates a quest/proxy user account for the username specified. This is useful if the POID being loaded references an unknown username.

-f `-f <pdiff_file> <pdiff_file> ...`

Optionally specify that the requests are to be imported from the PDIFF file(s) `<pdiff_file>`. Defaults to reading from the operating system standard input.

CAUTION! If a PDIFF file was created by dumping requests using a filter, such as the request identifier, there is no guarantee that related parent requests that do not match the filter will also have been dumped into the PDIFF file.

Submitting PDIFF Commands from a Parameter File

Syntax `pdiff -params <params-file>`

Specifies a parameter file containing `pdiff` commands to be sequentially executed.

Limitations

- Request-related revised baselines are not supported.
- Requests are exported with user role delegations relation to the current process model (control plan).
- When importing design parts, the target product must not contain any design parts other than the root one. The product process model must contain design part categories and held requests are not supported for import/export.
- Import/export of items and external requests is not supported.
- Cross-product relationships of any kind are not supported for import/export.
- When loading an import PDIFF file into Dimensions Change Management, you must ensure that only those attributes that are defined in the process model as *multiline* contain the carriage return character `"\n"` in their values. If this condition is not met, fatal errors are encountered when browsing an imported request. (An attribute which is given an edit box height of greater than zero in the process model is regarded as a *multiline* attribute.)

Chapter 18

PDIFF Syntax

Required statements are in **bold**.

```
PDIFF :c==
   <PDIFF Header>
   <POIDS>

<PDIFF Header> :==
   #PDIFF<version number>
   #DATE <DD-MMM-YYYY> <HH:MM:SS>
   #CREATED BY <name>
   #RESERVED
   #RESERVED
   #RESERVED
   #RESERVED
   #RESERVED
   #RESERVED
   #RESERVED

<POIDS> :==
   {<CHDOC POID>                    |
   <PART POID>                      |
   <VALID_SET POID>                 |
   <DELEGATEE POID>                 |
   <CHDOC_BROWSE_TEMPLATE POID>}
   [<POIDS>]

<CHDOC POID> :==
   OBJECT CHDOC
     <attributes>
     <composite chdoc attributes>
   END_OBJECT

<attributes> :==
   <system attribute> | <user attribute>
   [<attributes>]

<system attribute> :==
   $DOC_SEQ                   <number>          |
   $IDENTIFIER                <string>          |
   $ARCHIVED                  { Y | N }         |
   $TYPE                      <string>          |
   $CREATE_DATE               <date>            |
   $CURRENT_STATUS            <string>          |
   $ORIGINATOR                <string>          |
   $LIFECYCLE_ID              <string>          |
   $LC_SEQ                    <number>          |
   $DETAILED_DESC             <long string>     |
```

```
   $ACTION_DESC
      .CURRENT                  <long string>
      .ALL_PREV                 <long string>     |
   $AD_HISTORY
      .ACTION_NO                <number>
      .FILE                     \F<filename>
   $ATTACHMENT
         .FILE                  <filename>
         .USER_FILE             <filename>
         .USE                   <string>
         .COMMENT               <string>
         .DATE                  <date>
   $UPDATE_DATE                  <date>
   $OWNER_SITE                   <string>
   $REFERENCE_ONLY               <string>
   $LOCKED                       <string>
   $DELEGATED_OWNERSITE          <string>

<user attribute> :==
   <attribute variable name>    <attribute value>

<attribute variable name> :==
   <white space free string>

<attribute value> :==
   <string>

<composite chdoc attributes> :==
   <action> | <update> | <relation> | <delegate>
   {<composite chdoc attributes>}

<action> :==
   $ACTION
      .ACTION_NO                <number>
      .DATE                     <date>
      .USER                     <string>
      .STATE                    <string>
      .PHASE                    <string>
      .TYPE                     A
      .COMMENT                  <max 80 string>

<update> :==
   $UPDATE
      .ACTION_NO                <number>
      .DATE                     <date>
      .USER                     <string>
      .STATE                    <string>
      .PHASE                    <string>
      .TYPE                     {U|D}
      .COMMENT                  <max 80 string>

<relation> :==
   $RELATE
      .ACTION_NO                <number>
      .TYPE                     <string>
      .OBJECT                   CHDOC:<chdoc id>          |
                                PARENT_CHDOC: <chdoc id>  |
                                PART:<part spec>          |
```

```
                                    ITEM:<item spec>          |
                                    BASELINE:<baseline spec>
      .DATE                         <date>
      .USER                         <string>

<delegate> :==
   $DELEGATE
      .USER                         <string>
      .ROLE                         <string>
      .CAPABILITY                   {S|P|L}
      .OPERATION                    {ADD|REPLACE}

<attribute history> :==
   $ATTRIBUTE_HISTORY
      .ACTION_NO                    <number>
      <attribute info list>

<attribute info list> :==
   <attribute info> | <attribute info list>

<attribute info> :==
   .ATTRIBUTE         <attr_no>:<attr seq>:<string>

<attribute update history> :==
   $ATTRIBUTE_UPDATE_HISTORY
      .ACTION_NO                    <number>
      .ATTR_NO                      <number>
      .SEQ_NO                       <number>
      .DATE                         <date>
      .USER                         <string>
      .STATUS                       <string>
      .PREV_VALUE                   <string>
      .NEW_VALUE                    <string>

<PART POID> :==
   OBJECT PART
      <attributes>
      <relate_history>
   END_OBJECT

<attributes> :==
   <system part attribute> | <user attribute> | <attributes>

<composite part attributes> :==
   <part relate>                 |
   <part relate history>         |
   <composite part attributes>

<system part attributes> :==
   $PRODUCT_ID                   <string>          |
   $PART_ID                      <string>          |
   $VARIANT                      <string>          |
   $CURRENT_PCS                  <string>          |
   $PCS                          <string>          |
   $STATUS                       <string>          |
   $DESCRIPTION                  <string>          |
   $CREATE_DATE                  <string>          |
   $CATEGORY                     <string>          |
   $USER                         <string>          |
```

```
   $PREV_PART                    <part spec>

<part relate> :==
   $RELATE
      .TYPE                      {BREAKDOWN|USAGE}
      .OBJECT                    PARENT_PART:<part spec>|

PART:<part spec>
      .DATE                      <date>
      [.USER                     <string>]

<part relate history> :==
   $RELATE_HISTORY
      .TYPE                      {BREAKDOWN|USAGE}
      .OBJECT                    <part spec>
      .DATE                      <date>

<VALID SET POID> :==
   OBJECT VALID_SET
      $PRODUCT_ID                <string>
      $VALID_SET_NAME            <string>
      $NO_COLUMNS                <number>
      $DESCRIPTION               <string>
      <valid set values>
   END_OBJECT

   <valid set values> :==
      <value set value> [<valid set values>]

<valid set value> :==
   $VALUE
      .DISP_ORDER                <number>
      .COL1                      <string>
      .COL2                      <string>
      .COL3                      <string>
      .COL4                      <string>
      .COL5                      <string>
      .COL6                      <string>
      .COL7                      <string>
      .COL8                      <string>

<CHDOC_BROWSE_TEMPLATES POID> :==
   OBJECT CHDOC_BROWSE_TEMPLATES
      $PRODUCT_ID   <string>   <browse template definitions>
   END_OBJECT

<ITEM_BROWSE_TEMPLATES POID> :==
   OBJECT ITEM_BROWSE_TEMPLATES
      $PRODUCT_ID   <string>   <browse template definitions>
   END_OBJECT

<browse template definitions> :==

<browse template definition> [<browse template definitions>]

<browse template definition>  :==
   $TEMPLATE_DEF
      .IDENTIFIER                <string>
      .REVISION                  <string>
```

```
      .FILE                       \F<filename>

<VALID SET POID> :==
   OBJECT DELEGATEE
      $PRODUCT_ID                 <string>
      <delegatees user_role_assignments>
   END_OBJECT

<delegatees user_role_assignments> :==
   <delegatee user_role assignment>
   [<delegatees user_role_assignments>]

<delegatee user_role_assignment> :==
   <delegatee> | <user_role_assignment>

<delegatee> :==
   $DELEGATEE
      .ROLE                       <string>
      .USER                       <string>
      .CAPABILITY                 [P|S|L]
      .PART_ID                    <string>
      .PART_VARIANT               <string>
      .WORKSET_ID                 <string>

<user_role_assignment> :==
   $USER_ROLE_ASSIGNMENT
      .ROLE                       <string>
      .USER                       <string>
      .CAPABILITY                 [P|S|L]
      .PART_ID                    <string>
      .PART_VARIANT               <string>
      .WORKSET_ID                 <string>

<chdoc id> :==
   <string>

<part spec> :==
   [<product id>]:<part id>.<variant>;<pcs>

<date> :==
  <dd-mon-yyyy hh24:mi:ss>

<string> :==
   max 2000 character text, Dimensions limits

<long string> :==
   greater than 2000 character text
```

Part 4

Network Administration

Introducing Network Administration	251
Dimensions CM Network Guide	269
Network Applications	275
Network Administration Command Line Interface	287

Chapter 19

Introducing Network Administration

Overview	252
Network Architecture	253
Physical Node Types	253
Logical Nodes	254
Code Page	256
Byte Order Mark	258
Content Encoding	259
File System Registration for Remote Nodes	261
Item Operations in a Dimensions Network	261
Setting Up Physical Nodes	264
Dimensions Network Node Performance Issues	264
Default Dimensions TCP/IP Port Allocations	265

Overview

Dimensions CM is a client/server tool that enables users on UNIX, Windows, and z/OS[1] platforms to share working methods, status information, and files in a consistent and controlled manner. Dimensions CM integrates file access and change authority across all platforms where it is installed, without requiring a specialized file system such as NFS. Dimensions CM operates across common network protocols such as TCP/IP.

For example, the Dimensions CM central database server can be installed on a Sun SPARC station (running Solaris) together with the RDBMS (Relational Database Management Systems) database and the Dimensions item libraries.

Concurrent client Dimensions CM installations can be made on other machines in the network, such as:

- A PC running Windows.
- An HP 9000 running HP-UX.
- An IBM RISC System/6000 running AIX.
- A Sun SPARC running Solaris.
- An IBM mainframe running z/OS MVS.

Each of these Dimensions CM clients can successfully query and update the central database repository through the database server.

Each Dimensions CM repository contains information about the network nodes that are able to access the Dimensions CM file systems (item libraries) as clients, servers, or both. This information is specified using either the Administration Console or the Network Administration command line interface, which enables you to:

- Register operating systems, physical and logical network nodes, and resident software.
- Register the connection of network clients and servers.
- Register client nodes so that they may access a Dimensions CM Server node.
- Set up network nodes that may be used for item replication purposes. Set up is used by the separately licensed Dimensions Replicator component for replication of items, baselines, and requests and their metadata from one repository to another remote repository.

1. The term z/OS in this manual covers both the z/OS and OS/390 operating systems.

Network Architecture

For details about the Dimensions CM architecture, see the Dimensions CM online help.

Physical Node Types

A Dimensions CM physical network node can be one of the following types (in order of increasing complexity).

Client Node

A client node is a machine having only client Dimensions CM components (for example, `dmcli`, `dmpmcli`, and Dimensions desktop client [Windows only]). This directly corresponds to a Dimensions CM client-only installation.

Library Server Node

General

An item library, or area, node is a machine containing Dimensions CM library server components. It is capable of hosting an item library or a work area (a user file area) and a deployment area (what used to be called a build area). This directly corresponds to a Dimensions CM listener/agent-only installation.

Library Cache Area

NOTE See also Appendix F, "Using and Configuring Library Cache Areas" and the Dimensions CM online help for details about library cache area nodes.

An item library node also supports the optional "library cache areas" feature. The purpose of a library cache area is to improve Dimensions CM file transfer performance. It does this by caching a file's contents onto the library cache area node that is "closer" (in networking terms) to the "work" node (the node upon which you are working) than the "work" node is to a more "distant" Dimensions CM item library node (the centralized node upon which the Dimensions CM item libraries are stored). This avoids having to repeatedly transfer a file between your work node and the item library node through a potentially slow connection.

Library cache areas are set up by the Dimensions CM administrator to improve performance for sites that regularly access files that otherwise take a long time to be retrieved. Although the first requester of a file sees no performance improvement, Dimensions CM caches that file in the library cache area, and every subsequent request accesses the library cache copy rather than making another request for the original. For example, if developers connect to a remote Dimensions CM server through a virtual private network (VPN) across a relatively slow wide area network (WAN) connection—as compared to the local network on which developers' PCs are located—then a library cache area can be set up on the local network in order to reduce network traffic across the WAN connection.

The creation and management of such areas is done using the following `dmcli` commands:

- `scws` (to use library cache areas)
- `clca` (to create a library cache area definition)
- `ulca` (to update a library cache area definition)
- `llca` (list library cache area definitions)
- `rlca` (remove a library cache area definition)

See the *Command-Line Reference* for details.

NOTE The library cache area optimization is not used if any of the following conditions are true:

- A directory item is being gotten.
- Item header substitution is on.
- The library file is compressed.

CAUSTION! A library cache area should only be set up in the situation detailed above, not for every installation. In fact, if the Dimensions CM administrator set up a library cache area in those situations where delayed file access was not the case, the library cache area could actually decrease performance because some file access operations require two write operations instead of one.

Application Server Node

An application server node is a machine having Dimensions CM "business logic" server components (for example, `dmlsnr` + `dmpool` + dmappsrv + dmlibsrv and all the components supported by a client and item library node). It is capable of connecting to an RDBMS, item library servers, and area nodes. This directly corresponds to a full Dimensions CM server installation. Note that such application server nodes can also host item libraries and areas. In fact, this is the default set up.

Logical Nodes

IMPORTANT!

- Logical node names cannot be used for item library definitions.
- When you create a logical node name, you must specify a network node connection to a physical node name. The two node names must not be the same. Failure to specify a network node connection causes Dimensions CM to flag errors during attempted network operations.

One of the key Dimensions CM features is the ability to define a *logical (network) node*. Logical nodes are user defined aliases that help to provide a layer of abstraction on top of

your physical network. The node name can be used to prefix paths to files to control where a file is written, thus allowing one or more remote nodes functionality.

Logical nodes do this by allowing an association between the node name and a physical machine name by way of a specific connection. Connection information specifies details such as:

- The code page to use for the connection (see "Code Page" on page 256).
- The operating system and file system for the node (see "File System Registration for Remote Nodes" on page 261).

The node name can be used to prefix paths to files to control where a file is written, thus allowing remote node functionality. A remote mainframe file can be specified as follows:

```
MAINFRAME1::MERVK.WORK.C(MOD12)
```

where `MAINFRAME1` is the logical node name which might map onto a physical machine called `S390` using port number `1096`.

Once a logical node is defined, additional information about that node, or the connection to it, can be stored. It is possible to have multiple logical nodes referring to a single physical node using different connection information. This makes it possible to have more control over the networking environment. For example, each project team may want to connect, through a different connection, to the same physical node to allow for load balancing or enhanced security. Another example might be that one logical node on a machine is used as an EBCDIC environment, by default, whereas another may be ASCII by default.

NOTE One or more logical nodes can be associated with the same physical node; but you *cannot* associate the same logical node definition to more than one physical node.

Logical nodes are defined with the Administration Console (see Chapter 21, "Network Applications") or the Network Node command, see "Network Node" on page 297.

If your project/stream working location is on a z/OS mainframe, you always see the logical node name. If your project working location is on a Windows or UNIX machine, the way your Dimensions CM system has been set up determines whether or not you can see the logical node name.

Examples of project/stream working location names:

```
c:\mydir
MY_PC_NODE::c:\mydir
MY_PC_NODE::
MY_UNIX_NODE::/root/me
MY_MAINFRAME::MERVK.WORK
MY_WORK_AREA::
MY_WORK_AREA::src
```

Code Page

What is a Code Page?

Dimensions CM records the code page used by an item when it was last updated in the item library.

A code page in Dimensions CM defines the method of encoding characters. It encompasses both the different ways characters are encoded on various 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*— a code page assignment is mandatory for defining the setting for the connection between the database server and the logical node on which the user file resides. This code page becomes the default value, and this value is assigned to all applicable item operations unless it is overridden as described below.

  Connections are defined using the Administration Console (see Chapter 21, "Network Applications") or the Network Administration command line interface (see Chapter 22, "Network Administration Command Line Interface").

- *An individual item*— a code page is specified by means of an explicit setting on the command that *stores* (creates or updates) the item in the library, for example, when you create, check in, or update/revise an item. This code page value overrides the default value described above. You can also update an item's existing code page (if any) when you check in or update an item.

NOTE Code page *assignment* refers to the point at which an item revision is stored in the library by commands such as create, get, check in, check out, update/revise etc. This is not the same as code page *translation* that results from moving files between platforms that have different code page assignments.

We recommend using the default value of the code page rather than explicitly setting it.

Code Page Conversion Between Platforms

When Dimensions moves files between platforms (for example, when checking out from a mainframe to a PC), code page conversion depends on the nature of the files being moved:

- *Text files*—Dimensions automatically performs any necessary code page conversion between EBCDIC and ASCII or vice versa.

  Code page 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 in order for it to display correctly.

- *Binary files*—Dimensions does *not* perform any code page conversions.

When you first create an item, Dimensions stores it in the library using the code page associated with the item. Subsequently, Dimensions only performs code page conversion for items *moving between platforms* when they are checked out or gotten, not when they are checked in or updated/revised. Hence it is possible for different versions of an item to be stored using different code pages. For example, if you create an item from a user file

on z/OS, it is stored in EBCDIC (as long as you do not override the code page for the logical node). If you then check the item out to a PC, it is converted to ASCII during the move. If you subsequently check it back in, Dimensions stores the new version in ASCII.

NOTE Code page *translation* results from moving files between platforms that have differing code page assignments. This is not the same as code page *assignment* at the point at which an item revision is stored in the library by commands such as create, get, check in, check out, update/revise etc.

Code Page Parameters

The code page parameters are:

Code Page	Type
00037	EBCDIC USA/Canada
00273	EBCDIC Germany
00277	EBCDIC Denmark/Norway
00278	EBCDIC Finland/Sweden
00280	EBCDIC Italy
00284	EBCDIC Spain/Latin America
00285	UK English (EBCDIC)
00297	EBCDIC France
00437	OEM USA
00500	EBCDIC Multilingual
00819	ISO 8859-1 - Latin Alphabet 1
00850	OEM Multilingual Latin 1
00858	OEM Multilingual Latin 1 (Euro)
00871	EBCDIC Iceland
00923	ISO 8859-15 - Latin Alphabet 1 (Euro)
00924	EBCDIC ISO 8859-15 - Latin Alphabet 1 (Euro)
01047	EBCDIC Latin 1/Open Systems
01140	EBCDIC USA/Canada (Euro)
01141	EBCDIC Germany (Euro)
01142	EBCDIC Denmark/Norway (Euro)
01143	EBCDIC Finland/Sweden (Euro)
01144	EBCDIC Italy (Euro)
01145	EBCDIC Spain/Latin America (Euro)
01146	EBCDIC UK (Euro)
01147	EBCDIC France (Euro)
01148	EBCDIC Multilingual (Euro)

Code Page	Type
01149	EBCDIC Iceland (Euro)

NOTE You can omit the leading zeros.

Byte Order Mark

What is a Byte Order Mark (BOM)?

Because Unicode plain text is often encoded as a sequence of 16-bit code values, it is sensitive to the byte ordering used when the text was written. A byte order mark (BOM) is used to prefix a Unicode plain text file to indicate the byte ordering. A BOM is not a control character that selects the byte order of the text, it simply informs an application receiving the file that the file is byte ordered.

Ideally, all Unicode text would follow only one set of byte-ordering rules. This is not possible, however, because microprocessors differ in the placement of the least significant byte: Intel and MIPS processors position the least significant byte first; whereas, Motorola processors (and all byte-reversed Unicode files) position it last. With only a single set of byte-ordering rules, users of one type of microprocessor would be forced to swap the byte order every time a plain text file is read from or written to, even if the file is never transferred to another system based on a different microprocessor.

The preferred place to specify byte order is in a file header, but text files do not have headers. Therefore, Unicode has defined BOMs to indicate Unicode and non-Unicode text files respectively. They are mirror byte-images of each other.

BOM Implementation in Dimensions

During item creation, for example, from a `CI` or `RI` command:

1 If the request (item type being created) is `TEXT`, and the command does not provide a `"/CONTENT_ENCODING="` parameter to work with, then the BOM is examined. This occurs on the user node where the item is coming from. Dimensions CM currently checks the following BOMs:

UTF-8	EF BB BF
UTF-16 Big Endian	FE FF
UTF-16 Little Endian	FF FE
UTF-32 Big Endian	00 00 FE FF
UTF-32 Little Endian	FF FE 00 00

2 What happens then is exactly the same as if you had specified this in the `/content_encoding` parameter.

When /content_encoding is specified, or deduced from the BOM, the following occurs:

1. The item is checked in BINARY. The item in the Dimensions CM library is *identical* to the original file. No CR/LF or codepage conversions are made.
2. When the item is gotten or checked out later, the CR/LF conversion is handled in a completely different way to that in which Dimensions CM treats normal TEXT items.

   The conversion is performed in an "encoding-safe" way on the destination node of the get or check out operation. For example, a UTF-16 file containing double-byte CR+LF pairs that came from Windows Notepad, are altered to contain UNIX style LF (in UTF-16) if gotten or checked out to a UNIX node. (In other words "00 0D 00 0A" becomes "00 0A".) This amended client-side behavior for encoded text files has been added to the Dimensions CM Java clients as well. (The previous server-side CR/LF conversion would have worked on a byte-by-byte basis, and corrupted the UTF-16 structure.)
3. As well as new CR/LF handling, there is also a completely new header expansion engine used on the server that is encoding-aware. That means that if the UTF-16 file contains the UTF-16 equivalent of %PI%, it is replaced (in UTF-16) with the value of this field.
4. The above header expansion mechanism now works in harmony with any codepage conversion necessary for EBCDIC clients such as mainframes. So you should be able to have a UTF-32 file containing header variables, and get it or check it out with expansion to a mainframe node, and be able to read the result.

NOTES

- Currently there is no cross-encoding translation. An encoded file in UTF-16 remains that way. There is no way of getting it or checking it out in UTF-32BE.
- An item revision already checked in to Dimensions CM is not affected by this. BOMs are not examined when items are gotten or checked out—only during item creation.
- The BOM detection routines are also employed where detailed descriptions for Dimensions CM requests (change documents) are provided. This enables the data to be converted correctly into UTF-8 for storage in the Dimensions CM database.

Content Encoding

General

Content encoding can be appended to item-related Dimensions CM dmcli commands (such as ci, ei, mi, ri, ui, and upload) through a /content_encoding optional qualifier, see the *Command-Line Reference* for a full list of item-related commands.

By utilizing the /content_encoding qualifier, you are declaring that the item in question is in the encoding specified. This fact is stored in the Dimensions CM repository. The item is stored in binary so as to preserve the encoded bytes.

The purpose of encoding is to control how textual manipulation of the item occurs within Dimensions CM, with regard to multibyte characters. It enables operations like CR/LF end-

of-line substitution and header variable expansion to occur in files that have four (UTF-32) or two (UTF-16) bytes per character.

Mainframe Restrictions

For Dimensions 12.1 there are currently some mainframe restrictions, as detailed below:

- Content encoding does not affect the actual code-page translation chosen when the file is transferred to an EBCDIC system such as a mainframe. In this situation, a two-stage conversion is done through the single byte ISO-8859-1 system.
- Only single-byte EBCDIC text files, in the code page specified in the Dimensions network connection structure, are supported. This means that you can use UTF-16 and UTF-32 files with MVS mainframes, but only when the set of actual characters employed in the file falls within the normal ascii/ebcdic range. Obviously with a much smaller range, 8 bit EBCDIC cannot represent all unicode characters.

Valid Encodings

The valid encodings are:

646	ARABIC	ASCII
BIG5	CP1250	CP1251
CP1252	CP1253	CP1254
CP1255	CP1256	CP1257
CP1258	CP936	CSBIG5
CYRILLIC	GBK	GREEK
HEBREW	IBM-1089_P100-1995	IBM-5346_P100-1998
IBM-5347_P100-1998	IBM-5348_P100-1997	IBM-5349_P100-1998
IBM-5350_P100-1998	IBM-5354_P100-1998	IBM-9005_X100-2005
IBM-912_P100-1995	IBM-913_P100-2000	IBM-914_P100-1995
IBM-915_P100-1995	IBM-916_P100-1995	IBM-920_P100-1995
IBM-923	IBM-923_P100-1998	IBM-943_P15A-2003
IBM-9447_P100-2002	IBM-9449_P100-2002	ISO-8859-1
ISO-8859-2	ISO-8859-3	ISO-8859-4
ISO-8859-5	ISO-8859-6	ISO-8859-7
ISO-8859-8	ISO-8859-9	ISO-8859-15
ISO-IR-100	ISO-IR-101	ISO-IR-109
ISO-IR-110	ISO-IR-126	ISO-IR-127
ISO-IR-138	ISO-IR-144	ISO-IR-148
ISO-IR-149	KOREAN	LATIN0
LATIN1	LATIN2	LATIN3
LATIN4	LATIN5	MS1250

MS1251	MS1252	MS1253
MS1254	MS1255	MS1256
MS1257	MS1258	MS-874
SHIFT_JIS	SJIS	TIS-620
US-ASCII	UTF-16	UTF-16BE
UTF-16LE	UTF-32	UTF-32BE
UTF-32LE	UTF-8	WINDOWS-1200
WINDOWS-1201	WINDOWS-1250	WINDOWS-1251
WINDOWS-1252	WINDOWS-1253	WINDOWS-1254
WINDOWS-1255	WINDOWS-1256	WINDOWS-1256-2000
WINDOWS-1257	WINDOWS-1258	WINDOWS-20127
WINDOWS-65001	WINDOWS-874	WINDOWS-874-2000
WINDOWS-932	WINDOWS-936	WINDOWS-936-2000
WINDOWS-949	WINDOWS-949-2000	WINDOWS-950
WINDOWS-950-2000	X-UTF-16BE	X-UTF-16LE

File System Registration for Remote Nodes

The native file system in use by the operating system on a remote node can be registered and subsequently managed. This enables you to ensure that only unique file systems can be created, and that once defined that they cannot be deleted unless these definitions are no longer in use.

The file system name is the name of the new file system that you want to register. Dimensions CM does not validate this entry.

Item Operations in a Dimensions Network

In a homogeneous network environment, where both the Dimensions database server and its clients are using the same or compatible operating system types, no special considerations need to be addressed. The authorized user can use Dimensions without restriction from any Dimensions node. All items can be accessed from any client or server node, as the operating system recognizes and manages all the various file types consistently.

In a heterogeneous network environment, certain limitations may exist regarding the types of files and the manner in which they are handled by the operating system. Files types that are unique to a particular operating system cannot be transferred to a different operating system.

Privileged and User Command Modes

The commands submitted by a Dimensions CM client to a physical node can be in either of two modes: **privileged** and **user**.

Privileged commands. These run on the item library or application server physical node with privileges that enable files to be written and read from the Dimensions CM item libraries.

User commands. Dimensions CM initially attempts to access the item library or application server physical node using the username/password combination of the client physical node login. If that username/password combination is invalid for the item library or application server physical node, the user must either:

- Provide the valid username/password when prompted in the Dimensions CM desktop client.
- Use the Dimensions CM `AUTH` command to provide the valid username/password when using the command-line interface.

In user mode, ordinary user privileges are used when creating directories or setting file protections such as with the Dimensions function DPL (Define Product Libraries), and Dimensions users accessing the node through Dimensions Network may only access and create files as if they were logged into that node directly.

When three nodes are involved, for example, from Node A, a client node, there is a request to get an item file residing in a Dimensions CM library on Node B (which therefore must be an item library or application server node), placing the file on Node C (this third node must also be a item library or application server node, not just a client node)—the syntax if Node C is UNIX is:

```
fi <prod-id>:<item-id>.<var>-<type>;<rev> /user_file="C::<dir>/<file>"
```

Directory items and delta library items can also be used in this way.

NOTE To access Node C, the user is prompted by the desktop client to enter the username/password. Alternatively, the user can use the Dimensions CM `AUTH` command to provide these details or a work area could be defined for the remote working location that includes the username and password to be used when accessing the area. When accessing files on such a work area using the `<work area name>::<path to file inside work area>` syntax, the user is not prompted for a password.

Dimensions Network Limitations

The success of a multi-node item operation depends on a number of factors:

Operating system combination:	UNIX, Windows, z/OS
Node types:	client, item library, or application server node
File type:	Text, Binary, etc.
User interface:	Command-line, desktop client, etc.
Library type:	Standard, delta, Version Manager-delta

The `node::filepath` or `work_area::relative_filepath` syntax is applicable to all platforms in Dimensions CM, including Windows.

Where the node is Windows, the UNC path syntax (for example, `\\node\sharename\filepath`) used in earlier versions of Dimensions CM can still be specified. If using UNC syntax, ensure that your particular combination of operating system, node type, library type, file type, and user interface on client node, server node, library node, and target node functions as expected.

Directory items cannot be stored in a delta library.

Dimensions CM does not use revisions when it stores files under delta library systems; therefore, the delta library system's history mechanism does not show meaningful information.

IMPORTANT! Delta libraries are not supported on z/OS.

The following inter-platform item transfers are supported:

- All UNIX items (apart from directory items) can be transferred to and from a Windows or z/OS node.
- All Windows items can be transferred to and from a UNIX or z/OS node.
- All z/OS items can be transferred to a UNIX or Windows node.

If `/keep` is not specified for check in (return) Item or update item, then the normal action is to remove the user file if the operation is successful. This does not happen under the following conditions:

- When the item is a directory item.
- When the user does not have permission to delete the file.
- When the user is on a z/OS node.

Using Network Attached Storage Devices for Item Libraries

To set up and access a Dimensions CM item library located on a Windows Network Access Storage (NAS) device connected through a network share, you need a set of user credentials to log in to the network share. This is achieved, as detailed below, by adding a user login configuration symbol to the `dm.cfg` file located on the Dimensions CM *server* (*not the Dimensions CM library server*) and registering that user's password using the `dmpasswd` utility. After these user login credentials are registered, Dimensions CM authenticates that user whenever the NAS item library is accessed.

Example 1 To set up a user's login credentials if accessing a *specific* item library on the NAS device (that is, an item library that is *specific* to a particular product and item type):

1 Edit the `dm.cfg` file on the Dimensions CM server to add an entry of the form:

```
DM_LIBRARY_USER_<product id>_<item type> <username>
```

For example, for a product called `QLARIUS`, a specific item type called `SRC`, and a network share *domain* username of `acme\user1`, the symbol should be set to:

```
DM_LIBRARY_USER_QLARIUS_SRC acme\user1
```

2 Run dmpasswd to register that users password, as follows:

```
dmpasswd acme\user1 -add -pwd <password>
```

For further details of the dmpasswd utility, see the "Standalone Dimensions Utilities" chapter in the *Command-Line Reference*.

Example 2 To set up a user's login credentials if *all* item libraries for *all* item types for a given product reside on the same NAS device (that is, all item libraries that are *specific* to a particular product):

1 Edit the dm.cfg file on the Dimensions CM server to add an entry of the form:

```
DM_LIBRARY_USER_<product id> <username>
```

For example, for a product called QLARIUS and a network share *domain* username acme\user1, the symbol should be set to:

```
DM_LIBRARY_USER_QLARIUS acme\user1
```

2 Run dmpasswd to register that users password, as follows:

```
dmpasswd acme\user1 -add -pwd <password>
```

Example 3 To set up a user's login credentials if *all* item libraries in the base database happen to need the same set of credentials (that is, the item libraries are *not specific* to a particular product or item type):

1 Edit the dm.cfg file on the Dimensions CM server to add an entry of the form:

```
DM_LIBRARY_USER <username>
```

For example, for a product called QLARIUS and a network share *local* username of george, the symbol should be set to:

```
DM_LIBRARY_USER george
```

2 Run dmpasswd to register that users password, as follows:

```
dmpasswd george -add -pwd <password>
```

Setting Up Physical Nodes

Physical nodes are setup using the Administration Console or the Network Administration command line interface, introduced in Chapter 20, "Dimensions CM Network Guide" and described in full in Chapter 21, "Network Applications" and Chapter 22, "Network Administration Command Line Interface".

Dimensions Network Node Performance Issues

In a Dimensions CM Network, it is possible to configure the network to take advantage of the computing resources available. In addition to providing networking facilities that permit operations across a heterogeneous environment, a Dimensions CM Network is able to spread the processing load in various node configurations.

We recommend that the RDBMS processes should execute on the fastest node in the network, and if possible that node should have no Dimensions CM logins. Also, the operating system parameters must be optimized with as much RAM as possible for each Dimensions CM application server node in the network. If a single-user workstation is used on the network, the working set sizes can be significantly increased to reduce paging.

Performance can be influenced by many other factors outside the scope of this document.

See, also, "Library Cache Area" on page 253 and Appendix F, "Using and Configuring Library Cache Areas" concerning the use of library cache areas to increase network performance.

Default Dimensions TCP/IP Port Allocations

Dimensions Server or Agents

The Dimensions CM server requires a designated TCP/IP port to allow incoming connections. Any Dimensions CM agent nodes (used for deployment areas, library cache, work areas, remote working locations, etc) also needs to allow connections to that port.

By default this designated TCP/IP port is 671, but see "Setting Non-Default Service Name and Port Number" on page 502 if you want to allocate a different value.

Dimensions Web Applications

The Dimensions CM web applications (Dimensions CM web client and Administration Console) make TCP/IP connections as follows:

- Client TCP/IP Connections

  The applications make TCP/IP client connections to the Dimensions CM listeners to which they are instructed to connect. By default this is TCP/IP port 671. To use another port, see "Reassigning Default Web Client Connection Parameters" on page 31 and "Setting Non-Default Service Name and Port Number" on page 502.

- Sever TCP/IP Connections

  The applications do not listen on any TCP/IP server ports. Instead, they get passed the TCP/IP server connections from Common Tomcat.

  These Tomcat Connector Server ports, are configured within the Tomcat configuration file (see "Introduction" on page 26 for a definition of `<Tomcat-Root>`):

  - Windows

    `<Tomcat-Root>\conf\server.xml`

NOTE This file is in `<LF>` UNIX type format (not `<LF><CR>` format). The editor you use on Windows must be able to work with such files. Wordpad, Word, and textpad work with such files, but Notepad simply presents the file as a single line.

  - UNIX

    `<Tomcat-Root>/conf/server.xml`

Within the `server.xml` file, the port allocations can be found by searching for the string `'port='` (this finds strings such as port="8080", but valid port numbers can be any value between 1 and 32767).

The default TCP/IP port allocations are listed in the following table.

Common Tomcat Connector Server Port	Default TCP/IP Port Number	Comments
Non Secure Sockets Layer (SSL) HTTP/1.1 Port	8080	The server port number that is requested by the Dimensions installer is the non-SSL "HTTP/1.1" port. This accepts incoming connections from web browsers and it defaults to `8080` unless another value is specified during installation. **NOTE** This port needs opening on a firewall to allow access from remote web browsers to the local Tomcat.
"SHUTDOWN" Port	8005	The "SHUTDOWN" port enables Tomcat to be shutdown cleanly. In Dimensions 14.1, this has the value `8005` (although, in some versions of Dimensions 9.1, it was 8405). **NOTE** This port should not need to be opened by a firewall, since it is only used locally for shutdown. Only the HTTP and HTTPS port numbers should need opening on a firewall to allow access from remote web browsers to the local Tomcat.
SSL HTTP/1.1 Port	8443	If "HTTPS" has been enabled for connections from web browsers, the SSL HTTP/1.1 Port is usually set to a default value of `8443`, but the actual value may differ to this depending on customer preferences. **NOTE** This port needs opening on a firewall to allow access from remote web browsers to the local Tomcat.
AJP 1.3 Port	8409	The "AJP/1.3" port enables Tomcat to be integrated with external web servers like IIS and Apache httpd. **NOTE** This port should not need to be opened by a firewall, since it is only used locally for web server integrations.
Proxied HTTP/1.1	8082	For details about using this port, see the Tomcat proxy documentation.

Remote Serena-Supplied Runtime or Oracle Enterprise RDBMS

If the Serena-Supplied Runtime RDBMS or Oracle Enterprise RDBMS is remote from the Dimensions CM server, the RDBMS server additionally requires TCP/IP port 1521 to allow incoming connections.

Chapter 20

Dimensions CM Network Guide

Overview	270
Specifying Operating Systems	270
Specifying Network Nodes	270
Specifying Network Client and Server Connections	271
Specifying Code Pages	271
Specifying File Systems for Remote Nodes	271
Registering Resident Software	272
Dimensions CM Listener	272

Overview

The Administration Console and the Network Administration command line interface enables you to configure an installed Dimensions® CM Network. A network typically consists of physical nodes running different operating systems, which can be networked together in a variety of ways. The Dimensions CM database is installed on one of the nodes on the network. This database must be directly accessible by each of the other nodes. If a Dimensions CM application server or item library node is being used, Dimensions CM libraries may be located on the node and it must be made accessible to any other node in the network.

The sequence of operations required to set up the Dimensions CM Network, and a brief description of each command, is included in the following sections. For full details of the Network Administration command line interface, see Chapter 22, "Network Administration Command Line Interface".

Specifying Operating Systems

The Operating System command enables you to specify what operating systems are part of the network and the case convention that is required by an operating system for user names and filenames. The case required for filenames is applicable only for mixed operating systems—for example, the file case fetched to a UNIX machine from libraries on a Windows machine, can be used by other tools such as compilers. Check the importance of case-related files with any related projects before changing data.

For full details about this command and its qualifiers, see "Operating System" on page 301.

Specifying Network Nodes

The Network Node command enables you to specify the Dimensions CM Network physical nodes, their objects and protocols, databases, logical node associations, contacts, and resident software. Use this command to define all the physical nodes that are part of the Dimensions CM Network. See "Physical Node Types" on page 253 and "Logical Nodes" on page 254 for a discussion of the various node types. Network objects and protocols are used to set up communication among two or more nodes.

When the protocol and the network object are identical throughout the network, the node name may be defaulted using an asterisk (*). The same network object name may be used for more than one node even when the protocol is different. For TCP/IP, the network object name being used must be defined previously in the services file. The network object is used by the networking software to communicate between nodes using the TCP/IP protocol.

For full details about this command and its qualifiers, see "Network Node" on page 297.

Defining Logical Nodes

The Network Node command includes a qualifier that enables you to define (and update) the logical nodes that are part of a Dimensions CM Network. Logical nodes are user-defined aliases that help you to provide a layer of abstraction on top of your physical network. For more details, see "Logical Nodes" on page 254.

Specifying Network Client and Server Connections

The Network Node Connections command and the Network Object command enable you to establish client access to nodes where item libraries or user files may reside. For each client node, you specify all the server nodes that a client node may access and the network object and protocol that are used. If all nodes can be accessed from any node, then the client node and server node fields may be defaulted by using an asterisk (*) and specifying the network object name and protocol.

For full details about these commands and their qualifiers, see "Network Node Connections" on page 298 and "Network Object" on page 300.

Specifying Code Pages

The Codeset command enables you to specify various code pages for your installation. A code page defines the method of encoding characters, and encompasses:

- The different ways characters are encoded on various platforms (for example, EBCDIC on z/OS and ASCII on Windows and UNIX).
- The differences between human languages.

Every item in Dimensions CM has a code page associated with it, defined or derived for:

- The connection setting.
- An individual item.

For full details about this command and its qualifiers, see "Codeset" on page 293. For more details about code pages, see page 256.

Specifying File Systems for Remote Nodes

The File System command enables you to specify file systems for each registered operating system on a remote node. The command ensures that you can only create unique file systems. After a file system is defined, you cannot delete it unless its definitions are no longer in use.

For full details about this command and its qualifiers, see "File System" on page 296.

Registering Resident Software

The RSD command registers the resident software associated with the build environment in which Dimensions Make operates. Any Dimensions item which is built via Dimensions Make is registered against these definitions.

For full details about this command and its qualifiers, see "RSD" on page 303.

Dimensions CM Listener

Listener Startup and Shutdown

The Dimensions CM Listener must be started up on every machine that is to provide access to a Dimensions CM database, to item libraries hosted locally on that node, or to perform file operations in user work areas.

The Listener is started as part of the general Dimensions CM server or agent startup.

On UNIX servers and agents, startup is performed by the following scripts:

```
dmstartup
dmshutdown
```

or

```
dm_control cm_start
dm_control cm_stop
```

On Windows servers and agents, startup is performed automatically through a Windows service.

The Listener is activated by the program `dmlsnr` and is configured to listen to a specific port in order to service client network requests.

For details about the command syntax of `dmlsnr`, see Chapter 10, "General Operating Guide".

For details about the Dimensions CM network architecture, see the Dimensions CM online help.

Listener Functions

The Listener has three distinct functions:

Listener Function	Notes
Acts as a connection pool manager for Dimensions CM for accessing local or remote Dimensions CM databases. For more details, see the *Installation Guide.*	You must generally ensure that the Listener is run as the administrator user (but see page 169 for running as a restricted user): ■ On UNIX servers or agents: the "root" account. ■ On Windows servers or agents: normally the "Local Administrator" account.
Creates user-file server processes that perform file operations in user work areas on behalf of the client user, for example, checking in and checking out files.	
Creates library server processes that provide access to the Dimensions CM item libraries that are hosted on the local machine.	

Chapter 21

Network Applications

About Network Administration	276
Network Administration Main Window	276
Managing Network Nodes	280
Managing Client/Server Connections	284
Viewing Objects	285

About Network Administration

Purpose

Network Administration enables you to:

- Create network nodes and client/server connections.
- Add, edit, and delete contacts for use when creating physical nodes.
- Assign and edit network objects an nd base databases from network nodes.
- View details about the predefined operating systems, file systems, and code sets for use when creating network nodes or client/server connections.

Constraints

Only users with ADMIN_NETWORK privileges on the base database can modify the network configuration.

Invocation

Dimensions CM Administration Console | Distributed Development | Network administration

Network Administration Main Window

The Network Administration main window consists of the following areas:

- **Menu area:** Displays the program's tabs and toolbars to help you carry out various tasks. See "Network Administration Menu Area" on page 276.
- **Navigation area:** Enables you to view and select from a list of network nodes, connections, or other objects, depending on the selected tab. See "Network Administration Navigation Area" on page 277.
- **Content area:** Displays details about the selected object. See "Network Administration Content Area" on page 278.
- **Status area:** Displays log in details. See "Network Administration Status Area" on page 280.

Network Administration Menu Area

The Network Administration menu area consists of three menu tabs and their associated toolbars.

Network Nodes Tab: Menu Area

The **Network Nodes** tab enables you to create physical and logical network nodes. See the table below for a description of the toolbar buttons that appear on the **Network Nodes** tab. These toolbar buttons change for each of the top left drop down selections, namely, All Nodes, Physical Nodes, or Logical Nodes.

.

Button	Function
	Add a new physical or logical node using the New/Edit Node Dialog Box.
	Delete the selected nodes.

Node Connections Tab: Menu Area

The **Node Connections** tab enables you to create connections between client and server nodes. See the table below for a description of the toolbar buttons that appear on the **Node Connections** tab.

Button	Function
	Add a new connection using the Register/Edit Client Server Connection Dialog Box.
	Delete the selected connections.

Advanced Tab: Menu Area

The **Advanced** tab enables you to view top-level objects, such as operating systems and file systems. You can also manage contact details using this tab. See the table below for a description of the toolbar buttons that appear on the **Advanced** tab. Note these options *only* appear in the context of Contacts management.

Button	Function
	Add a new contact using the New/Edit Network Node Contact Dialog Box.
	Delete the selected contacts.

Network Administration Navigation Area

The list of objects displayed in the navigation area depends on which tab you have selected.

Network Nodes Tab: Navigation Area

With the **Network Nodes** tab selected, the navigation area enables you to:

- View the list of existing physical and logical network nodes.
- Filter nodes by type: **All Nodes**, **Physical Nodes**, or **Logical Nodes**.
- Select a node to view associated details in the content area and perform actions such as edit or delete.

Node Connections Tab: Navigation Area

With the **Node Connections** tab selected, the navigation area enables you to:

- View the list of existing client/server connections.
- Select a connection to view associated details and perform actions such as edit or delete.

Advanced Tab: Navigation Area

With the **Advanced** tab selected, the navigation area enables you to:

- Filter the top-level objects used when defining nodes or client/server connections: **Operating Systems**, **Contacts**, **File Systems**, or **Code Sets**.
- Select an object to view its associated details in the content area.

Network Administration Content Area

The information that displays in the content area varies depending on which tab is active and which object you selected in the navigation area. The tables below describe this.

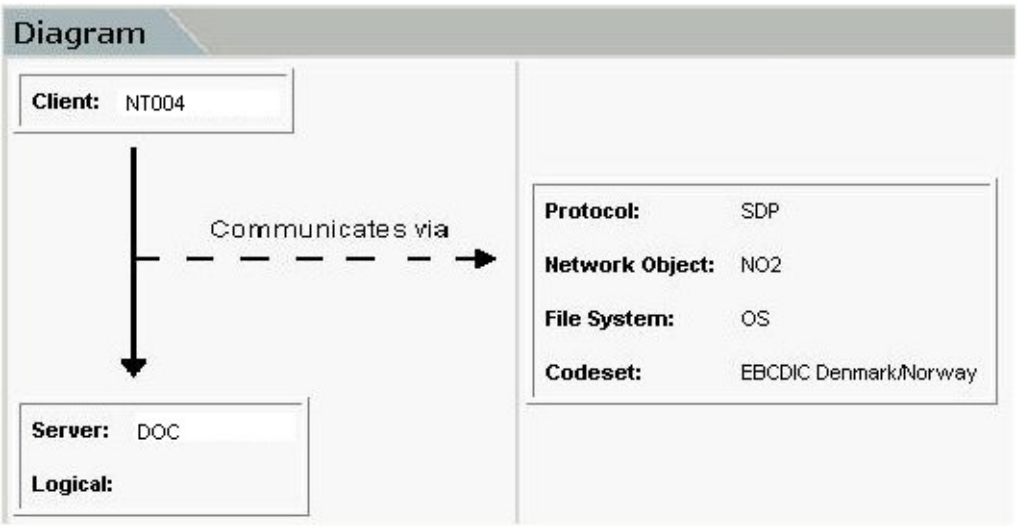
Network Nodes Tab: Content Area

If you select...	Then the content area displays:
A single node	■ **General:** This section displays details about the selected node and enables you to edit it. For a selected physical node, you can see all of the associated logical nodes in this section. For a selected logical node, you can see the associated physical node as well as all of the logical nodes associated with that physical node. ■ **Contact:** This section displays details about the user responsible for the node. ■ **Network Object:** Section that displays details about the network object assigned to the node and enables you to edit or assign another network object. ■ **Base Database:** This section displays details about the base databases assigned to the node and enables you to manage these associations.
Network Nodes	■ **List of objects:** This section displays a detailed summary of all network nodes in the base database. ■ : Print the summary of nodes. ■ : Save the summary of nodes as comma-separated values.

Node Connections Tab: Content Area

If you select...	Then the content area displays:
A single node connection	▪ **General:** This section displays details about the selected node connection and enables you edit it. ▪ **Diagram:** This section displays a graphical representation of the connection. See the example of a Diagram section below.
Connections	▪ **Node Connections:** This section displays a detailed summary of all node connections in the base database. ▪ : Print the summary of connections. ▪ : Save the summary of connections as comma-separated values.

Example of a Diagram for a Node Connection

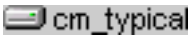


Advanced Tab: Content Area

If you select...	Then the content area displays:
A single operating system, contact, file system, or code set	Predefined details about the selected object.
A single contact	Details about the selected contact and enables you to edit these.
Operating Systems Contacts File Systems Code Sets	▪ Detailed summary of all operating systems, contacts, file systems, or code sets in the base database. ▪ : Print the summary of objects. ▪ : Save the summary of objects as comma-separated values.

Network Administration Status Area

The status area contains the following icons:

Icon	Description
DMSYS	User ID of the currently logged in user.
Preferences	Click this link to display the Set User Preferences dialog box.
	Name of the current product. You can click the icon to change the product.
	Name of the base database you are currently connected to.
	Authenticate with a remote host or change a remote host password.

Managing Network Nodes

Creating Nodes

Purpose

Follow this procedure when you want to add a physical or logical network node to the network. A *physical node* is an actual physical machine name, while a *logical node* is a user-defined alias for that physical machine. You can assign multiple logical nodes to a single physical machine.

NOTE Logical node names cannot be used for item library definitions.

See "About Network Administration" on page 276 for more details of physical and logical nodes.

Constraint

Before creating a logical node, you must first create a physical node to which you associate the logical node. The logical node inherits the values of the physical node, except its description.

To create a physical node:

1. From the Network Administration window, click the **Network Nodes** tab.
2. Click the New Node button then select **Physical Network Node**. The New/Edit Node Dialog Box appears.
3. Enter the name of the new node in the **Physical node name** field.
4. Select the operation system of the node from the **Operating system** list.
5. Optionally, select a predefined contact if applicable from the **Contact** list or complete the **Description** field.
6. Click **OK** to add the node.
7. Optionally, assign a contact to the node if one is not present. See "Managing Contacts" on page 281.

8 Optionally, assign a network object to the node. See "Managing Network Objects" on page 282.

9 Optionally, assign a base database to the node. See "Managing Base Databases" on page 282.

To add a logical node:

1 From the Network Administration window, click the **Network Nodes** tab.

2 Click the New Node button then select **Logical Network Node**. The New/Edit Node Dialog Box appears.

3 Enter the name of the new node in the **Logical node name** field.

4 Select the physical node to which you want to associate the logical node from the **Physical node** list.

5 Optionally, complete the **Description** field.

6 Click **OK** to add the node.

Managing Contacts

Purpose Follow this procedure to transfer objects between remote when you want to add a network contact to Dimensions CM You can select the contact when creating a physical node.

Once you've added a contact, you can modify or delete it as necessary.

Constraints
- Only one contact at a time can be associated with a physical node.
- You cannot delete a contact that is currently assigned to a physical node.

To add a contact:

1 From the Network Administration window, click the **Advanced** tab.

2 In the navigation area, select **Contacts** from the list.

3 Click the New button. The New/Edit Network Node Contact Dialog Box appears.

4 Enter the name of the contact person in the **Contact Full Name** field.

5 Optionally, complete the rest of the fields in the dialog box.

6 Click **OK** to create the contact.

To edit or delete a contact:

1 Select the contact on the **Advanced** tab.

2 Do one of the following:
   - Click the Edit button and modify the details of the New/Edit Network Node Contact Dialog Box as necessary.
   - Click the Delete button and confirm that you want to delete the contact.

Managing Network Objects

Purpose Follow this procedure when you want to assign a network object to a network node. The name of *network object* is taken as being the TCP/IP port number of the Dimensions CM listener on that node and is used to communicate between nodes in the network. For TCP/IP this is a socket listed in the services file or its equivalent.

After assigning a network object, you can edit it or unassign it from the node if necessary.

To assign a network object:

1. From the Network Administration window, click the **Network Nodes** tab.
2. Select the node to which you want to assign the network object in the navigation area.
3. Click the Add button in the Network Object section. The Assign/Edit Node Object Dialog Box appears.
4. Select the existing network object name from the **Network Object Name** list or enter a name to define a new object. The existing name is the `'- service'` entry in the `listener.dat` file located in the `dfs` subdirectory.
5. Select the appropriate network protocol from the **Protocol** list, either `SDP` for the standard network protocol or `SSLSDP` if a Secure Sockets Layers network protocol is being used.
6. Optionally, complete the rest of the fields in the dialog box.
7. Click **OK** to assign the network object to the node.

To edit or unassign a network object:

1. Select the node to which the network object is assigned in the navigation area.
2. Select the network object from the Network Object section.
3. Do one of the following:
   - Click the Edit button and modify the details of the Assign/Edit Node Object Dialog Box as necessary.
   - Click the Deassign button and confirm that you want to unassign the network object from the node.

Managing Base Databases

Purpose Follow this procedure to assign a base database to a network node. This assignment may be used by Replicator to transfer objects between remote Dimensions repositories.

NOTE If you are using Replicator, then you need to ensure that not only are the base databases that you are replicating to are correctly registered, but also that the base database which you are logged into is correctly registered too. In addition, if the `PCMS_SYS` database was imported from another database, then the data in `pcms_sys.pcms_db_details` may need the column `db_name` updating (to `NODENAME-DBCONN`) before doing replication administration.

After assigning a base database to a node, you can edit it or unassign it from the node if necessary.

To assign a base database:

1. From the Network Administration main window, click the **Network Nodes** tab.
2. Select the network node to which you want to assign the base database.
3. Click the Add button in the Base Database section. The New/Edit Base Database Dialog Box appears.
4. Enter the name of the base database in the **Base Database Name** field.
5. Select or enter the name of the network instance in the **DB Connection String** field.
6. Enter the Dimensions CM version number in the **Dimensions Release Version** field.
7. If this node is to be included in request replications between sites, make sure that the site number is unique and consistent across all the sites involved in the replication. This is controlled by the **Site No** field.
8. Optionally, complete the rest of the fields in the dialog box
9. Click **OK** to add the base database to the node.

To edit or unassign a base database:

1. Select the node to which the base database is assigned.
2. Select the base database from the Base Database section.
3. Do one of the following:
   - Click the Edit button and modify the details of the New/Edit Base Database Dialog Box as necessary.
   - Click the Deassign button and confirm that you want to unassign the base database from the node. If **Replication Enabled** was checked, you have to first clear it by using the Edit button before you can successfully use the Deassign button.

Editing or Deleting Nodes

Purpose Follow this procedure when you want to edit or delete an existing physical or logical node definition.

Constraint When you edit a physical or logical node, you cannot change its name.

To edit or delete a node:

1. From the Network Administration window, click the **Network Nodes** tab.
2. Select the node that you want to edit or delete in the navigation area.
3. Do one of the following:
   - Click the Edit button in the General section and modify the details of the New/Edit Node Dialog Box as necessary.
   - Click the Delete button and confirm that you want to delete the node.
4. Optionally, do the following:

- To edit the details of a contact that is currently assigned to the node, see "Managing Contacts" on page 281.
- To edit or unassign a network object that is currently assigned to the node, see "Managing Network Objects" on page 282.
- To edit or unassign a base database that is currently assigned to the node, see "Managing Base Databases" on page 282.

Managing Client/Server Connections

Creating Client/Server Connections

Purpose Follow this procedure when you want to establish access from a Dimensions CM client node to a server node. In Dimensions CM networking terms, a *client* node is actually the node where the server is located and the *server* node is the node where the Dimensions CM listener is located. This connection enables a client to access the item libraries and user files that reside on the server.

To create a connection:

1. From the Network Administration window, click the **Node Connections** tab.
2. Click the New button. The Register/Edit Client Server Connection Dialog Box appears.
3. Select the network client that you want to connect to from the **Client Node** list.
4. Select the network server that you want to connect to from the **Server Node** list.
5. Select the network object to facilitate the connection from the **Network Object** list.
6. Optionally, select the appropriate network protocol from the **Protocol** list—either `SDP` for the standard network protocol or `SSLSDP` if a Secure Sockets Layers network protocol is being used.
7. Optionally, complete the rest of the fields in the dialog box.
8. Click **OK** to add the connection.

Editing or Deleting Connections

Purpose Follow this procedure when you want to edit or delete an existing client/server connection.

To edit or delete a connection:

1. From the Network Administration window, click the **Node Connections** tab.
2. Select the connection that you want to edit or delete in the navigation area.
3. Do one of the following:
   - Click the Edit button and modify the details of the Register/Edit Client Server Connection Dialog Box as necessary.
   - Click the Delete button and confirm that you want to delete the connection.

Viewing Objects

This section shows you how to view the values for predefined operating systems, file systems, and code sets. You can choose from among these values when you create a network node or a client/server connection.

- **Operating system:** Network Administration provides a list of supported platforms with standard case conventions. When you define a physical node, you choose the operating system to be used by that node.
- **File system:** Network Administration provides a list of file system types. When you define a client/server connection, you choose what file system is part of your network.
- **Code set:** Network Administration provides a list of code sets available for the network. A code set or code page is a method for encoding characters for different platforms and for different human languages. When you define a client/server connection, you choose a code set.
- **Contact:** Network Administration enables you to add, edit, or delete a list of network contacts. When you create a physical node, you choose a single contact to be responsible for the node.

Viewing Objects

Purpose Follow this procedure when you want to view a list of available operating systems, file systems, code sets, or contacts. You can also view values for each operating system, file system, code set, or contact.

To view an object:

1. From the Network Administration main window, click the **Advanced** tab.
2. In the Navigation area, select **Operating Systems**, **File Systems**, **Code Sets**, **Contacts** from the filter.
3. Select the top-level object to view a list of all operating systems, file systems, code sets, or contacts, or click a defined object to view its details.

Chapter 22

Network Administration Command Line Interface

Using the Dimensions CM Command Line Interface	288
Sample Network Administration Scenario	289
Base Database	291
Codeset	293
Contact	293
Database Instance	295
File System	296
Network Node	297
Network Node Connections	298
Network Object	300
Node Objects	301
Operating System	301
Protocols	302
RSD	303

Using the Dimensions CM Command Line Interface

Most Dimensions® CM networking commands have the following functions:

- Create
- Delete
- List
- Update

If you do not enter any command parameters, the mandatory parameters for that command are displayed, for example:

```
Dimensions> cos

Error: command requires qualifier(s)
 /case
 /lib_protection
 /os_name
Error: Dimensions command not specified correctly
Dimensions>
```

All qualifiers must be within the stated size, otherwise the command line displays an error and fails. For example:

```
Dimensions> cos /case=xxxxxxxxxxx /lib_protection=rwx,rx,r /os_name=xxx
Error: CASE specified should be no more than 10 characters long
Dimensions>
```

Using Key Qualifiers

Key qualifiers are qualifiers that are mandatory for a command's operation. The key qualifiers are listed in the sections in this chapter that describe the syntax for each command.

If you did not specify a value for the key qualifier, an error message is displayed, for example:

```
> ERROR:/nn_name not found
```

Listing Qualifiers

You can output a summary list for each particular networking command, by running the `dmcli help` utility. For example, to list the qualifiers for the `cos` networking command, type:

```
help cos
```

Sample Network Administration Scenario

The following sample flow describes a typical scenario for setting up and administering a Dimensions CM network. The scenario explains how to use the main network commands described in this chapter, and comprises the following steps:

Add an operating system

To add an operating system to an installation, use the `cos` command (for full details see "Operating System" on page 301). The following command adds the operating system "Windows XP":

```
cos /os_name=XP
    /case=NN
    /lib_protection="RWX,RX,R"
```

Add a contact

To add a contact to an installation, use the `cco` command (for details see "Contact" on page 293). The following command adds a contact called "Server Manager":

```
cco /co_name="Server Manager"
    /title="Senior Server Manager" /email=Server.Manager@@company.com
    /address="St Albans" /contact_type=M /contact_id="John Brown"
```

Add a file system

To add a new file system for each registered operating system, use the `cfs` command (for full details see "File System" on page 296). The following command adds the "NTFS" file system:

```
cfs /fs_name=NTFS
    /description="NT File System"
```

Add a codeset

To add a codeset to an installation, use the `ccst` command (for full details see "Codeset" on page 293). The following command adds the codeset "EBCDIC Ireland (Euro)":

```
ccst /cdst_number="2000"
    /description="Description - EBCDIC Ireland (Euro)"
```

Add a network object

To add a network object to an installation, use the `cnwo` command (for full details see page 300). The following command adds a network object called "pcms_sdp" that uses SDP (Standard Dimensions CM Protocol):

```
cnwo /nwo_name=pcms_sdp
    /protocol=sdp
    /description="Network object using Standard Dimensions CM Protocols"
```

Add a physical network node:

To add a physical network node, use the `cnn` command and specify "N" for the "logical" qualifier (for full details see "Network Node" on page 297). The following command adds "machine.company.com" as a physical node (you must specify a physical machine name):

```
cnn /nn_name=machine.company.com
    /logical=N
    /os_name=XP
    /description="Sources held on DFS dimensions XP server"
```

Add a logical network node:

To add a logical network node, use the `cnn` command and specify "Y" for the "logical" qualifier (for full details see "Network Node" on page 297). The following command adds "My_Mainframe" as a logical node (you must also specify a physical machine name for the node):

```
cnn /nn_name="My_Mainframe"
    /physical_name=mf390.company.com
    /logical=Y
    /os_name=MVS
    /description="Mainframe server situated at head office"
```

List network nodes

To verify that the nodes specified above were added, use the `lnn` command (for full details see "Network Node" on page 297).

List node objects

To list the node objects currently defined, use the `lndo` command (for full details see "Node Objects" on page 301). A node object represents an association between a network node and a network object.

Add a node connection

To add a node connection, use the `cnc` command (for full details see "Network Node Connections" on page 298). The following command creates a client/server network connection between "test_machine.company.com" and "machine.company.com" using the network protocol "pcms_sdp":

```
cnc  /client_name=test_machine.company.com
    /server_name=machine.company.com
    /fs_name=ntfs
    /nwo_name=pcms_sdp
    /cdst_number="2000"
```

List network connections

To verify that the node connection specified above was added, use the `lnc` command (for full details see "Network Node Connections" on page 298).

Create a database instance

To add a database instance to an installation, use the `cins` command (for full details see "Database Instance" on page 295). The following command adds a database instance called "dim10":

```
    cins /nn_name=machine.company.com
    /db_service=dim10
    /db_home_dir=.
    /db_name=qlarius_cm
```

List database instances

To verify that the database instance specified above was added, use the `lins` command (for full details see "Database Instance" on page 295).

Add a base database

To add a base database to an installation, use the `cbdb` command (for full details see "Base Database" on page 291). The following command adds a base database called "example_dmcm":

```
cbdb /bdb_name=example_dmcm
     /nn_name=machine.company.com
     /db_service=dim10
     /cap_replicate=N
     /pcms_root_dir="Dimensions CM Root Directory"
     /pcms_ver="Dimensions CM 14.5"
     /description="Details of base database for node
     machine.company.com"
```

List base databases

To list all the base databases in an installation, use the `lbdb` command (for full details see "Base Database" on page 291).

Base Database

Description

The Base Database command enables you to add a new base database to an installation, and edit, list, or delete an existing base database.

Commands

`cbdb` - Create

`dbdb` - Delete

`lbdb` - List

`ubdb` - Update

Qualifiers

Qualifier	Description	Type	Qualifier is mandatory for:		
			Create	Delete	Update
bdb_name	Dimensions CM base database name (25 characters maximum) Key qualifier	Char	Y	Y	Y
description	Describes the base database (240 characters maximum)	Char			

Qualifier	Description	Type	Qualifier is mandatory for:		
			Create	Delete	Update
pcms_ver	Dimensions CM version (20 characters maximum)	Char	Y		
pcms_root_dir	Dimensions CM root directory (1024 characters maximum)	Char			
co_name	Contact name (key qualifier for Contact command)	Char			
cap_replicate	Project replication (Y/N).	Char	Y		
db_service	Base database instance (key qualifier for Database Instance command)	Char	Y	Y	Y
nn_name	Network node name (key qualifier for Network Node command)	Char	Y	Y	Y
site_no	Dimensions CM database site number.	Char			

Notes

1. To identify an instance, you must use the 'db_service' and 'nn_name' qualifiers.
2. To ensure that the requests that are created have a unique id across all Dimensions sites participating in a replication, you must specify a site number when you are replicating requests. The 'site_no' qualifier staggers the id of the first request created on a site, and staggers the increment to the request number when subsequent requests are created.

Examples

To create a base database:

```
cbdb /bdb_name=example_dmcm
     /nn_name=machine.company.com
     /db_service=dim10
     /cap_replicate=N
     /pcms_root_dir="C:\Program Files\OpenText\Dimensions <version>\CM"
     /pcms_ver="Dimensions CM <version>"
     /description="Details of base database for node
     machine.company.com"
     /site_no=1
```

To delete a base database:

```
dbdb /bdb_name=example_dmcm
     /db_service=dim10
     /nn_name=machine.company.com
```

Codeset

Description

The Codeset command enables you to add a new codeset to an installation, and edit, list, or delete an existing codeset.

Commands

`ccst` - Create

`dcst` - Delete

`lcst` - List

`ucst` - Update

Qualifiers

Qualifier	Description	Type	Qualifier is mandatory for:		
			Create	Delete	Update
cdst_number	Codeset number Key qualifier	Char	Y	Y	Y
description	Describes the codeset	Char	Y		

Examples

To add a codeset:

```
ccst /cdst_number="2000"
    /description="Description - EBCDIC Ireland (Euro)"
```

To delete a codeset:

```
dcst /cdst_number="2000"
```

Contact

Description

The Contact command enables you to add a new contact to an installation, and edit, list, or delete an existing contact.

Commands

`cco` - Create

`dco` - Delete

`lco` - List

`uco` - Update

Qualifiers

Qualifier	Description	Type	Qualifier is mandatory for:		
			Create	Delete	Update
co_name	The name of the contact person (25 characters maximum) Key qualifier	Char	Y	Y	Y
title	The title of the contact person (256 characters maximum)	Char			
email	The email address of the contact person (256 characters maximum). **Note** The default `dmcli` "escape" character is `'@'`, therefore, an address such as `IT.Head@company.com` needs to be entered as `IT.Head@@company.com` See the *Command-Line Reference* for details about the dmcli escape character.	Char			
address	The address of the contact person (2000 characters maximum)	Char			
contact_type	User defined	Char			
contact_id	Identity of the contact person (256 characters maximum)	Char			

Examples

To add contacts:

```
cco /co_name="Server Manager"
    /title="Senior Server Manager" /email=Server.Manager@@company.com
    /address="Abbey View, St Albans"
    /contact_type=M /contact_id="John Brown"

cco /co_name="Mainframe Manager"
    /title="Mainframe Architect"   /
    email=Mainframe.Manager@@company.com
    /address="Abbey View, St Albans"
    /contact_type=M /contact_id="Janet Green"
```

```
cco /co_name="DB Admin"
    /title="DBA"
    /email=dba.Manager@@company.com
    /addrss="Abbey View, St Albans"
    /contact_type=M /contact_id="Fred Bowyer"
```

To delete contacts:

```
dco /co_name="Server Manager"
dco /co_name="Mainframe Manager"
dco /co_name="DB Admin"
```

Database Instance

Description

The Database Instance command enables you to add a new database instance to an installation, and edit, list, or delete an existing database instance.

Commands

`cins` - Create

`dins` - Delete

`lins` - List

`uins` - Update

Qualifiers

Qualifier	Description	Type	Qualifier is mandatory for:		
			Create	Delete	Update
db_service	Instance name Key qualifier	Char	Y	Y	Y
nn_name	Network node name (key qualifier for Network Node command)	Char	Y	Y	Y
db_transport	Database specific transport, such as Oracle SID	Char			
db_name	Database name	Char			
db_home_dir	Home directory of the database instance	Char			
db_two_task	Oracle specific remote connection string	Char			
db_active	(Unused/reserved) Actively used (Y/N)	Char			
co_name	Contact name (key qualifier for Contact command)	Char			

Examples

To create a database instance:

```
cins /nn_name=machine.company.com
     /db_service=dim10
     /db_home_dir=.
     /db_name=qlarius_cm
```

To delete a database instance:

```
dins /nn_name=machine.company.com
     /db_service=dim10
```

File System

Description

Use the File System command to add, edit, list, or delete specific file systems for each registered operating system. The command ensures that you can only create unique file systems. After a file system is defined, you cannot delete it unless its definitions are no longer in use.

Commands

`cfs` - Create

`dfs` - Delete

`lfs` - List

`ufs` - Update

Qualifiers

Qualifier	Description	Type	Qualifier is mandatory for:		
			Create	Delete	Update
fs_name	File system name Key qualifier	Char	Y	Y	Y
description	Describes the file system	Char			

Examples

To add a file system:

```
cfs /fs_name=NTFS
     /description="XP File System"
```

Network Node

Description

The Network Node command enables you to create a new node, and edit, list, or delete an existing node. The node can be physical or logical (an alias for a physical node). To create a logical node, set the `/logical` qualifier to 'Y' (`/logical=y`). The command also enables network objects, databases, resident software definition details and contacts to be associated with a node.

Commands

`cnn` - Create

`dnn` - Delete

`lnn` - List

`unn` - Update

Qualifiers

Qualifier	Description	Type	Qualifier is mandatory for:		
			Create	Delete	Update
nn_name	Network node name Key qualifier	Char	Y	Y	Y
os_name	Operating system name (key qualifier for Operating System command)	Char	Y		
logical	Logical node name (Y/N) (127 characters maximum)	Char	Y		
physical_name	Physical name of a node (	Char	Y (for logical nodes)		
co_name	Contact name (key qualifier for Contact command)	Char			
description	Describes the network node	Char			
rsd_name	Resident software definition (key qualifier for RSD command)	Char			

Examples

To create a physical node:

```
cnn /nn_name=machine.company.com
    /logical=N
    /os_name=XP
    /description="Sources held on Dimensions XP server"
```

```
cnn /nn_name=test_machine.company.com
    /logical=N
    /os_name=XP
```

NOTE If you omit the optional qualifier 'physical_name' when you create a physical node, it inherits the value of 'nn_name'. If you use 'physical_name', it must have the same value as 'nn_name'.

To delete a physical node:

```
dnn /nn_name=machine.company.com
```

To create a logical node:

```
cnn /nn_name="My_Mainframe" /physical_name=mf390.company.com
    /logical=Y
    /os_name=MVS
    /description="Mainframe server situated at head office"
```

Network Node Connections

Description

The Network Node Connections command enables you to add a new network node connection to an installation, and edit, list, or delete an existing connection.

Commands

`cnc` - Create

`dnc` - Delete

`lnc` - List

`unc` - Update

Qualifiers

Qualifier	Description		Type	Qualifier is mandatory for:		
				Create	Delete	Update
server_name	Network server node	**Key qualifiers** that uniquely identify the network connection	Char	Y	Y	Y
client_name	Network client node		Char	Y	Y	Y
nwo_name	Network object		Char	Y	Y	Y
fs_name	File system name (key qualifier for File System command)		Char			

Qualifier	Description		Type	Qualifier is mandatory for:		
				Create	Delete	Update
cdst_number	Codeset number (key qualifier for Codeset command)		Char			
direct_file_copy	Used, where possible, to enable direct file transfers between two nodes, thus offering performance enhancement.		Char			
file_compression	Enable compressed file transfers between two nodes, thus offering performance enhancement.		Char			

NOTE You must use all the mandatory qualifiers to create, delete or update a network node.

Notes

1. When you create a logical node you must specify a value for the qualifier 'physical_name', and it cannot have the same value as 'nn_name'.
2. The direct file copy feature only works on physical nodes, not logical nodes. So, to enable a direct copy between machine A and machine B, a connection should be created between the physical node for A and the physical node for B, with the feature enabled. Note that this should also be done for any machines performing the role of Library Caches.
3. To allow direct file copy from a mainframe item library back to that same mainframe, the connection needs to be defined with the mainframe physical node at both ends. This is a common requirement on MVS but may also be a requirement on other platforms. For example, you create a network connection from A001.ACME.COM to A001.ACME.COM, and enable direct copy on that connection.
4. The `/FILE_COMPRESSION` option (to switch on file-compression along a network connection link) is available everywhere that `/DIRECT_FILE_COPY` is available. See "File Compression Settings" on page 55.

Examples

To create a network node connection:

```
cnc  /client_name=test_machine.company.com
     /server_name=machine.company.com
     /fs_name=NTFS
     /nwo_name=PCMS_SDP
     /cdst_number="2000"
```

To delete a network node connection:

```
dnc  /client_name=test_machine.company.com
     /server_name=machine.company.com
     /nwo_name=PCMS_SDP
```

Network Object

Description

The Network Object command enables you to add a new network object to an installation, and edit, list, or delete an existing network object.

Commands

`cnwo` - Create

`dnwo` - Delete

`lnwo` - List

`unwo` - Update

Qualifiers

Qualifier	Description	Type	Qualifier is mandatory for:		
			Create	Delete	Update
nwo_name	Network object name (20 characters maximum) **Key qualifier**	Char	Y	Y	Y
protocol	Communications protocol used by the network object	Char	Y		
description	Describes the network object (240 characters maximum)	Char			
process	Network object process name (10 characters maximum)	Char			

Examples

The protocols table is read-only and is populated by the install/upgrade script. The example below assumes there is an SDP entry in the Protocols table (use the `lprt` command to check the protocol).

To create a network object:

```
cnwo /nwo_name=pcms_sdp
    /protocol=sdp
    /description="Network object using Standard Dimensions CM Protocols"
```

To delete a network object:

```
dnwo /nwo_name=pcms_sdp
```

Node Objects

Description

The Node Object command enables you to add a new node object to an installation, and list or delete an existing node object. A node object represents an association of a network node and a network object.

Commands

`cndo` - Create

`dndo` - Delete

`lndo` - List

Qualifiers

Qualifier	Description		Type	Qualifier is mandatory for:	
				Create	Delete
nn_name	Network node name	**Mandatory Key qualifiers**	Char	Y	Y
nwo_name	Network object name		Char	Y	Y

Operating System

Description

The Operating System command enables you to add new operating systems to an installation, and edit, list, or delete an existing operating system.

Commands

`cos` - Create

`dos` - Delete

`los` - List

`uos` - Update

Qualifiers

Qualifier	Description	Type	Qualifier is mandatory for:		
			Create	Delete	Update
os_name	Operating system name (four characters maximum). **Key qualifier**	Char	Y	Y	Y
case	Operating system case convention. A two character string 'XY', where: X specifies filename case conversions Y specifies username case conversion and the values of X or Y can be: N Mixed case U Uppercase L Lowercase	Char	Y		
lib_protection	Type of library protection (depends on the type of operating system and must be enclosed in double quotes). For example, "RWX,RX,R"	Char	Y		

Examples

To add an operating system:

```
cos /os_name=XP
    /case=NN
    /lib_protection=RWX,RX,R
```

Protocols

Description

The network protocol is the protocol used by a network object. This command lists the available protocols.

Command

```
lprt
```

RSD

Description

The RSD command registers the resident software associated with the build environment in which Dimensions Make operates. Any Dimensions item which is built through Dimensions Make is registered against these definitions.

Commands

`crsd` - Create

`drsd` - Delete

`lrsd` - List

`ursd` - Update

Qualifiers

Qualifier	Description	Type	Qualifier is mandatory for:		
			Create	Delete	Update
rsd_name	A name for the software details **Key qualifier**	Char	Y	Y	Y
rsd_version	Version number of the software details	Int	Y		
rsd_data	Software definition details	Char			

Part 5
Replication

Multi-Site Development Using Replicator	307
Online Replication Concepts and Scenarios	319
Preparing for Replication	349
Replication Rules and Constraints	371
Running Replicator	379
Replication Administration	355

Chapter 23

Multi-Site Development Using Replicator

Introduction	308
Getting Started	313
Running Replicator	317
Replication Results	317
Replicator Published Views	317

Introduction

Overview

Software Engineering can be performed in a variety of different physical environments, involving teams of users distributed on a number of *networked* or *physically isolated* machines. These users do not want to be concerned about how these machines are set up or connected. Their focus is to perform their development unhindered and have access to up-to-date common configurations of objects that are pertinent to their work, even if they were developed in other geographically distributed locations.

If software is to be developed and maintained in parallel, or in collaboration, at different geographical sites, it is imperative that a proven process be put in place which enables the distributed teams of developers to effectively and predictably handle the complexity of changes during the development process. This process must include the deployment of effective procedures and controls for allowing code changes to occur across multiple sites while catering for the diversity in the development culture of the developer teams at those sites.

Dimensions Replicator is designed specifically to support software development effectively in decentralized and distributed team environments. Replicator supports remote and distributed development of Dimensions items, baselines, and requests by allowing teams of developers, located at different geographical locations, to work seamlessly in parallel on the same project files while accessing their own local repository. Behind the scenes, Replicator ensures that items, baselines, and requests, plus their associated metadata, are replicated from one repository to another according to the replication policies set up by the project administrator. It ensures that these separate repositories are always kept in synchronization with any changes in the policies.

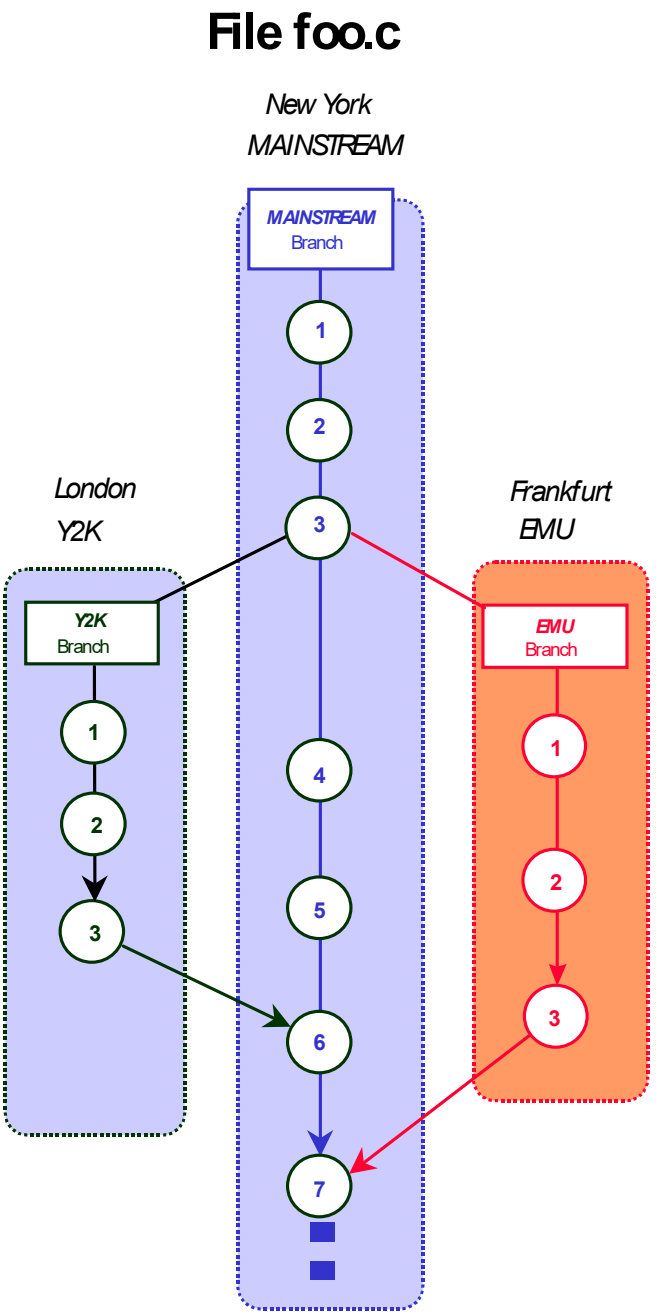
IMPORTANT!

- Replicator does not support the use of Dimensions CM streams.
- Replication of design parts is not currently supported. A separate data transfer utility, "PDIFF Import/Export", is available for transferring design parts between databases. Please refer to Chapter 16, "Data Interchange File Format (PDIFF)" and Chapter 17, "Using PDIFF to Import/Export Data" for details.
- Replication of requirements information from one Dimensions server to another is not currently supported in Dimensions. However, by carefully coordinating Dimensions requests, you can still fully track the fulfillment of requirements in a replicated Dimensions environment. See, "Integrating Dimensions RM and CM in a Replicated Environment" on page 313.

Item Replication

The version tree of a file under configuration control can normally be broken into separate branches, each branch containing a specific chain of revisions that were created in the context of a specific development goal, for example, mainstream, customization, or maintenance. In a multisite development environment, Replicator ensures that each version branch of a file is owned by only one of the participating sites. The golden rule is that a version branch can be modified by only those developers who work on the site that owns the branch. In this way Replicator segregates completely the development activities that take place at these different sites. By using the comprehensive merging facilities

provided by the PVCS Merge Tool, a site is able to resolve, manually or automatically, the conflicting or non-conflicting distributed changes that were made to a file by merging the tip nodes of the relevant "remote" branches with a node on a "local" branch.



Consider the example diagram above that illustrates a version tree for a file developed across three different sites: New York, London, and Frankfurt. In this example the file `foo.c` is owned by New York, whose developers have been doing their work on the `mainstream` branch. At `Revision 3` the file is replicated to London for Year 2000 development and to Frankfurt for EMU changes. London does its work on the `y2k` branch and Frankfurt on the `emu` branch. Further concurrent work at these locations is then replicated back to New York to be included in `Revision 7` (which also includes further concurrent work in New York).

New default replication behavior

Dimensions CM 2009 R1 or later includes new default replication behavior, first made available as an earlier customer-specific patch, namely:

> When items are dumped as part of item replication all relationships between the item and change requests, design parts and other items are also replicated. In addition the related items (and any required placeholder items) are also dumped as objects—this does not apply to the related design parts or change requests. The relationships of any placeholder items dumped are not dumped.

With respect to the items replicated:

- All usage relationships to design parts are replicated.
- All relationships to items are replicated when the items are on a version branch that matched the replication policy.
- Only the `Affected` item to change request relationships are replicated. `InResponsTo` and `Info` relationships are ignored as part of the item replication. These relationships are still replicated as part of change request replication. Furthermore, only relationships to change requests in the primary catalogue are replicated, relationships to request in the secondary catalogue are ignored.

Baseline Replication

Replicator not only enables the replication of items in a project, but also baselines and all the items contained in those baselines.

When a baseline is replicated from a primary to one or more replica sites, then the following information is transferred as part of that replication:

- The metadata associated with the baseline itself. This refers to details such as user attributes, the date the baseline was created, and the user who created it.
- The items and their associated metadata that are encapsulated with that baseline.

Although, baselines as containers also incorporate design parts and requests, these are not currently replicated.

Consequently, the mechanism for baseline replication is very similar to that for item replication. The underlying transfer process makes use of temporary projects that Replicator creates and then deletes after the replication process has been completed. These temporary projects are populated with the contents of the baseline that matches the replication policy, and then the replicated baseline is created using the contents of these projects. By using temporary Dimensions projects that are maintained for the scope of the replication process, Replicator is able to both separate the process of replication from any other 'live' projects, and, also, provide the capability of resuming the replication process should any failures occur. For example, if a baseline contains 8000 items and 300 of them fail to replicate successfully due to non-existent item types, when the replication process is restarted, only 300 items need to be replicated before the baseline is created. All these temporary projects have the naming format

```
<product_id>:<TEMPREPL(repl primary uid)_(repl replica uid)!
    (a sequence number)>
```

New default replication behavior

Dimensions CM 2009 R1 or later includes new default replication behavior, first made available as an earlier customer-specific patch, namely:

> When items are dumped as part of baseline replication all relationships between the item and change requests, design parts and other items are also replicated. In

addition the related items (and any required placeholder items) are also dumped as objects—this does not apply to the related design parts or change requests. The relationships of any placeholder items dumped are not dumped.

With respect to the items replicated as part of baseline replication, the items are replicated with all relationships to:

- Design parts
- Change Requests
- Items

If these relationships are not required, or are not applicable to the receiving system then you are advised to do one of the following:

- When dumping the baseline, specify the `-dont_replicate_item_item_rels`, `-dont_replicate_item_request_rels`, `-dont_replicate_item_part_rels options` (or the single option `-dont_replicate_item_all_rels`).
- When loading the baseline specify the `-ignore_related_errs` option.

Request Replication

Replicator also enables the replication of Dimensions CM requests between development sites. In this case, you can specify one or more request types in the replication configuration, and all the requests of those types are replicated between the primary and replica sites. However, at any one time, only one site has ownership of any given request. Other sites can only view its details but cannot update it. The ownership of a request can be delegated to another site by the site that currently owns it. Ownership can also be requested by another site that can see a replicated copy of the request in their pending list. Ownership is transferred when the next replication cycle is run.

The following information is transferred when a request is replicated:

- The associated metadata, user attributes, file attachments, action and detailed descriptions.
- By default, any relationships to other Dimensions objects, such as parts, items, or other requests.
- Any related items and requests.

Related design parts and baselines are not replicated. By default, they must already exist on the sites that receive the request. However, you can override this behavior when you run the Replicator command.

Before trying to replicate a request to a recipient, Replicator first transfers any related items or requests to the recipient site. If any of these transfers fail, the replication of the request also fails.

When all of the requests have been processed, Replicator resets the ownership of any requests that have been delegated or requested.

New default replication behavior

Dimensions CM 2009 R1 or later includes new default replication behavior, first made available as an earlier customer-specific patch, namely:

The following new replicator option is enforced, this may not be altered:

```
-dont_replicate_item_request_rels
```

This ensures that requests related to items performing item replication are not replicated.

Processing Items or Requests That You Do Not Locally Own

Dimensions CM 2009 R1 or later includes the ability to process items or requests that you do not locally own.

To enable the processing of items or requests that you do not locally own, add the following settings to the server `dm.cfg` file:

```
DM_DONT_ENFORCE_ITEM_OWNERSHIP  Y
DM_DONT_ENFORCE_REQ_OWNERSHIP   Y
```

CAUTION! These server settings enable concurrent updates of objects between different sites. This may cause data loss.

When the item ownership flag is set, the following operations are permitted on an item that is not locally owned:

AI	Action item.
UIA	Update item attributes.
RII/XII	Relate/unrelate item to item.
RIP/XIP	Relate/unrelate item to design part.

Once the request ownership flag is set, the following operations are permitted on a request that is not locally owned:

RICD/XICD	Relate/unrelated item to change request.

Changes to Administration Console Behavior

By default, when configuring a project replication remote replica, the Administration Console prevents the selection of any version branch names that are not owned by the replica being configured. To enable the Administration Console to bypass this ownership check when the `DM_DONT_ENFORCE_ITEM_OWNERSHIP` and `DM_DONT_ENFORCE_REQ_OWNERSHIP` parameters are set, add the following setting to the server `dm.cfg` file:

```
DM_REPL_NO_PROJECT_BRANCH_CHECK   1
```

The remote replica site can then be suitably configured to allow item and request updates to be replicated back to the site which owns them.

In addition to a suitable configured configuration, it is also necessary to use the following `replicator` option:

```
-replicate_remote_branches
```

to replicate branches not owned by the sending site. This option should be used when dumping items that are on a version branch that is not owned by the "sending" site.

Integrating Dimensions RM and CM in a Replicated Environment

Currently Dimensions RM does not have the capability to replicate Dimensions RM requirements from one repository to another; however, by carefully coordinating Dimensions requests, you can still fully track the fulfillment of requirements in a replicated Dimensions environment.

To track work on requirements across replicated CM servers:

1. Install Dimensions CM and Dimensions RM, as described in their respective installation guides. Then set up the replicated CM environment, as described in this section.
2. For each requirement on the primary Dimensions server, create a request, and relate it to the requirement.
3. As needed, create related child requests for these requests and replicate them to the other replica Dimensions sites. As work is completed on the child requests at the replica sites, history is tracked by the parent requests on the primary server, provided Replicator is run. You can continue to review the requirements associated with the parent requests, allowing you to display detailed implementation history for each requirement.

Getting Started

The following sequence of instructions enables you to 'get started' with your replication scenarios after you have successfully transparently installed Replicator as part of a Dimensions server installation.

IMPORTANT! Although Replicator is automatically installed as part of a Dimensions server installation, it requires an additional license before you are able to use it.

Familiarization

See Chapter 24, "Online Replication Concepts and Scenarios" on page 319 before you attempt any work on machines that are connected together on a network. This section provides user scenarios to explain the basic conceptual framework underlying Replicator. It introduces such terms as Replication Configurations, Primary DB site and *Replica* DB sites. It also discusses the mechanism used by Replicator to replicate items from the Primary DB Site to Replica DB Sites and back.

See Chapter 24, "Online Replication Concepts and Scenarios" on page 319 and Chapter 25, "Offline Replication Concepts and User Scenarios" on page 341 before you attempt any work on machines that are physically isolated (so-called "air gap replication"). Many of the concepts introduced in Chapter 24, "Online Replication Concepts and Scenarios" on page 319 are also applicable to offline replication, and those particular to offsite replication are introduced in Chapter 25, "Offline Replication Concepts and User Scenarios" on page 341.

Refer also to:

- Chapter 26, "Preparing for Replication".
- Chapter 27, "Replication Administration" on page 355
- Chapter 28, "Replication Rules and Constraints" on page 371.
- Appendix D, "Replicator Utilities".
- Appendix E, "Replicator Error and Warning Messages".

Planning your Replication Scenario

Item Replication

Having familiarized yourself with the concepts, you should be able to plan your item replication scenario using the examples given in Chapter 24, "Online Replication Concepts and Scenarios" on page 319.

As part of your replication scenario you must nominate a primary DB Site and one or more replica DB Sites. Remember that you must install Replicator on the other replica DB sites as well as on the primary DB site.

You also need to identify the following:

- The Dimensions CM project on the primary DB site receiving/sending item branches to/from replicas.
- The names of version branches on the primary DB site to be replicated to the replicas.
- The names of version branches on each replica DB site which can send item branches back to the primary DB site.
- The project on each replica DB site receiving/sending item branches to/from the primary.

Ensure that the replica projects exist on the replica DB sites. You may need to contact the manager for each replica site to verify this. Replicator terminates with a failure if the designated replica projects do not exist.

Baseline Replication

The process of baseline replication is controlled through the use of a replication configuration definition that is almost exactly the same as that defined for replicating items. However, as source and target projects are irrelevant for baselines, the configuration definition comprises:

- A nominated primary.
- A list of replica databases.
- The names of version branches on the primary that are to be replicated.
- An optional list of item types that can be used to filter the type of items to be replicated.

Request Replication

The replication configuration definition for Dimensions CM requests comprises:

- A nominated primary site.

- A list of replica base databases.
- An indicator for each replica specifying whether the replicated requests are reference only. If they are reference only, they can only be viewed by the replica site, and ownership cannot be requested.
- A list of requests types to be replicated.

Before attempting any replication, check that the recipient contains the same design parts that are related to your requests. Replicator does not replicate related design parts, therefore, if these parts are not present, the replication fails.

Defining an Item Replication Configuration

To implement your item replication scenario, you need to create a replication configuration using either:

- The Replication administration cluster of the Administration Console (see Chapter 27, "Replication Administration" on page 355); or
- The `dmpmcli` scripting interface (see the *online help* and the example scripts in the `examples` subdirectory of the `AdminConsole` directory located in the Dimensions home directory).

You are only required to define this configuration on the primary DB site because the configuration is replicated to each replica as and when necessary. The primary DB site is allowed to update the replication configuration, but any replica sites can only browse it. Thus the primary DB site is in complete control of the configuration.

Only the manager of the primary DB site is permitted to define and update replication configurations.

To create the replication configuration for your scenario, use the Administration Console, Dimensions networking commands, or the `dmpmcli` scripting interface as follows:

- **On the primary DB site:** Register all the physical node names of the replicas, if not already entered. For each node registered, also register the Dimensions base database that is to act as a replica DB. Flag these replicas as `Workset Replication Enabled`.
- **On each replica DB site:** Register the primary DB site and, if not already entered, the replica DB details. After this is done, ensure that they have been flagged as `Workset Replication Enabled`.
- **Against the primary DB site:** Create your replication configuration by defining the primary details and the replica details. You can define more than one replica if your scenario requires this.

Refer to Chapter 24, "Online Replication Concepts and Scenarios", Chapter 26, "Preparing for Replication", and Chapter 27, "Replication Administration" for more details and examples.

NOTE Replicator *only* supports the use of physical network nodes. Logical nodes are *not* available for replication purposes.

Defining a Baseline Replication Configuration

Baseline replication is very similar to implement in most respects to item replication. You define a baseline replication configuration as you do for item replication, which contains a primary and a list of replica sites involved in the replication. Unlike item replication, this configuration specifies only the base database and version branches that are involved in replication. Source and target projects are not used.

Refer to Chapter 24, "Online Replication Concepts and Scenarios", Chapter 26, "Preparing for Replication", and Chapter 27, "Replication Administration" for more details and examples.

Defining a Request Replication Configuration

For request replication, you define a replication configuration as you do for item replication, except you only specify base databases and the list of request types to take part in the replication.

See Chapter 24, "Online Replication Concepts and Scenarios", Chapter 26, "Preparing for Replication", and Chapter 27, "Replication Administration" for more details and examples.

Ensuring the Minimum Level of Compliance between Process Models

Please note that a recipient DB site can only accept items from a sender if the following minimum criteria are met:

- The product owning the item, baseline, or request on the sender also exists on the recipient.
- The type of the item, baseline, or request to be received (for example, SRC or OBJ) must also exist on the recipient and be assigned against a lifecycle.
- An item library has been defined on the recipient for each type of item that is to be received.

NOTE You can use a default item library if you wish.

Refer to Chapter 28, "Replication Rules and Constraints" on page 371 for details on the rules used by a recipient when receiving an item via replication.

Defining a Temporary Work Area

For the primary DB site as well as all the replica DB sites, you can control the temporary work area that Replicator is to use when sending and receiving items by setting the variable

```
DM_REMOTE_FILE_AREA
```

on each primary and replica machine.

This variable is defined in the file `dm.cfg`. This file is located at the top level installation directory for Dimensions.

Replicator does not run if this variable is not set.

Example: `DM_REMOTE_FILE_AREA d:\replicator_temp\`

NOTE On Windows, by default this variable is defined to use an area within the `<DM_ROOT>` directory. On UNIX, by default this variable is set to `/tmp`.

You can redefine this temporary area to use another file area.

Online Replication Network Objects

Online replication has a client-server architecture. It uses a TCP/IP-based communication protocol as the communication layer between the primary and the replica sites. This communication occurs over the standard Dimensions network object (TCP/IP object) `pcms_sdp`.

For more information on defining network objects, please refer to Part 4, "Network Administration".

The network object name and socket number used *must* be common to all the nodes participating in replication (that is, the primary and replica sites). If they are not, the primary fails to connect to the replica sites.

Running Replicator

Initiate the replication process from the command line as described in Chapter 29, "Running Replicator" on page 379.

Refer to Appendix E, "Replicator Error and Warning Messages" on page 511 for a list of Error Messages with their causes and recommended actions.

Replication Results

To check the results of replication from or to a DB site, use the Replicator log viewing functions of the Administration Console, see Chapter 27, "Replication Administration" on page 355 for details.

Replicator Published Views

A set of published views for Replicator is available through which third-party report writers may be used to generate custom reports. Please refer to the related document *Reports Guide* for further details.

Chapter 24

Online Replication Concepts and Scenarios

Basic Concepts	320
Local Item Replication	323
Remote Item Replication	323
Remote Item Replication Example Scenarios	331
Baseline Replication	334
Request Replication	336
Request Replication Example Scenarios	337

Basic Concepts

The main function of online replication is to facilitate parallel development of items, baselines, and requests across different networked geographical locations (as distinct from physically isolated locations) where some or all of the sites have their own local repositories for performance or operational reasons. Currently Dimensions Replicator supports only the replication of Dimensions CM requests, items, and baselines.

IMPORTANT!

- Replicator does not support the use of Dimensions CM streams.
- Replication of design parts is not supported. A separate data transfer utility, "PDIFF Import/Export", is available for transferring design parts between databases. See Chapter 16, "Data Interchange File Format (PDIFF)" and Chapter 17, "Using PDIFF to Import/Export Data" for details.
- Replication of requirements information from one Dimensions CM server to another is not supported in Dimensions CM. However, by carefully coordinating Dimensions CM requests, you can still fully track the fulfillment of requirements in a replicated Dimensions environment. See, "Integrating Dimensions RM and CM in a Replicated Environment" on page 313.

Related Dimensions Concepts

This topic explains some of the main Dimensions CM concepts. For a more detailed description of the concepts, see the *Dimensions CM online help*.

A Dimensions CM server installation typically consists of a *relational database* and a set of *item libraries* that can optionally be distributed across several machines. The item libraries are secure operating system directories containing the files under source code control. The database largely contains information about the files in the libraries including:

- How the different revisions of each file are related to each other.
- System and user-defined attributes.
- A complete audit trail (for example, who updated what file and why).
- Relationships to other project objects such as design document and bug reports.

The database also contains important work project and design information about the software product under development as well as security and change authorization policies set up by the manager to control project access and to define who has what role on the project.

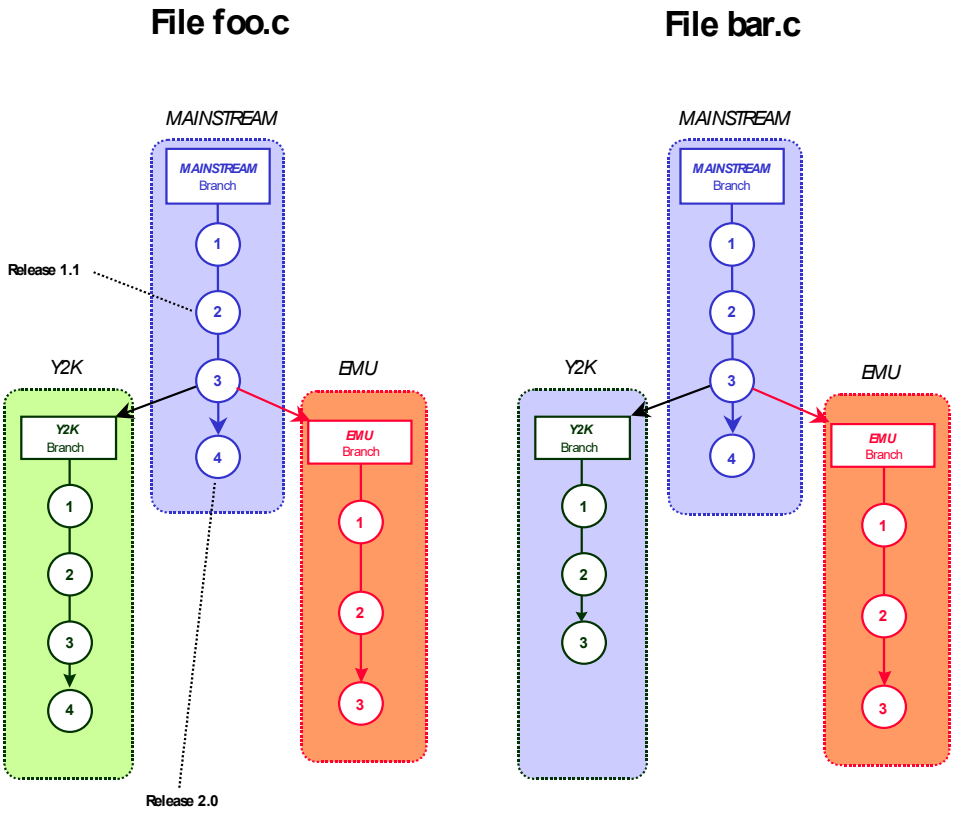
When the same Dimensions CM database is to be used to store data for more than one work project, the database is divided into several partitions, called *Dimensions base databases*, one for each project. Each base database has its own separate associations with item libraries and contains metadata as well as other types of work project data about the items stored in those libraries.

A user interacting with a Dimensions client interface must always first connect to a base database for which they have access authorization. This base database may be local or remote to the machine to which the user is logged in. Dimensions requires that users perform their version management functions, such as check out and check in, in the

context of a Dimensions CM project. A project is essentially a container for items and their revisions. It provides a well-defined and auditable context for users to perform version management functions. A project grows when users create new item revisions while working in it, and shrinks when users either explicitly remove item revisions from it or physically delete item revisions from the database. In Dimensions, projects provide complete segregation of parallel development activities that may be carried out as part of mainstream, maintenance, or custom development.

An item may be revised many times into different versions within different project contexts throughout its development. Figure 24-1 below shows examples of version trees where Revisions 1 and 2 of item `foo.c` were created for the development of Release 1.1, and Revisions 3 and 4 were created at a later date for Release 2.0. The version tree of an item can normally be broken into separate *branches*, each branch containing a specific chain of revisions that were created in the context of a specific development goal, for example, `mainstream`, `y2k maintenance`, and `emu maintenance`. It is important to note that a branch name may be shared across several items, for example, items `foo.c` and `bar.c` may both have revisions that belong to a branch with the name `emu`, for example, revisions `foo.c; emu#1`, `foo.c; emu#2`, `bar.c; emu#1`. For the purpose of this document, the term *named branch* is used to refer to the collection of all item revisions that belong to branches that share a common name. For example, the named branch `emu` refers to the collection of all revisions of all items that belong to an `emu` branch.

Figure 24-1. Version Trees, Releases and Branches



A Dimensions baseline is like a Dimensions project except that it is a static view of the contents of the work product/project made at a certain time. There are two types of baseline: a design baseline, which is a snapshot only; and a release baseline, which represents a code milestone. It is this release type of baseline that Replicator can transfer from site to site.

A Dimensions request is used to report a defect, suggest an enhancement, or detail other work for a particular product. Requests are of different types, and are logged in the database in relation to one or more item revisions that need to be changed. Requests have relationships with items, baselines, other requests and design parts. In order for these related objects to exist on the receiving site, the relationships must be maintained when a request is replicated.

Types of Synchronization and Replication

Dimensions supports the following synchronization or replication mechanisms:

Item Synchronization	Enables you to work with item files in your local work area outside the Dimensions repository.
Local Item Replication	The transfer of Dimensions items from and to Dimensions projects that have all been defined within the *same* base database. In this mode only pointers for items are added to the target project. No physical transfer of items takes place. Refer to the next section below for further details.
Remote Item Replication	The transfer of items from and to projects located in base databases that are *different from* that where the item originated. In this mode item revisions are physically copied by replication from the source project to a target project where the source and the target base databases are separate. Refer to "Remote Item Replication" on page 323 for further details.
Baseline Replication	The replication of Dimensions baselines from a sender site to one or more replica sites.
Request Replication	The transfer of Dimensions requests from and to a sender site, and one or more replica sites.

Item Synchronization

NOTES

- Item synchronization does not use Replicator. Also, there is no equivalent synchronization mechanism for baselines or projects.
- Your current Dimensions CM license is sufficient to run item synchronization.

Dimensions for Windows provides a Synchronize Wizard that enables you to synchronize operating system files and their status in relation to the Dimensions repository. This can be run from within either the Dimensions desktop client or Windows Explorer. For details, see the Dimensions CM online help.

Local Item Replication

NOTES

- Local item replication *does not* use Replicator. Also, there is *no* equivalent local replication mechanism for baselines or projects.
- Your current Dimensions license is sufficient to run local item replication.

Local item replication is automatically triggered and takes place synchronously when one of the following Dimensions operations is performed on an item revision belonging to a named branch for which local item replication has been setup.

- **Create Item** (CI)
- **Check In Item** (RI)
- **Update Item** (UI)
- **Update Item Attributes** (UIA)
- **Delete Item** (DI)
- **Action Item** (AI)
- **Suspend Item** (SI)

Local item replication automatically causes the addition or removal of item revisions to the Dimensions projects nominated to take part in replication. This is achieved by copying pointers without any physical transfer.

Remote replication of items also triggers local replication in the recipient base database.

IMPORTANT! A particular item revision may have been imported into several local projects either manually or as a result of project replication (it should be remembered that a project is basically a logical grouping 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 is updated. Conversely, if you edit the content of an item revision in one project and change its revision, the changes in content are reflected only in that project and any project replicated from it.

Remote Item Replication

NOTES

- Remote item replication *does* use Replicator. Also, there *are* equivalent remote replication mechanisms for baselines and projects.
- Your current Dimensions license is not sufficient to run remote item replication. You need an additional Replicator license.

Remote item replication is said to be in place if the target project resides in a different base database to the source project. A target base database may reside:

- Within the same database instance as the source base database.
- Within a different database instance to that hosting the source base database, which is either on the same machine or on a remote machine accessible via a LAN or WAN.

Remote item replication is designed to support teams of developers situated at different geographical sites to work on the same software product, either in parallel or in collaboration. Each site can have its own local base database and item libraries that Replicator synchronizes with the base databases and item libraries on the other sites. Replicator supports remote development by:

1. Allowing the project administrator to nominate a Dimensions project and a list of named branches in that project from which item revisions can be replicated to selected projects in remote base databases.
2. Physically replicating item revisions belonging to the nominated named branches from the project to the selected remote projects.
3. Segregating the development activities that have been carried out at the different sites by way of assigning ownership of sites to the named branches, for example, London own the `y2k` branch, Frankfurt the *`emu`* branch and New York the `mainstream` branch. The rule is: a named branch is owned by only one of the base databases participating in replication, and only users of that base database are allowed to modify that branch (refer to "Site Ownership of Named Branches" on page 324 for additional information).
4. Providing a comprehensive branch merge facilities for merging "remote" and "local" branches; for example `emu` and `y2k` branches could be merged into `mainstream`.

Primary and Replica Base Database Sites

When an item replication process is initiated, Dimensions items are copied from a source Dimensions project to a number of target projects. The location of these projects within the context of the replication environment is specified by defining a base database *site* that is then used to locate that project. This *site* definition consists of a physical network node name, a Serena-Supplied Runtime RDBMS or Oracle Enterprise database instance identifier (SQL Server is not supported by Replicator), and a Dimensions base database name.

The term *Primary Base Database* is used to refer to a Dimensions base database from at item named branches are to be replicated to other Dimensions base databases.

The term *Replica Base Database* is used to refer to a Dimensions base database that is to receive the replicated named branches and which may, optionally, replicate other branches back to the primary.

Thus a primary base database always replicates named branches into its replicas and may receive other named branches back from those replicas. Furthermore, the primary and replica base databases may reside in different Oracle databases located at the same or different geographical locations.

The terms *Pirmary DB Site* and *Replica DB Site* are used to refer respectively to the sites upon which the primary base database and replica base databases are located.

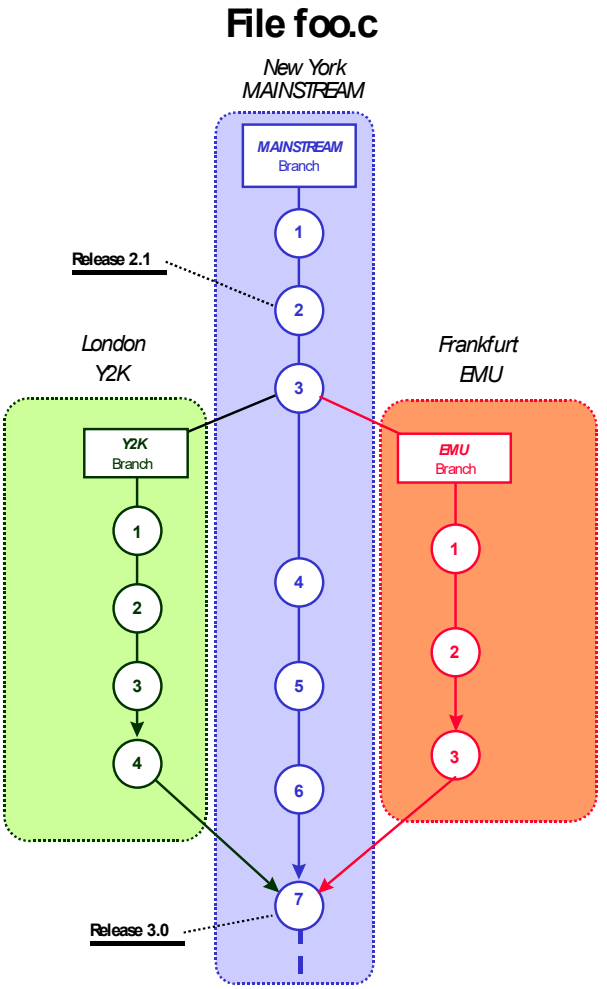
Site Ownership of Named Branches

A named branch is said to be owned by a Dimensions base database if the manager for that base database explicitly creates the named branch in that base database by the standard Dimensions Administration tools. A named branch is said to be a *replicated branch* if it is created implicitly as part of a replication process.

Within a replication configuration, a named branch must be owned by only one base database, and only users of that base database are permitted to create new versions on that named branch. These restrictions ensure complete segregation of development activities in a distributed, parallel, development environment. In practice, this is achieved by locking the replicated branch and its constituent item revisions on the recipient site. The lock prevents the users on the recipient sites from modifying the item revisions on that named branch.

Consider, for example, Figure 24-2 below. This shows the version tree of the file `foo.c` developed in parallel in New York, London and Frankfurt. New York took the lead in developing `foo.c` and thus owns the `MAINSTREAM` named branch; whereas, London and Frankfurt were in charge of Year 2000 and European Monetary Union aspects impacting on the development of `foo.c` and therefore own the named branches `Y2K` and `EMU` respectively. New York is also responsible for merging London's and Frankfurt's changes to `foo.c` back into those it made itself in the named branch `MAINSTREAM`, together with resolving any merge conflicts.

Figure 24-2. Multiple Named Branches



When a named branch is replicated from one DB site to another DB site, the following checks and actions are taken to ensure that only one DB site owns a named branch.

- A primary DB site can replicate any named branches regardless of whether or not these branches are owned locally by the primary or have been replicated remotely to the primary. However, the replica DB site can replicate back only those named branches that it owns and cannot replicate back remotely received named branches.

This model therefore enables a primary to receive a named branch from its replica DB site and then act on this named branch; actions could include merging item revisions into the original base named branch or replicating it to other DB sites.

- If the named branch does not exist on the recipient, then as part of the replication process such a named branch is created automatically on the recipient. It is also registered as a replication branch originating from the sender.
- If the named branch already exists on the recipient, but the recipient does not own it, then replication is allowed only if the branch was originally created by the sender. If the branch was created by a different sender, a warning message is generated and replication to this recipient is terminated.

For more replication rules concerning system and user-defined attributes and relationships, see Chapter 28, "Replication Rules and Constraints".

Replication Configurations

Replication configuration is the term used to refer to the setup information that enables the primary DB site to control and scope the replication process.

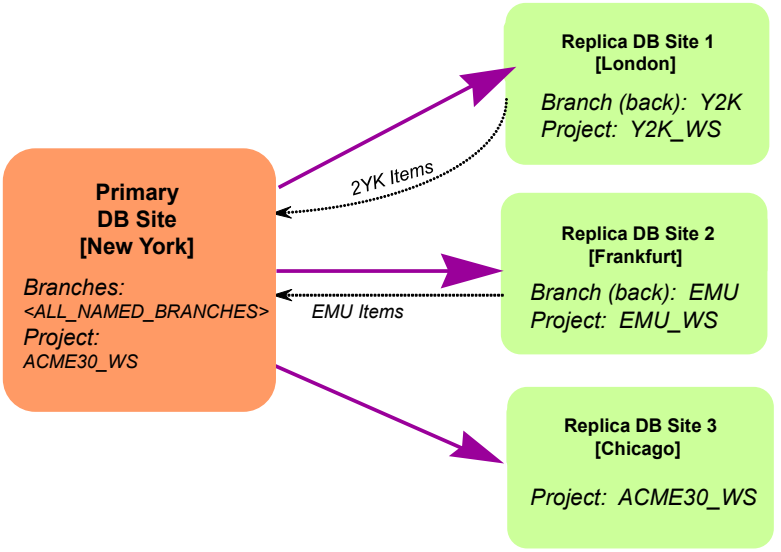
NOTE Replication configurations are defined solely on the primary DB site (by that site's replication manager). Administrators on the replica DB sites are not required to set up such configurations. Instead, each time the primary DB site initiates a replication transfer the configuration definition is copied to that replica, but *only* if that configuration has been updated since the last transfer.

A replication configuration enables the following.

- It identifies a collection of base database sites, of which one site acts as the primary base database and the rest act as replica base databases.
- For items, it identifies:
  - A set of named branches in a project in the primary base database from which items are to be replicated to the replica base databases.
  - For each replica, the project on the replica into which the replicated items are to appear, and from which items are to be replicated back to the primary.
- For requests, it identifies:
  - The request types in the primary database that are to be replicated to the replicas.
  - The request types in the replica that are to be replicated back to the primary.
- It specifies whether the primary base database has items or requests replicated back to it from the replica base databases. For each replica, the primary defines named branches for items the replica replicates back to it.

For example, consider Figure 24-3, "An Example Replication Configuration for a Primary DB Site at New York," on page 327. This illustrates a primary DB site (New York) replicating items to three replica DB sites (London, Frankfurt, and Chicago).

Figure 24-3. An Example Replication Configuration for a Primary DB Site at New York



In this example:

- New York, as the primary DB site, is replicating all named branches that it owns *locally* in the project `ACME30:ACME30_WS` to all the replica sites. It does this by specifying the pseudo-named-branch `<all_named_branches>`. (Note that the product-id part of the project specification – *ACME30* – has been omitted from Figure 24-3, "An Example Replication Configuration for a Primary DB Site at New York," on page 327 for the sake of brevity.)
- London receives the replicated files into its local project `ACME30:Y2K_WS`. It has created a named branch `Y2K` in which it performs its local development on those items. Because London can only browse replicated items, any development work is undertaken on the `Y2K` branch and then replicated back to New York.
- Frankfurt receives the replicated files in similar fashion to London. Its local project and named branch for replicating back development are `ACME30:EMU_WS` and `EMU` respectively.
- Chicago receives the replicated files into a local project `ACME30:ACME30_WS`. However, it does not intend to replicate any item revisions back to New York and consequently it has not set up a named branch for that purpose. Therefore, the replication configuration at New York is not aware of any named branches at Chicago.

NOTE Named branches are defined globally across Dimensions base databases. For details about named branches, see the Dimensions CM online help.

Invoking Remote Replication

The Replicator product for remote replication includes:

- A *replicator* utility (see "Launching Replicator from the Command Line" on page 380).
- The Administration Console for defining replication configurations and browsing logs generated by the replication process (see Chapter 27, "Replication Administration").
- The `dmpmcli` scripting interface as an alternative means for defining replication configurations and browsing logs generated by the replication process. See the *online help* and the example scripts in the `examples` subdirectory of the `AdminConsole` directory located in the Dimensions home directory.

The main input to Replicator is the identifier of the replication configuration that defines what to replicate. When connected to a base database, Replicator treats it as the primary base database and triggers named-branch replication to its remote replica base databases according to the rules defined in that specified replication configuration. If, as part of the replication definition, named branches need to be replicated back to the primary, then Replicator also performs that task.

Replicator has a client-server architecture. It uses the TCP/IP communication protocol. When replication is initiated from a primary database, it:

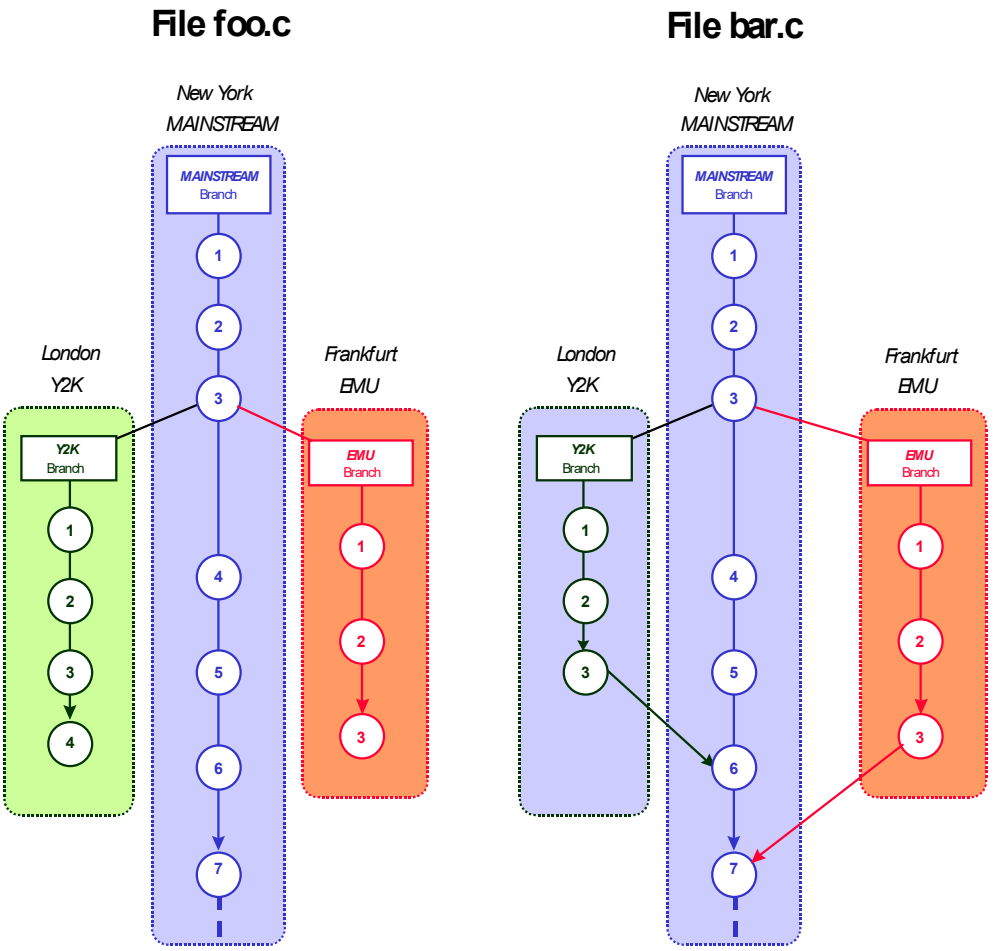
- Connects to each replica base database in turn.
- Receives back replicated item revisions from them (where such back replication has been specified).
- Replicates its synchronized and updated named branch changes to each replica.

A connection request from a primary to a replica machine is processed by a listener daemon on the replica. This listener is dedicated to listen to a predesignated TCP/IP port (socket). The primary uses this port to establish a connection with the replica for subsequent to and from replication of items.

Determining which Item Revisions to Replicate

When remote replication has been set up by defining appropriate replication configurations, Replicator must determine what item revisions are to be replicated from the source primary project to each replica target project in a given replication configuration. The algorithm used to implement this is described with the example shown below. This example illustrates the item revision pedigree (history) of two source files `foo.c` and `bar.c` which, while primarily controlled by New York, have been replicated to London and Frankfurt for Year 2000 (`Y2K`) and European Monetary Union (`EMU`) development.

Figure 24-4. Example Item Pedigrees of Remotely Replicated Files



For `foo.c`, New York has not yet merged and conflict-resolved any changes from the back-replicated item revisions developed on the remote `Y2K` or `EMU` named branches. The following revisions of `foo.c` is replicated from New York to London and Frankfurt at the next invocation of Replicator:

```
mainstream#1, mainstream#2, mainstream#3, mainstream#4,
    mainstream#5, mainstream#6 and mainstream#7
```

For file `bar.c`, New York has taken the changes from the back-replicated item revision `y2k#3` (from the remote named branch `Y2K`) and merged, plus resolved the conflicts for those changes together with its own item revision `mainstream#5` to create `mainstream#6`. It subsequently merges and resolves conflicts in `mainstream#6` with changes from the back-replicated item revision `emu#3` (from the remote named branch `EMU`) to create `mainstream#7`. The following item revisions of `bar.c` are then replicated from New York to London and Frankfurt at the next invocation of Replicator:

```
mainstream#1, mainstream#2, mainstream#3, mainstream#4,
    mainstream#5, mainstream#6 and mainstream#7
```

NOTE Replicator preserves merge history information wherever plausible. When `bar.c` is replicated to London and Frankfurt, these sites can see from the item revision pedigree that New York has absorbed their `Y2K` and `EMU` changes respectively into the `MAINSTREAM` named branch.

Item Revision 'Holes'

Up till now, it has been assumed that there is a one-to-one mapping of item revisions on a named branch to those that appear in the project. For `foo.c` we have been assuming that `mainstream#1`, `mainstream#2`, `mainstream#3`, `mainstream#4`, `mainstream#5`, `mainstream#6`, and `mainstream#7` all appear in the project to be replicated (`ACME30:ACME30_WS`). It is possible, however, that the project may contain some revision "holes", for example, `ACME30:ACME30_WS` might only contain the following revisions of `foo.c mainstream#1`, `mainstream#3`, and `mainstream#5`.

If there are revision "holes", then Replicator adopts the following algorithm (we use the scenario discussed above for `foo.c` as an example):

- New York
  - Project `ACME30:ACME30_WS`

    Contains `mainstream#1`, `mainstream#3`, and `mainstream#5`, that is, has "holes" for `mainstream#2` and `mainstream#4` and does not contain `mainstream#6` and `mainstream#7`.
  - Global Project

    Contains `mainstream#1`, `mainstream#2`, `mainstream#3`, `mainstream#4`, `mainstream#5`, `mainstream#6`, and `mainstream#7`.
- Recipient of Replicated Item Revisions (London and Frankfurt)
  - Project Specified in Replication Configuration

    After replication, the recipient project contains `mainstream#1`, `mainstream#3`, and `mainstream#5`, that is, has "holes" for `mainstream#2` and `mainstream#4` and does not contain `mainstream#6` and `mainstream#7`.
  - Recipient Global Project

    Following successful replication this contains `mainstream#1`, `mainstream#2`, `mainstream#3`, `mainstream#4`, `mainstream#5`, but does not contain `mainstream#6` and `mainstream#7`. Those replicated items are known in the log as placeholder items and serve to maintain the pedigree of the replicated items. The only difference between a placeholder item and a non-placeholder item is that the former is included only in the global project of the recipient, whereas the latter is also included in the recipient target project specified by the replication configuration.

To summarize, item revision "holes" in the sender's primary project are replicated as placeholder items from the sender's global project to the recipient's global project. However, any revisions in the sender's global project later than the last one to appear in the primary project are not replicated to the recipient's replica project.

Replication of Updates to Item Metadata

Replicator enables the re-replication of changes to an item's metadata to replica sites that have already received that item. For example, after an item has been replicated from one site to another, updating the attributes of that item or actioning it results in those changes being re-replicated to those replicas. This means that the replica sites are kept in sync with any changes that happen on the site responsible for that item.

Remote Item Replication Example Scenarios

General Guidelines

Before a replication configuration can be set up, you need to perform certain administrative tasks. This section outlines the general steps that the manager on the primary DB site needs to perform.

NOTE The definitions in this topic have to be performed only on the primary DB site, not on the replica. These definitions are replicated automatically to replica base database sites, so the primary is in complete control of what it replicates and receives.

1 Define the replication strategy.

Spend some time doing an initial pen-and-paper exercise to get clear in your mind what the replication process is to be. You need to:

a Identify which replica DB sites are going to participate in replication.

b Identify the named branches and the project from which items are to be replicated from the primary.

c Liaise with the manager of each replica DB site to establish, for example, the following with respect to the replication process:

- The identity of the Serena-Supplied Runtime RDBMS or Oracle Enterprise database (SQL Server is not supported by Replicator) hosting the replica base database.
- The name of the replica base database.
- The name of the project on the replica into which items are to be received from the primary and the replica from which items are to be replicated back to the primary, if any.
- Branch names on the replica to be replicated back to the primary, if any.

Having decided what you want the replication process to achieve, proceed to implement it as follows.

2 Register the sites participating in replication (Chapter 26, "Preparing for Replication").

A prerequisite for replication is that the sites to be designated as primary and replica DB sites are registered with respect to one another and be declared as replication sites. Only a site declared as a replication site can participate in replication. This can be done by using the Administration Console, Dimensions networking commands, or the `dmpmcli` scripting interface.

The Dimensions networking commands are fully described earlier in this guide and the `dmpmcli` scripting interface is described in the *online help*.

3 Identify primary and replica DB sites (Chapter 26, "Preparing for Replication").

Once sites have been registered, you need to specify which site is going to be the primary DB site from which replication is to be driven and which are going to be replica DB sites for the replication strategy you want to implement. This is done as part of defining a replication configuration ID. These nominations can change from one replication configuration ID to another.

4 Identify what the primary is to send to each replica and what it is to receive from each replica.

As a further part of defining a replication configuration ID, you must identify what the primary DB site is to send to each replica DB site and vice versa.

In the primary base database, you declare: the replication source named branches; the source project; the target project in each replica; and, optionally the named branches to be replicated back from each replica.

In each replica base database, ensure that the corresponding target projects exist and the named branches to be replicated back also exist. When the primary's named branches are replicated to the replicas they are flagged as replicated branches on the replica databases; similarly, when named branches on the replica are replicated back to the primary, they are flagged as replicated branches in the primary's database.

Replication entails the following actions.

a For each replica that has been set up to replicate back to the primary, the replication process first replicates the specified named branches to the primary. This is deliberately done before data is replicated to the replicas so as to enable the primary to "push" any data it receives from replicas to all other designated replicas. Note that when back replicating a named branch from a replica to a primary, only item revisions created since the previous replication are transferred.

b For each replica defined in the replication configuration, the primary replicates its designated named branches to this replica. Only those item revisions that are new since the previous replication are transferred.

5 Run the replication process (Chapter 29, "Running Replicator").

Once having registered the replication sites and created the replication configurations, you can initiate the replication process for a particular replication configuration. For each requested replication you can specify:

a Optionally, the name of the replica to which you want to replicate. If one is not specified, then the primary's default list is used. However, if one is specified (for example, `Node B`), then replication is targeted only at that node.

b Reporting options.

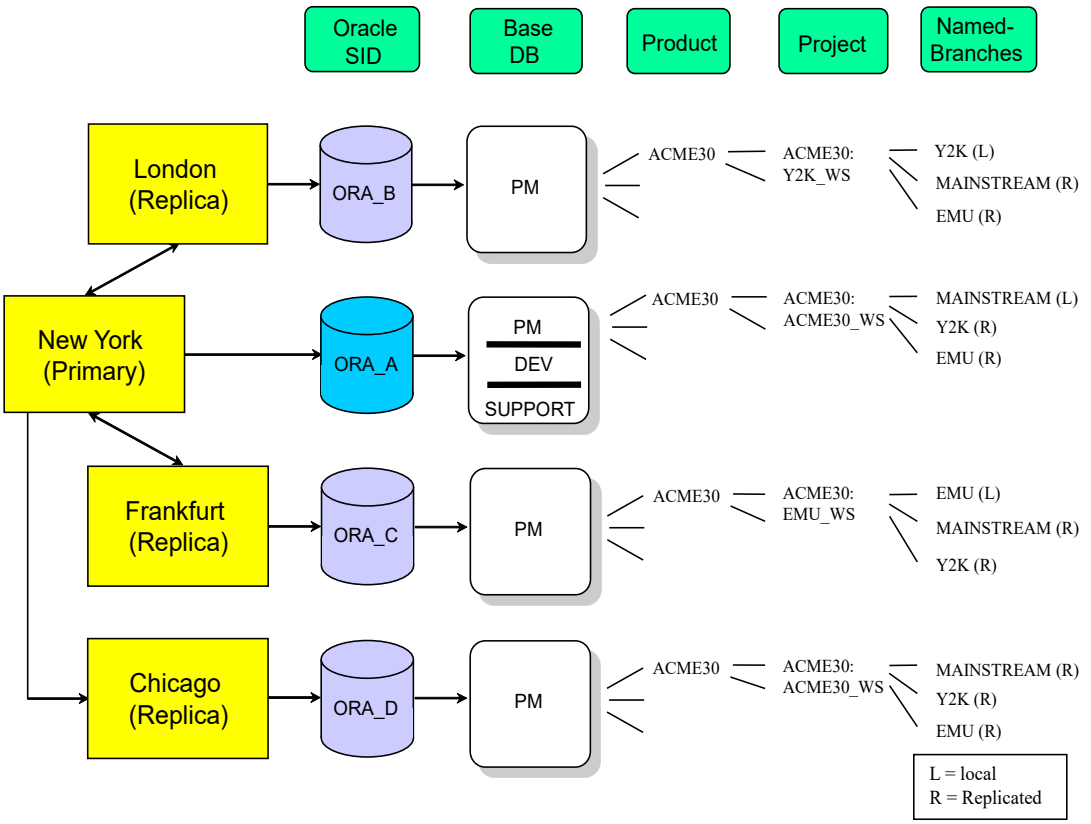
6 View Logs of items transferred and received by a base database (Chapter 27, "Replication Administration").

When a primary receives items from a replica and vice versa, a log of all items is received and transferred on both the primary and the replica together with any error conditions. The log is stored in both the primary and replica databases and can be queried using the Administration Console.

Dimensions-Specific Details

Here we take the user scenario and add some of the Dimensions-specific details—refer to the diagram on the following page, which illustrates an Oracle-based scenario.

Figure 24-5. Dimensions Specific Details



- New York Primary Node

  Let us assume that on the New York node there is an Oracle instance `ORA_A` that contains three Dimensions base databases, namely, `PM`, `DEV`, and `SUPPORT`. Within the base database `PM` is the product `ACME30` that contains the project `ACME30:ACME30_WS` and has the named branch `MAINSTREAM`. Replication has been enabled such that items on all named branches are replicated to the London, Frankfurt and Chicago nodes (by specifying the `<all_named_branches>` pseudo named branch).

- London Replica Node

  Let us assume that on the London node there is an Oracle instance `ORA_B` that contains the single base database PM. Within that base database is the Dimensions product `ACME30` that contains the project `ACME30:Y2K_WS` and has a named branch `Y2K`. Replication has been enabled such that items in New York's `MAINSTREAM` named branch are replicated to the London node for use in the locally named branch `Y2K`. Additionally, the results of development on the `Y2K` branch are replicated back to New York.

- Frankfurt Replica Node

  Let us assume that on the Frankfurt node there is an Oracle instance `ORA_C` that contains the single base database `PM`. Within that base database is the Dimensions

product ACME30 that contains the project ACME30:EMU_WS and has a named branch EMU. Replication has been enabled such that items in New York's MAINSTREAM named branch are replicated to the Frankfurt node for use in the locally named branch EMU. Additionally the results of development on the EMU branch are replicated back to New York.

- Chicago Replica Node

  Let us assume that on the Chicago node there is an Oracle instance ORA_D that contains the single base database PM. Within that base database is the Dimensions product ACME30 that contains the project ACME30:ACME30_WS.

NOTES

- While this scenario uses Y2K and EMU named branches in London and Frankfurt respectively, the involved projects can own other branch names as well. However, these branch items are outside the scope of Replicator.
- The development process model and procedures used at London and Frankfurt need not be the same as that at New York. This means that London can use its local process for development. However, there must be a matching of defined lifecycle item types, and various other replication rules must be observed (for example, concerning replication of system and user-defined attributes and relationships). See "Metadata Replication Rules" on page 375 for full details. These rules also apply to Chicago, albeit that no development is taking place there.

Setting Up

A subset of the preceding scenario is used as an example when discussing the various GUI-based forms of Replicator in this section:

- New York as a primary DB site replicating item revisions on the named branch MAINSTREAM to London as a replica DB site.
- London performing development in the named branch Y2K and replicating those item revisions back to New York.

An overview of this defining and setting up process is given in "Invoking Remote Replication" on page 328.

Baseline Replication

Similarly to remote item replication, baseline replication is said to occur when baselines, and all the associated items contained in those baselines, are replicated from a primary to one or more replica sites. This means that a release baseline that has been created on one base database can be propagated to many other geographical sites, representing a transfer of a known milestone of source code. In principle, baseline replication is very similar to item replication, except that the medium of the transfer used is not a project that might be frequently changed, but is instead, a fixed milestone that has its own associated attributes and audit trail. The advantage of using baseline replication over item replication is that it provides much higher control over exactly which items are replicated.

Primary and Replica Base Databases

In baseline replication, the concept of the primary and replica sites is exactly the same as they are in remote item replication. The primary base database owns the baseline to be replicated, and transfers this baseline to the respective replica sites contained within a replication configuration definition. Unlike remote item replication, however, baseline replication does not have the concept of replicating objects back. This means that all transfers are only performed from the primary to the replica base database.

Replication Configuration

A baseline replication configuration definition is slightly different from that used for item replication. While the details such as the primary base database, the list of replicas, and the branches to replicate are the same, there are a number of differences that differentiate these two types of replication configurations. Firstly, baseline configuration definitions do not require the setting of a source or target project on any of the sites involved in the replication process; and, secondly, the baseline replication configuration definition enables the setting of a list of optional item types that detail what sorts of items within the baseline are to be replicated.

Initiating Baseline Replication

The replication of baselines is initiated from the command line using the `replicator` executable.

When replication of a baseline is initiated, a temporary project is created on the replica site into which all the items involved in the baseline replication are placed. This temporary project is maintained until the baseline has been successfully transferred to all replica sites, whereupon it is automatically deleted.

Baseline Replication Item Transfer Failures

As in item replication, during baseline replication items can potentially fail to transfer to a replica due to a number of reasons. If an item fails to transfer, the baseline is not created on that replica site. If this happens, the replication logs can be used to determine why particular items may have failed to be transferred, and to determine any corrective measures to undertake before re-replicating the baseline.

In circumstances such as above:

- The temporary project created on the replica site is not deleted until all the baseline's items have been successfully transferred in later sessions. Any of the baseline's items successfully transferred before the failure, are stored in this temporary project pending successful transfer of the remaining items in later sessions. Only when all the baseline's items have been successfully transferred, the baseline is created and the temporary project are deleted.
- The replication configuration used to control the replication process is 'locked' so that it can only be used to finish that specific replication. The reason for this is that the same replication configuration can be used to replicate many different baselines, and to insure the integrity of the replication logs that are used to track the success or failure of any replication, a configuration, once used to initiate a baseline replication, is only freed for general use after that replication succeeds. If a baseline replication fails due to item replication failures, the same replication configuration can be used at

a later date to restart that specific replication, and only the previously failed items are reattempted.

When all baselines have been successfully transferred, the replication configuration used to control this transfer process becomes available for use with other baselines.

Differences between Replicated and Release Baselines

When a baseline is replicated to a replica site, there are a number of differences between this replicated baseline and a normal release baseline that you should be aware of:

- A normal release baseline, created using the Dimensions Create Baseline (CBL) function, includes all the design parts that are children of the design part that the baseline was first created against—whether or not those design parts have items related to them. In a replicated baseline, however, design parts are added to that baseline only if they or their children own any items.
- To differentiate replicated baselines from normal release baselines, the template-id is always set to `$$REPLICATED$$`.

Determining which Item Revisions to Replicate

The algorithm used to determine which items are to be replicated is the same as that used for remote item replication, except that the scope of the items considered is limited to the baseline being replicated. As release baselines only have one revision of each item, however, issues relating to revisioning 'holes' are not so relevant.

Replication of Design Parts to Replica Sites

As documented in Chapter 23, "Multi-Site Development Using Replicator", the design parts in a baseline are not replicated to the replica site. The reason for this is principally to do with the rights of the primary to dictate to the replica a project design part hierarchy that might conflict with one already in place. However, any items that are replicated must have their associated part existing on the replica. If this part does not exist, that item fails to be transferred and an appropriate error message is generated.

When the baseline is created, then any appropriate design parts, no matter where they are in the replica project hierarchy, are related to that baseline so that a valid design part hierarchy is maintained. These relationships, however, are only scoped to that baseline and are not propagated to the project hierarchy itself.

Request Replication

Request replication consists of copying the requests identified by a group of request types between the primary and replica sites. In addition, the reassignment of requests that have been delegated or requested is activated when replication is run.

The concepts of the primary and replica sites are the same as they are in remote item replication. The primary base database initiates the replication, and transfers requests to the respective replica sites contained within a replication configuration definition.

Request Replication Configuration

A request replication configuration is different from an item or baseline configuration as it enables you to specify a list of request types that are replicated between sites. As with items, you can replicate requests to and from replica sites. You can define these sites as reference sites (read-only access) or normal replica sites.

Reference Sites

One option that can be set up as part of a replication configuration definition is whether or not a particular site is to be a reference site. If a site is defined as a reference site, then requests replicated to that site cannot be worked on or requested by developers on that site. These requests are read-only copies of the replicate data.

Initiating Request Replication

The replication of requests is initiated from the command line using the `replicator` executable. The parameters are described in Chapter 29, "Running Replicator".

Request Replication Transfer Failures

As in item replication, during request replication objects can potentially fail to transfer to a replica for a number of reasons. If a request or its related items or requests fail to transfer, that request is not replicated to the replica site. If this happens, the replication logs can be used to determine why particular requests may have failed to be transferred, and to determine any corrective measures to undertake before re-replicating those requests.

Reasons why errors may be logged are:

- Replicator is unable to transfer one or more related items or requests for that request.
- A related design part or baseline for the request does not exist on the target site.
- A request has been either delegated to a different site, or requested by a different site, but it was subsequently worked on before the replication was run. The request is considered to be worked on if any items have been extracted against it at the time the replication is run.

Request Replication Example Scenarios

Overview of Replication Models

These examples are based on two conceptual working models for request replication—the delegation model and the request model.

The delegation model works on the assumption that requests are created on one site, and then ownership is delegated to another site to work on.

The request model follows the principle that if a user on any site sees a request that they want to work on, they can request that the ownership of that request is handed over to them.

The following sections build on these conceptual models and describe in more detail how they work.

Example Development Environment

The two example replication models are described with the following development scenario in mind. These examples involve issue replication within a multi-site development community sharing a common code base. Within this scenario, developers, testers, and designers are all working on many different sites and sharing code between them. For the sake of simplicity, this multi-site environment is broken down into the following:

- Site A is responsible for mainstream development. This includes design work, architecture changes, testing, and major project features.
- Site B is responsible for customer project development. This includes the development of customized solutions based on specific versions of the product set, designing and implementing new customer features, and, finally, testing these features.
- Site C is responsible for helping with mainstream development, product maintenance, and fixing priority customer issues. This includes the development and testing of such fixes, and then merging the code branches back into the latest official version of the code set.
- Site D is an external company that is responsible for portation work and integrating the product set with other third party software. They have limited access to the code base and are only allowed to make changes within specific mandates.
- Site E represents a customer who is closely involved in beta testing a product set. As such, they receive a copy of all requests raised against the product that they are testing with all "company sensitive" information removed. They also raise requests against the beta product that is then routed back to the appropriate development site.

Each site controls development currently through the use of Dimensions and uses Replicator, local project replication, and requests to control changes. An official release of a code set is represented by a baseline and any issues, changes, or customizations are controlled through the use of different types of requests. Replication of code sets is done both on a baseline basis and a project basis, and is performed at periodic intervals as necessary.

Each site has a similar design part component structure that is used to control those users who have responsibility for the various product components. Issues or enhancement requests are raised against the relevant design part and as a result are forwarded to the team lead responsible for further consideration.

Implementing a Major New Feature

This section discusses the model of delegating ownership for a request from one site to another. Ownership of a request means that the owning site can work on the request while all other sites can only view it.

This example describes what happens when a new "major" feature or set of features is required for the main product set. These features require numerous changes to a number of product components and are required for a new major release of the product. This work involves developers and testers on more than one site and is controlled through the specification of a work package that contains requests, each specifying a feature to be delivered. Requests within the work package are created on the main development site (Site A), and are targeted at developers on Site B and C. Each request is related to a specific design part and represents a specific feature that must be implemented. As part of this process, the request is delegated to a specific site that is then responsible for performing the work described in that document. After all such requests have been created and assigned, Replicator is requested to distribute the work package, and all its related components to the appropriate sites. As part of this replication, each request is copied across to the specified sites and placed in the pending list of the appropriate user based on the local part structure. By default, this access is read-only and shows for each request which site is responsible for implementing it. Only those sites that have been delegated the request can work on it.

After the initial replication is successfully completed, each request representing the piece of work to be done is delegated by the appropriate team lead to one or more developers. As part of implementing a feature, these developers relate item revisions to those requests and update the request attributes/action description as necessary. When a feature has been coded and unit-tested, the developer actions the request to the next stage. Each developer works at a different pace and implements features that may be necessary for other developers to get hold of. An example of this is where there is a critical path of features upon which other features are dependent. Because of this, periodic replications may be necessary to ensure that all the sites are in sync with any changes. These replications require items, requests, and any associated metadata changes (for example, item to request relationships) to be replicated to all the sites involved in the development process.

Summary of Delegation Model

The sites involved are A, B, and C.

The following tasks are undertaken:

- Site A creates a number of related requests that are allocated to the appropriate design parts. These requests are then delegated to particular sites for them to work on.
- The work package is then replicated to each site with the associated requests and relationships. Ownership is then assigned so that only those sites responsible for the requests can work on them.
- Developers then work on the code set, actioning requests when they have completed and relating the new item revisions as appropriate.
- At various stages, the work in progress is re-replicated to each of the sites. This updates each site with the latest items and changes.

In the example above, the delegation model is one where responsibility for a request is assigned by the site that currently owns it and is then transferred out to all the other sites involved in the replication process. As such, the ability to perform work on a request is delegated to the required site with all the other sites only being able to view the request contents. Following this model, there is one clear owner of a request at a time and conflicts do not occur from multiple sites performing updates in an uncontrolled fashion.

Fixing Issues in a Distributed Development Environment

This section discusses the example of a user being assigned a request as a result of a specific request.

This situation occurs when developers on Site A and C are working together on a major release that is currently undergoing both system and customer testing. Both sites have an up to date copy of the same code set and both sites have item replication active between them. Defect reports that are raised by testers on either site are replicated freely among them with the current owner of the defect being the site that raised it. In this environment, developers on either site are capable of fixing the defects that have been raised. If, while looking through their pending list, a developer on Site C spots a defect raised on Site A that they know how to fix, they request for the system to allocate that defect to them so that they can then fix it. This request is processed by Replicator so that ownership is transferred from the site that currently owns that request to the requesting site. The developer is thus able to work on fixing this defect.

Summary of Request Model

The sites involved are A and C.

The following tasks are undertaken:

- A defect report is created on Site A detailing a bug in one of the GUI utilities. This defect goes to the pending lists of the GUI developers on that site as normal.
- As part of the regular replication process, this request is replicated over to Site C and is placed in the pending lists of the GUI developers on that site.
- A developer on Site C recognizes the problem detailed in the request and requests that it be reassigned to them so that they can work on it.
- When the next replication takes place the ownership for that request is reassigned to the requesting site and the developer is then able to fix the issue.
- The fixed issue with the appropriate code is then re-replicated back to Site A when the next scheduled replication occurs.

As can be seen from the example above, the request model complements the delegation model in that any valid user from any site can request that a request be assigned to them without having to wait for the owning site to delegate it. This is useful because when there are a pool of developers who are able to work on issues, it is not up to the site that currently owns the request to find and allocate a resource to undertake a piece of work.

A request for a request, however, is allowed only if it is not already being worked on. Replicator determines that a request is being worked on if any items have been checked out against it at the time the replication is run. In this case, the request is not reassigned and the request is denied.

Chapter 25

Offline Replication Concepts and User Scenarios

Basic Concepts	342
Air-Gap Replication	343
Setting Up Air-Gap Replication	343
Performing an Air-Gap Replication Run	344
What Gets Created on the Operating System Disk?	346
Air-Gap Directory Structure Example	346
The Replication Metadata File (RMF)	347
Considerations	347

Basic Concepts

The main function of offline replication, or **air-gap replication**, is to facilitate parallel development of Dimensions CM requests, items, and baselines across different physically isolated geographical locations (as distinct from wired or wireless networked locations) where some or all of the sites have their own local Dimensions repositories for performance or operational reasons.

Example user scenario

In a security-conscious environment an internal network often does not have any connections to the outside world, so it is not possible to use online replication between a Dimensions CM repository on the local isolated network and one located at another geographical site. Air-gap replication fulfills this requirement.

The following scenario provides an example of air-gap replication:

> A particular military organization, for security reasons, does not have wired or wireless connected networks between sites. However, they want their multiple sites to work together on a work project using Dimensions. At regular intervals, they use air gap replication to generate a large compressed file hierarchy that they put on disk and securely physically deliver to remote sites.
>
> The remote sites then upload it, synchronize it locally, and send it back to the primary site when they've worked on it. When a replicated data set is uploaded, Replicator writes time stamps so that it knows what the gaps are and can identify changes.
>
> In the past, the customer could only achieve this sharing of offline data by transferring baselines out and transferring them back in. With air-gap replication, in-built automation ensures that the packages remain fully synchronized.

IMPORTANT!

- Before you start on this section, see Chapter 24, "Online Replication Concepts and Scenarios". Air-gap replication and online replication share many concepts that you need to understand before you work on this section.
- Replicator does not support the use of Dimensions CM streams.
- Replication of design parts is not supported. A separate data transfer utility, "PDIFF Import/Export", is available for transferring design parts between databases. Refer to Chapter 16, "Data Interchange File Format (PDIFF)" and Chapter 17, "Using PDIFF to Import/Export Data" for details.
- Replication of requirements information from one Dimensions CM server to another is not supported in Dimensions. However, by carefully coordinating Dimensions CM requests, you can still fully track the fulfillment of requirements in a replicated Dimensions environment. See, "Integrating Dimensions RM and CM in a Replicated Environment" on page 313.

Air-Gap Replication

Air-gap replication is essentially a three phase process:

- A Dumping Phase

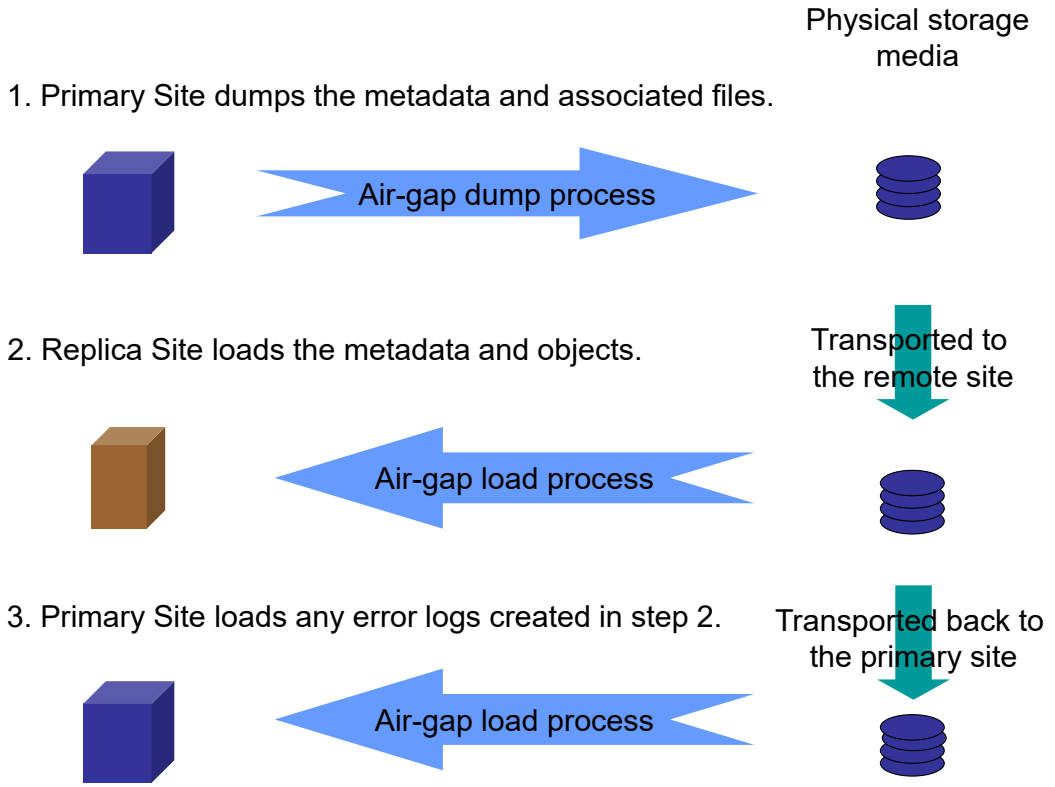
  Data is dumped from the primary site to a writable physical medium.

- A Loading Phase

  Data from the physical medium is loaded on the replica site.

- A Logging Phase

  Logs generated in the loading phase on the replica site are loaded back into the primary site.



Setting Up Air-Gap Replication

NOTE See also Chapter 26, "Preparing for Replication", Chapter 27, "Replication Administration", Chapter 28, "Replication Rules and Constraints", and Chapter 29, "Running Replicator".

Air-gap replication is defined as a "no network" configuration through clearing the default "online" check box on Replicator screens in the Administration Console. In all other respects the replication configuration is defined as normal on the primary site, namely:

1. Select the type of replication required:
   - Project (item)
   - Baseline
   - Request
2. Select the object types required for replication.
3. Select the desired branch names must also be selected (item and baseline replication).
4. Add the remote replica sites.

Performing an Air-Gap Replication Run

NOTE See also Chapter 29, "Running Replicator".

For air-gap replication, the following `replicator` standalone utility qualifiers are available:

`-dump <dir>`	Specifies a directory to which information is dumped. `-dump` and `-load` or `-load_logs` are mutually exclusive.
`-load <dir>`	Specifies a directory from which information is loaded. `-load` and `-dump` or `-load_logs` are mutually exclusive.
`-load_logs <dir>`	Specifies a directory from which logs are loaded. `-load_logs` and `-dump` or `-load` and are mutually exclusive.

IMPORTANT!

- The directories referred to above must be a either a full physical UNIX or a full physical Windows pathname (UNC pathnames are *not* supported).
- If a directory is not specified with one of above qualifiers, an error is displayed if you try to replicate to an "offline" subordinate site. Directory paths are required for these qualifiers.

Dumping on the Primary Site

The dumping process comprises the following steps:

1. Specify that you require an air-gap replicator dump on the primary site and nominating the associated directory on the removable medium.

   For example to dump:

   ```
   replicator –c ITEM_CFG_AG –dump "c:\temp\work"
   ```

2 On the primary site, ending up with a tree that always has a particular structure, see "What Gets Created on the Operating System Disk?" on page 346 and "Air-Gap Directory Structure Example" on page 346.

Loading on the Replica Site

The loading process comprises the following steps:

1 Take the removable medium with the dumped operating system tree contents from the primary site to the replica site.

2 Through the operating system on the replica site, extract the tree contents from the removable media onto the replica writable disk.

IMPORTANT! Make sure that all the extracted files attributes on the primary site writable disk are writable. Normally, they are.

3 Specify on the replica site that you require an air-gap replicator load and nominating the associated directory on the replica site writable disk.

For example to load:

```
replicator -c ITEM_CFG_AG -load "c:\temp\work"
```

4 Replicator inspects the tree (for example, the Replicator configurations must match). If the inspection is successful, it loads the Replicator configuration metadata into the replica site's Dimensions repository.

5 After loading the Replicator configuration, Replicator looks for the operating system directory to load from.

NOTE Because the path includes the replica site name, there is no possibility of loading other site info.

6 Replicator loads the Dimensions object metadata into the replica site repository and puts the item files into the appropriate replica site Dimensions item libraries.

7 Replicator writes a log file on the replica site.

Capturing the Replica Site Load Log Directory

Once the load has completed on the replica site and the log has been written, you need to capture the log so that you can return it to the primary site for synchronization.

1 Take the specified directory (the operating system tree and all its contents). This includes the log file just generated.

2 Transport it back to the primary site.

Remember that the operating system tree is always particular to the Replicator configuration and the replica site.

Loading the Replica Load Log Directory on the Primary Site

On return to the primary site:

1 Load the logs.

2 Put it onto a removable medium.

What Gets Created on the Operating System Disk?

Replicator creates a directory tree for the configuration being dumped:

1 The top level of this tree is a directory named after the configuration being dumped.

2 Under the top-level directory, there is a directory to indicate the type of objects being dumped by the configuration. This can be one of items, requests, or baselines.

3 Under that, a Replication Metadata File (RMF) file s created along with a directory for each replica in the configuration, containing the actual dump files and any log files for that replica. See "The Replication Metadata File (RMF)" on page 347. The directory name takes the following format:

```
<node_name>_<database>_<dsn>
```

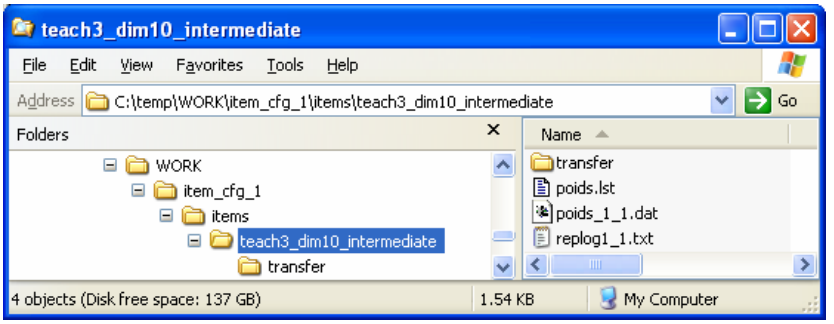
For example:

```
teach3_intermediate_dim10
```

Air-Gap Directory Structure Example

Consider an item replication configuration with a single replica definition for site `teach3:intermediate@dim10`. The following directory tree is created by:

```
replicator -c ITEM_CFG_1 -dump -dump_dir "c:\temp\work"
```



The Replication Metadata File (RMF)

A Replication Metadata File is a text file similar in format to Dimensions POIDS (pdiff dump files). See Chapter 16, "Data Interchange File Format (PDIFF)".

The RMF contains the configuration metadata required to create or validate the replication configuration on all of the replicas involved.

When replicating a baseline, the RMF also includes all the data required to create this baseline on the replica.

Considerations

Replication Logs

Following a `-dump` replication run on the primary site, the logs indicate that the objects were dumped successfully. Not until the objects have been loaded on the replica, the medium transported back to the primary, and the logs imported indicate a successful loading.

Data Integrity

You must not modify any of the files or directories created in a Replicator work area, specified by the `-load_dir` or `-dump_dir` parameters.

Baselines

These are special cases because they include items. The dump/load/log processes are, however, the same from the user point-of-view.

Chapter 26

Preparing for Replication

Defining and Controlling Remote Sites for Replication	350
Setting Up Replication Configurations	352
Setting Up Remote Users for Replication	353
Remote Password Security	353

Defining and Controlling Remote Sites for Replication

Overview

Before a replication configuration can be defined, base database sites that are to participate in replication must be registered as sites. This can be done by using the Administration Console, Dimensions networking commands, or the `dmpmcli` scripting interface.

The Administration Console, Dimensions networking commands, or the `dmpmcli` scripting interface enables you to specify the names of the nodes that comprise the Dimensions network. Included in this specification are the following (see earlier chapters in Part 4, "Network Administration" for details):

- Details on the operating system that is running on that node.
- The type of node from a Dimensions network aspect, for example, item library node or application server node.
- Additional information about the node, for example, the person responsible for the Dimensions databases on that node.
- Details on the Resident Software Definitions applicable (this is used to register information associated with the build environment in which Dimensions Make operates and is not relevant for Replicator).
- Details on Dimensions base databases resident on that node.

To view or modify site definitions you must have the `ADMIN_NETWORK` privilege. This privilege is validated whenever you invoke the Network Administration Tool.

The definition of a Dimensions replication site is made up of the following main components:

- Node Name

  The name of the network node that hosts the remote Dimensions base database.

- Database DSN

  The system identifier of the database Data Source Name in which the Dimensions base database resides.

- Dimensions Base Database Name

  The name of the base database involved in the replication process.

IMPORTANT!

- Replicator does not support the use of Dimensions streams.
- Replication of design parts is not currently supported. A separate data transfer utility, "PDIFF Import/Export", is available for transferring design parts between databases. Please refer to Chapter 16, "Data Interchange File Format (PDIFF)" and Chapter 17, "Using PDIFF to Import/Export Data" for details.
- Replication of requirements information from one Dimensions server to another is not currently supported in Dimensions. However, by carefully coordinating Dimensions requests, you can still fully track the fulfillment of requirements in a replicated Dimensions environment. See, "Integrating Dimensions RM and CM in a Replicated Environment" on page 313.

Defining a Physical Node Name and Database

The definition of a Dimensions node is undertaken through the use of the Administration Console, Dimensions networking commands, or the `dmpmcli` scripting interface.

NOTE Only physical nodes which have been registered are valid for use with replication.

For details about Network Node Registration, see Part 4, "Network Administration".

Registering Valid Databases for Replication Against a Node

Once a node has been registered for use against the Dimensions network, you are ready to register information about those base databases which reside on that node and which are to be involved in replication.

For details about registering databases against a node, see Part 4, "Network Administration".

Only when Dimensions base databases have been registered against a node, and have been enabled for replication, can replication configurations be defined that make use of them.

Setting Up Replication Configurations

Item, baseline, and request replication configurations are used to drive the replication process and define the context in which such replication is to occur. These configurations can be set up using either the Administration Console or the `dmpmcli` scripting interface as explained in the following subsections.

Setting Up Replication Configurations Using the Administration Console

The Administration Console can be used to:

- Administer the item, baseline, or request replication process.
- View what objects have been transferred from or received by a base database.

See Chapter 27, "Replication Administration" for details.

Setting Up Replication Configurations Using dmpmcli

As an alternative to using the Administration Console, the administration of the replication process can also be performed through the `dmpmcli` scripting interface.

For examples of how to perform such administration using `dmpmcli` scripts, see the file:

`%DM_ROOT%\AdminConsole\examples\replicationLibrary.js (Windows)`

`$DM_ROOT/AdminConsole/examples/replicationLibrary.js (UNIX)`

The *online help* explains the use of `dmpmcli`.

Such JavaScripts enable you to:

1. List existing Replication configurations.
2. Define item, baseline, or request replication configurations, which is essentially restricted for the use of users with the privilege of `ADMIN_REPL`, although a normal user may view what information has been set up.
3. Maintain the version branch names that have been currently defined in this installation.

Constraints

- Only users with the privilege of `ADMIN_REPL` can define or edit replication configurations. The ability to add or edit replication configurations only applies to those configurations that are *owned* by the resident primary base database—all *other* replication configurations are displayed for information purposes only.
- To be able to define or edit replication configurations, you must have previously specified whether or not replication is *allowed* for the Dimensions base database involved in replication.
- With respect to baseline replications:
  - The design part structure is not replicated from the primary to the replica site.

- Requests that are associated with the baselines are not replicated.
- Template definitions or process model objects, such as types, attributes, or lifecycle definitions, that the baseline might use are be replicated. Similar to item replication, intelligent defaulting is used where possible if such objects are missing.

- With respect to request replications:
  - The design part structure is not replicated from the primary to the replica site.
  - Baselines that are associated with the requests are not replicated.

Setting Up Remote Users for Replication

By default, Replicator connects to and runs processes on remote replicas as the operating system user that initiated the Replicator executable. This default behavior, however, can be modified by the definition of following environment variable, which enables a different operating system user to be set up

```
REPLICATORUSER_<NODENAME>_<BASEDB>_<SERVICE>     <username>
```

where

`<NODENAME>`	is the name of the replica node.
`<BASEDB>`	is the name of the replica base database.
`<SERVICE>`	is the name of the replica database service.

By defining this variable in the `dm.cfg` file, you can make Replicator run the replication process on the remote replica as the specific user you defined.

For example:

```
REPLICATORUSER_STALHITOMI_INTERMEDIATE_DIM10       repluser
```

Remote Password Security

To ensure that only certified users run it, Replicator performs an operating system authentication check against any user attempting to start Replicator to a remote replica node. You need to use the `dmpasswd` utility (see the *Command-Line Reference*) to register the user and password on the remote replica node. Otherwise, Replicator cannot connect that remote node.

Chapter 27

Replication Administration

About Replication Administration 356
Replication Administration Main Window 357
Managing Replication Configurations for Projects 361
Managing Replication Configurations for Baselines 364
Managing Replication Configurations for Requests 366
Managing Replication Logs 368

About Replication Administration

Purpose

Replication Administration enables you to:

- Define configurations for replicating items in projects and baselines, and for requests.
- Define replica sites to receive replications from primary database (DB) sites.
- View, edit, and delete existing replication configurations.
- View a list of sent or received transfers.
- View the details and result of a transferred or received replication.
- Delete log files for transferred and received replications.

Constraints

- Only users with the privilege ADMIN_REPL can define or edit replication configurations that are owned by current base database.
- To define or edit replication configurations, replication must be allowed for the base database. Use the Administration Console | Distributed Development | Network administration function to enable replication for a base database.
- For baseline replications, the design part structure, requests, template definitions, and other process model objects, such as types, attributes, or lifecycle definitions, that the baseline might use are not replicated.

IMPORTANT!

- Replicator does not support the use of Dimensions streams.
- Replication of design parts is not currently supported. A separate data transfer utility, "PDIFF Import/Export", is available for transferring design parts between databases.
- Replication of requirements information from one Dimensions server to another is not currently supported in Dimensions. However, by carefully coordinating Dimensions requests, you can still fully track the fulfillment of requirements in a replicated Dimensions environment.
- For details, see the Dimensions CM online help.

Invocation

Administration Console | Distributed Development | Replication administration

What is a Replication Configuration?

A replication configuration is the setup information which enables the primary DB site to control and scope the replication process. The manager of the primary DB site defines the replication configuration, which consists of a primary base database site and its named branches for replication, and replica base database sites and the projects in which replicated items are to appear.

What is a Replication Log?

A replication log is a report of the execution of a replication of an item or baseline configuration. It can be for:

- Project replications sent from this base database.
- Baseline replications sent from this base database.

- Request replications sent from this base database.
- Project replications received by this base database.
- Baseline replications received by this base database.
- Request replications received by this base database.

Each log consists of:

- A transfer record showing the basic properties of the replication.
- A list of objects that have transferred successfully.
- A list of objects that have failed to transfer.

Replication Administration Main Window

The Replication Administration main window consists of the following:

- **Menu area:** Displays the program's toolbar to help you carry out various tasks. See "Replication Administration Menu Area" on page 357.
- **Navigation area:** Enables you to view and select from a list of replication configurations or logs. See "Replication Administration Navigation Area" on page 358.
- **Content area:** Displays details about the selected replication configuration or log, or a summary of all configurations or logs if you selected the top-level icon in the navigation area. See "Replication Administration Content Area" on page 359.
- **Status area:** Displays log in details and the current product. See "Replication Administration Status Area" on page 361.

Replication Administration Menu Area

The Replication Administration menu area consists of two tabs and their associated toolbars:

Configurations Tab: Menu Area

The **Configurations** tab enables you to define replication configurations for projects and baselines.

Button	Function
	Add a new project, baseline, or request replication configuration using the New Replication Configuration dialog box.
	Delete the selected configurations.

Logs Tab: Menu Area

The **Logs** tab enables you to view the replications that have taken place to and from this base database.

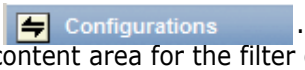
Button	Function
	Displays the Show Contents dialog box General tab.
	Displays the Show Logs dialog box.
	Deletes the selected replication logs.
	Purges all of the logs for the selected configuration except for the latest log.

Replication Administration Navigation Area

The navigation area displays a list of objects that varies depending on which tab you've selected.

Configurations: Navigation Area

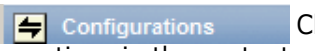
The navigation area lists the replication configurations that have been defined for the base database. It consists of:

- A filter to select which type of configuration to view. The options are:
  - **Project Primary:** Configurations where this base database is the primary site that replicates projects to replica sites.
  - **Project Replica:** Configurations where this base database is the recipient of project replications.
  - **Baseline Primary:** Configurations where this base database is the primary site that replicates baselines to replica sites.
  - **Baseline Replica:** Configurations where this base database is the recipient of baseline replications.
  - **Request Primary:** Configurations where this base database is the primary site that replicates requests to replica sites.
  - **Request Replica:** Configurations where this base database is the recipient of request replications.
- A top-level **Configurations** icon: 

. Clicking this icon displays a list of the configurations in the content area for the filter option selected above.
- A node for each Configuration ID: . Clicking one of these nodes displays the details of the configuration and its replica sites in the content area.

Logs: Navigation Area

The navigation area lists the logs for replications that have been sent or received for the base database. It consists of:

- A filter to select which logs to view. The options are:
  - **Projects Sent**.
  - **Projects Received**.
  - **Baselines Sent**.
  - **Baselines Received**.
  - **Requests Sent**.
  - **Requests Received**.
- A top-level **Configurations** icon: 

Clicking this icon shows the replication logs for all the configurations in the content area for the filter option selected above.
- A node for each Configuration ID: . Clicking one of these nodes shows only the logs for that configuration in the content area.

Replication Administration Content Area

The information that displays in the content area varies, depending on which tab is active and which object you selected in the navigation area.

Configurations: Content Area

When you select the *top-level icon* in the navigation area, the content area shows a list of the configurations for that selection. You can:

- Select a configuration to edit its details or delete it. Click to select or deselect all of the logs in the list.
- Print the list of configurations by clicking
- Save the list of configurations as comma-separated values by clicking
- Sort the configurations by clicking the column headings.

The content area contains a row for each configuration. The details include:

Field	Description
Check Box	A check box to select or deselect the replication configurations for the operations performed by the toolbar buttons.
Configuration ID	The name that identifies the replication configuration.
Site	The primary DB site that is responsible for sending the replication. This string consists of the node name, base database, and DB connection string.
Project Id (only present when **Project Primary** or **Project Replica** is selected)	The full specification for the project to be replicated by the primary site or the project to receive the replication by the replica site.
Replication Enabled	Indicates whether replication can take place between a primary site and replica sites.

When you select a *specific configuration* in the navigation area, you can:

- View and edit its details in the **General** section of the content area.
- View, add, edit, and delete remote replica sites in the **Remote Replicas** section.
- View, add, edit, and delete local replica sites in the **Local Replicas** section (Workset Primary and Workset Replica configurations only).

Logs: Content Area

When you select a node in the navigation area, the content area shows a list of the replications for that selection. You can:

- Filter the list according certain values. You do this by clicking the Filter link to display the Replication Logs Filter dialog box.
- Select a replication log to view its details or delete it. Click to select or deselect all of the logs in the list.
- Print the list of replication logs by clicking
- Save the list of replication logs as comma-separated values by clicking
- Sort the replication logs by clicking the column headings.

The content area contains a row for each replication log.

The details for a *sent* replication are:

Field	Description
Check Box	A check box to select or deselect the replication log(s) for the operations performed by the toolbar buttons.
Configuration ID	The ID of the configuration for which the replication was run.
Target Node	The network node to which the replication was sent.
Target Base Database	The base database to which the replication was sent.
Target Project/ Baseline	The project to which the items were transferred for an item replication or The baseline to which the items were transferred for a baseline replication or Not displayed for a request replication.
Sent Date	The date and time the replication was sent.

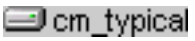
The details for a *received* replication are:

Field	Description
Check Box	A check box to select or deselect the replication log(s) for the operations performed by the toolbar buttons.
Configuration ID	The ID of the configuration for which the replication was run. Clicking the name link displays the Show Logs dialog box for the replication.

Field	Description
Source Node	The network node to which the replication was sent.
Source Base Database	The base database to which the replication was sent.
Source Project/ Baseline	The project to which the items were transferred for an item replication or The baseline to which the items were transferred for an baseline replication or Not displayed for a request replication.
Received Date	The date and time the replication was sent.

Replication Administration Status Area

The status area contains the following icons:

Icon	Description
DMSYS	The user ID of the user currently logged in.
Preference	Click this link to display the Set User Preferences dialog box.
	The name of the current product.
	The name of the base database currently accessed.
	Authenticate with a remote host or change a remote host password.

Managing Replication Configurations for Projects

Defining Replication Configurations for Projects

Purpose Follow this procedure to create a new project replication configuration for the current DB site. The replication configuration enables you to specify the project and branches to be replicated.

After defining the configuration, you can define the replica sites that are to receive the replicated project. When you run Replicator, it replicates the items that match the configuration to the replica sites.

To define a replication configuration for a project:

1 From the Replication Administration main window, click the **Configurations** tab.

2 From the filter in the navigation area, select **Project Primary**.

3 Click the New button. The **New Replication Configuration** dialog box appears with the **General** tab selected.

4 Enter a name for the configuration in the **Configuration ID** field, up to 25 characters.

5 Optionally, enter comments describing the configuration in the **Description** field, up 240 characters.

6 Select the project to replicate from the **Primary project** list. You can enter search text in the **Filter** field. The list is automatically filtered to reflect what you enter.

7 Leave **Is replication enabled to the replicas** selected.

8 Optionally, click the **Branches** tab and select all of the named version branches that you want to replicate.

Two special options include:

- **<ALL_LOCAL_BRANCHES>:** Replicates all named branches owned by the primary DB site to the replica DB sites.
- **<ALL_NAMED_BRANCHES>:** Replicates all named branches on the primary DB site to the replica DB sites except those that are owned by the replica DB sites.

NOTE These options are not supported for local replication. You must explicitly select the branches to be replicated to local replica sites.

9 Click **OK** to create the configuration.

Defining Replica Sites for Project Replication

Purpose Once you have defined a project replication configuration, follow this procedure to specify the replica projects into which the primary site replicates items. You can also specify whether the replica site can replicate branches back to the primary site.

A replica site can be:

- **Local:** On the same machine that hosts the primary DB site.
- **Remote:** On a different machine on the Dimensions network that does not host the primary DB site.

To define a replica site for a project replication:

1 From the Replication Administration main window, click the **Configurations** tab.

2 From the filter in the navigation area, select **Project Primary**.

3 Select the configuration to which you want to add the replica site from the Configurations list.

4 In the content area, click the Add button in the Remote Replicas section or the Local Replicas section.

5 For a remote replica site (if you have not already defined the appropriate remote replica network parameters *on the primary site*, you need to use the Administration Console **Network administration** cluster tools before proceeding. See "Network Applications" on page 275):

a From the **Node name** list, select the remote network node.

b From the **DB connection** list, select the string that you want to use to connect to the database server.

c From the **Base database** list, select the base database name.

d In the **Project** field, enter the full specification of the project to receive the replicated items.

e The **Is replication enable for this replica** check box should normally be left checked.

f If you are using offline/"air-gap" replication, clear the **Is this online Replication** option.

g To allow replication from the replica site back to the primary site, select the **Is replica replicating branches back to the primary** check box. Initially, you may find it advantageous not to select this check box until you have established and tested replication from the primary to the replica site.

h (Optional) Click the **Branches** tab and enter the named branches that you want to replicate back to the primary site. If you selected the **Is replica replicating branches back to the primary** option, you should select the requisite named branches on this tab, otherwise all named branches are selected by default.

i Click **OK**.

6 For a local replica site:

a From the **Project** list, select the local project to receive the replicated items from the primary site. You can enter search text in the **Filter** field. The list is automatically filtered to reflect what you enter.

b To allow replication from the replica site back to the primary site, select the **Is replica replicating branches back to the primary** check box. Initially, you may find it advantageous not to select this option until you have established and tested replication from the primary to the replica site.

c Optionally, click the **Branches** tab and enter the named branches that you want to replicate back to the primary site. If you selected the **Is replica replicating branches back to the primary** option, you should select the requisite named branches on this tab, otherwise all named branches are selected by default.

d Click **OK**.

e Restart the Dimensions Listener service.

Modifying Project Replication Configurations

Purpose Follow this procedure when you need to change the details of a project replication configuration or its replica sites.

You can also delete configurations or replica sites as necessary.

To edit or delete a project replication configuration:

1 From the Replication Administration main window, click the **Configurations** tab.

2 From the filter in the navigation area, select **Project Primary**.

3 Select the configuration that you want to edit or delete.

4 Do one of the following:

- Click the Edit button on the General section and change the configuration details as necessary, see "Defining Replication Configurations for Projects" on page 361.
- Click the Delete button on the toolbar and confirm that you want to delete the configuration.

To edit or delete a replica site:

1. From the Replication Administration main window, click the **Configurations** tab.
2. From the filter in the navigation area, select **Project Primary**.
3. Select the configuration that contains the replica site that you want to edit or delete.
4. In the content area, select the replica site in the Remote Replicas section or the Local Replicas section.
5. Do one of the following:
   - Click the Edit button and change the replica site's details as necessary. See "Defining Replica Sites for Project Replication" on page 362.
   - Click the Delete button and confirm that you want to delete the replica site.

Managing Replication Configurations for Baselines

Defining Replication Configurations for Baselines

Purpose Follow this procedure to create a new baseline replication configuration for the current DB site. The replication configuration enables you to filter the items in a baseline to be replicated.

Replicator replicates the items that match the configuration to the replica sites and then creates a baseline using those replicated items.

To define a replication configuration for a baseline:

1. From the Replication Administration main window, click the Configurations tab.
2. From the filter in the navigation area, select **Baseline Primary**.
3. Click the New button. The New Replication Configuration dialog box opens, with the **General** tab selected.
4. Enter a name for the configuration in the **Configuration ID** field, up to 25 characters.
5. (Optional) Enter comments describing the configuration in the **Description** field, up 240 characters.
6. Leave **Is replication enabled to the replicas** checked.
7. (Optional) Click the **Branches** tab and select all of the named version branches that you want to replicate.

   Two special options include:

   - **<ALL_LOCAL_BRANCHES>:** Replicates all named branches owned by the primary DB site to the replica DB sites.

- **<ALL_NAMED_BRANCHES>:** Replicates all named branches on the primary DB site to the replica DB sites except those that are owned by the replica DB sites.

8 Click the **Item Types** tab and select all of the item types that you want to replicate.

9 Click **OK** to create the configuration.

Defining Remote Replicas for Baselines

Purpose Once you have defined a baseline replication configuration, follow this procedure to specify the replica site into which the primary site replicates the baseline.

To define a replica site for a baseline replication:

1 From the Replication Administration main window, click the **Configurations** tab.

2 From the filter in the navigation area, select **Baseline Primary**.

3 Select the configuration to which you want to add the replica site from the **Configurations** list.

4 In the content area, click the Add button from the Remote Replicas section. The New Remote Replica dialog box appears (if you have not already defined the appropriate remote replica network parameters *on the primary site*, you need to use the Administration Console **Network administration** cluster tools before proceeding. See "Network Applications" on page 275).

5 From the **Node name** list, select the remote node.

6 From the **DB Connection** list, select the string that you want to use to connect to the database server.

7 From the **Base Database** list, select the base database name.

8 The **Is replication enabled for this replica** option should normally be left checked.

9 If you are using offline/"air-gap" replication, clear the **Is this online Replication** option.

10 Click **OK**.

Modifying Baseline Replication Configurations

Purpose Follow this procedure when you need to change the details of a baseline replication configuration or its replica sites.

You can also delete configurations or replica sites as necessary.

To edit or delete a baseline replication configuration:

1 From the Replication Administration main window, click the **Configurations** tab.

2 From the filter in the navigation area, select **Baseline Primary**.

3 In the navigation area, select the configuration that you want to edit or delete.

4 Do one of the following:

- Click the Edit button on the General section and change the configuration details as necessary. See "Defining Replication Configurations for Baselines" on page 364.
- Click the Delete button on the toolbar and confirm that you want to delete the configuration.

To edit or delete a replica site:

1. From the Replication Administration main window, click the **Configurations** tab.
2. From the filter in the navigation area, select **Baseline Primary**.
3. In the navigation area, select the configuration that contains the replica site that you want to edit or delete.
4. In the content area, select the replica site from the Remote Replicas section.
5. Do one of the following:
   - Click the Edit button and change the replica site's details as necessary. See "Defining Remote Replicas for Baselines" on page 365.
   - Click the Delete button and confirm that you want to delete the replica site.

Managing Replication Configurations for Requests

Defining Replication Configurations for Requests

Purpose

Follow this procedure to create a new request replication configuration for the current DB site. The replication configuration enables you to specify the request types to be replicated.

After defining the configuration, you can define the replica sites that are to receive the replicated requests. When you run Replicator, it replicates the requests that match the configuration to the replica sites.

Replication sites for request replication need to have a unique site number to avoid conflicts between request numbers from different sites. Dimensions uses the site number to generate unique request IDs between the sites. The site number is defined in the `Administration Console | Network administration` function. See "Managing Base Databases" on page 282 for details. All the sites involved in the replication need to be defined in the base database of each of the sites involved, ensuring that the allocated site numbers are consistent.

CAUTION! If you have request replication running between a number of sites and you want to add a new site, you must perform a replication to synchronize all the databases before creating any requests on that new site. You need to do this immediately after you have updated all the base database site numbers currently on the affected nodes. Failure to do so may lead to conflicting request IDs.

To define a replication configuration for requests:

1. From the Replication Administration main window, click the **Configurations** tab.
2. From the filter in the navigation area, select **Request Primary**. The New Replication Configuration dialog box opens, with the **General** tab selected.

3 Click the New button.

4 Enter a name for the configuration in the **Configuration ID** field, up to 25 characters.

5 (Optional) Enter comments describing the configuration in the **Description** field, up 240 characters.

6 Leave **Is replication enabled to the replicas** selected.

7 (Optional) Click the **Request Types** tab and select all of the request types that you want to be included in the replication.

8 Click **OK** to create the configuration.

Defining Remote Replicas for Request Replications

Purpose Once you have defined a request replication configuration, follow this procedure to specify a replica site into which the primary site replicates the requests.

To define a replica site for a request replication:

1 From the Replication Administration main window, click the **Configurations** tab.

2 From the filter in the navigation area, select **Request Primary**.

3 In the navigation area, select the configuration to which you want to add the replica site in the navigation area.

4 In the content area, click the Add button from the Remote Replicas section.

The New Remote Replica dialog box opens (if you have not already defined the appropriate remote replica network parameters *on the primary site*, you need to use the Administration Console **Network administration** cluster tools before proceeding. See "Network Applications" on page 275).

5 From the **Node name** list, select the remote node.

6 From the **DB connection** list, select the connection string that you want to use to connect to the database server.

7 From the **Base database** list, select the base database name.

8 The **Is replication enabled for this replica** option should normally be left selected.

9 If you are using offline/"air-gap" replication, uncheck the **Is this online Replication** option.

10 To allow replication from the replica site back to the primary site, select the **Is replica replicating back to the primary** option.

11 To restrict the access to the replicated requests to read-only, select the **Replicate requests to this site as read-only** option**.**

12 (Optional) Click the **Requests Types** tab and enter the request types that you want to replicate back to the primary site.

13 Click **OK**.

Modifying Request Replication Configurations

Purpose Follow this procedure when you need to change the details of a request replication configuration or its replica sites.

You can also delete configurations or replica sites as necessary.

To edit or delete a request replication configuration:

1. From the Replication Administration main window, click the **Configurations** tab.
2. From the filter in the navigation area, select **Request Primary**.
3. In the navigation area, select the configuration that you want to edit or delete.
4. Do one of the following:
   - Click the Edit button on the General section and change the configuration details as necessary. See "Defining Replication Configurations for Requests" on page 366.
   - Click the Delete button on the toolbar and confirm that you want to delete the configuration.

To edit or delete a replica site:

1. From the Replication Administration main window, click the **Configurations** tab.
2. From the filter in the navigation area, select **Request Primary**.
3. In the navigation area, select the configuration that contains the replica site that you want to edit or delete.
4. In the content area, select the replica site from the Remote Replicas section.
5. Do one of the following:
   - Click the Edit button and change the replica site's details as necessary. See "Defining Remote Replicas for Request Replications" on page 367.
   - Click the Delete button and confirm that you want to delete the replica site.

Managing Replication Logs

Listing and Viewing Replication Logs

Purpose Follow this procedure to list and view the replication logs in the base database.

To list replication logs:

1. In the Replication Administration main window, click the **Logs** tab.
2. Choose which type of logs you want to view from the filter in the Navigation Area:
   - **Projects Sent**
   - **Projects Received**
   - **Baselines Sent**

- **Baselines Received**
- **Requests Sent**
- **Requests Received**

3 Do one of the following:

If you want to...	Then...
List all logs for the base database	Click **Configurations** at the top of the navigation tree.
List the logs for a specific Configuration ID	Select **Configuration ID** in the navigation area.

To filter the list of replication logs:

1 Click the Filter Link at the top of the content area.

2 Complete the fields in the Replication Logs Filter dialog box. To clear the filter criteria, click the **Reset Filter** button.

To view the replication logs:

1 Select the appropriate replication log in the content area.

2 Do one of the following:

If you want to...	Then...
View the details of a replication	Click **Show Contents** icon. The Show Contents dialog box (General tab) is displayed.
View the contents of a replication log	Click **Show Log**. The Show Logs dialog box is displayed.

To delete replication logs:

1 Select the replications to delete in the content area.

2 Do one of the following:

- Click the Delete button to delete the selected logs.
- Click the Delete All button to purge all logs for the selected configuration, except for the latest transfer.

3 Click **Yes** to confirm that you want to delete the logs.

Chapter 28

Replication Rules and Constraints

Introduction	372
General Item-Specific Restrictions	373
General Baseline-Specific Restrictions	374
General Request-Specific Restrictions	374
Minimum Process Model Compliance Requirements	375
Metadata Replication Rules	375
Integrity Constraints on Replicated and Transferred Version Branches	377
Data Integrity and Extending Replication Configurations	378

Introduction

To replicate objects successfully between base databases, and to preserve the integrity of the replicated information on the sender and recipient databases, certain rules and constraints are enforced by Dimensions Replicator. Similarly, for baselines to be replicated successfully from one site to another, baseline-specific rules and constraints are enforced.

These rules and constraints are divided into the following main categories.

- **General item-specific restrictions** such as what kind of items can be replicated and what the security policy for replication is.
- **General baseline-specific restrictions** such as what kind of baselines can be replicated and what the security policy for replication is.
- **General request restrictions** such as under what conditions requests can be replicated, and what the security policy for replication is.
- **General constraints** with respect to the minimum level of compliance required between the process model on the primary DB site and the process models on the replica DB sites. These must be satisfied before replication is possible from a primary to a replica or vice versa.
- **Metadata replication rules** defining how the various system-defined and user-defined attributes and relationships of an item revision on the sender are to be mapped on the recipient, for example, what is the rule used when the lifecycle state of the item revision does not exist in the lifecycle used on the replica.
- **Integrity constraints** preserving the integrity of site ownership of version branches, for example, users should only be able to work on their own local branches and must not be allowed to modify the item revision of a branch owned by another DB site.

Each of the above categories is considered separately in the following sections.

IMPORTANT!

- Replicator does not support the use of Dimensions CM streams.
- Replication of design parts is not supported. A separate data transfer utility, "PDIFF Import/Export", is available for transferring design parts between databases. Refer to Chapter 16, "Data Interchange File Format (PDIFF)" and Chapter 17, "Using PDIFF to Import/Export Data" for details.
- Replication of requirements information from one Dimensions CM server to another is not supported in Dimensions. However, by carefully coordinating Dimensions CM requests, you can still fully track the fulfillment of requirements in a replicated Dimensions environment. See, "Integrating Dimensions RM and CM in a Replicated Environment" on page 313.

General Item-Specific Restrictions

The following limitations apply to item replication:

- Replication is supported only for item revisions on named branches. Item revisions on "nameless" branches cannot be replicated.

  For example, `ACME30:FINAL_RELEASE.UNIX-C;4` is an item revision on the nameless branch whereas `ACME30:FINAL_RELEASE.UNIX-C;main#4` is an item revision on branch `main`.

- To replicate item revisions belonging to a "nameless" branch, you must first give the branch a name using the Label Branch Utility described in Appendix D, "Replicator Utilities".

- While derived items (for example, object modules and executables) that have been created through the Dimensions Make tools can (optionally) be replicated, the 'made-of' metadata that describes the 'derived source to object' relationships is not transferred. This means that on the replica sites, Dimensions Make regards these replicated derived items as uncontrolled files.

- Checked-out item revisions (those at state `$TO_BE_DEFINED`) are not replicated.

- Item revisions that have been placed offline and archived using the Dimensions Archive & Retrieval Tool (Dimensions ART) are not replicated.

- Directory items are not replicated.

- To replicate from a primary DB site to a replica DB site, you must have `ADMIN_REPL` privilege for the primary DB site. Furthermore, every machine hosting a replica DB site must already be running the Replicator Listener. It is imperative that the user owning this listener has `ADMIN_REPL` privilege for the replica DB (or DBs), otherwise the primary is not able to establish a connection to the replica. Note that the user with `ADMIN_REPL` privilege for a primary DB site and the replica DB sites may be different users.

- When items are dumped as part of item replication, all relationships between the item and change requests, design parts and other items are also replicated. Also, the related items (and any required placeholder items) are dumped as objects—this does not apply to the related design parts or change requests. The relationships of any placeholder items dumped are not dumped.

- All usage relationships to design parts are replicated.

- All relationships to items are replicated when the items are on a version branch that matched the replication policy.

- Only the `Affected` item to change request relationships are replicated. `InResponsTo` and `Info` relationships are ignored as part of the item replication. These relationships are still replicated as part of change request replication. Furthermore, only relationships to change requests in the primary catalogue are replicated, relationships to request in the secondary catalogue are ignored.

General Baseline-Specific Restrictions

The following limitations apply to baseline replication:

- The design part structure is not replicated from the primary site to the replica.
- Requests associated with baselines are not replicated.
- Template definitions or process model objects, such as types, attributes or lifecycle definitions, that the baseline might use are not replicated. Similar to item replication, intelligent defaulting is used where possible if such objects are missing.
- Baseline replication based on metadata updates is not supported. Altering a baseline's attributes does not cause it to be replicated.

  When items are dumped as part of baseline replication all relationships between the item and change requests, design parts and other items are also replicated. Also, the related items (and any required placeholder items) are dumped as objects—this does not apply to the related design parts or change requests. The relationships of any placeholder items dumped are not dumped.
- For items replicated as part of baseline replication, the items are replicated with all relationships to:
  - Design parts
  - Change requests
  - Items

  If these relationships are not required, or are not applicable to the receiving system, you are advised to do one of the following:
  - When dumping the baseline, specify the `-dont_replicate_item_item_rels`, `-dont_replicate_item_request_rels`, `-dont_replicate_item_part_rels` options (or the single option `-dont_replicate_item_all_rels`).
  - When loading the baseline specify the `-ignore_related_errs` option.

General Request-Specific Restrictions

The following limitations apply to request replication:

- The related design part structure is not replicated from the primary site to the replica.
- Baselines related to requests are not replicated.
- External requests are not replicated.
- Template definitions or process model objects, such as types, attribute, or lifecycle definitions, that the request might use are not replicated. Similar to item replication, intelligent defaulting is used where possible if such objects are missing.
- The following new replicator option is enforced, this may not be altered:

  ```
  -dont_replicate_item_request_rels
  ```

  This ensures that requests related to items performing item replication are not replicated.

Minimum Process Model Compliance Requirements

Although in general the process model on the primary DB site and those on the replica DB sites may be different, the following minimum set of requirements must be met before replication can take place:

- Generic

Dimensions Product Name	The name of the product owning an item revision or request that is to be replicated must also exist in the recipient DB site.
Defined Object Types and their Lifecycles	The type of an item revision or request that is to be replicated (for example, `SRC` and `OBJ`) must exist and be assigned a lifecycle on the recipient DB site before the item or request can be replicated.
Item Library for Specified Item Type or Types	For each matching item type on a recipient base database, there must exist on that base database an item library into which such items (of that type) can be placed. This may be created explicitly for each type or a default library defined for all types.
Relationship Types	The relationship type must exist on the receiving site. Otherwise, the replication fails.

- Baseline Specific

Dimensions Design Parts	For each item in a baseline, the design part owning that item must exist on the replica. Otherwise, that item fails to be replicated.

- Request Specific

Dimensions Design Parts	For each request, the related design part must exist on the replica Otherwise, that request fails to be replicated.

Metadata Replication Rules

The following mapping rules are used when replicating the metadata of an item revision or request:

Action History	The full action history is replicated to the recipient.
Baseline Item Relationships and Which Baselines Item Is In	Baseline item relationships and "which baselines item is in" are not replicated.
Baseline Request Relationships	Baselines related to replicated requests are not replicated.
Built Item History	Built item history (for example, `"I am OBJ"` and `"I came from SRC revision 3 and went into an EXE revision 5"`) is not replicated.

Built Item,Dimensions Make information and Made-of List Relationships	Built item, Dimensions Make information and Made-of List relationships are not replicated.
Request to Item Relationships	Request to item relationships for replicated items are not replicated.
Change Management Rules	Change management rules for item creation in the recipient are bypassed.
File Attachments, Action Descriptions, and Detailed Descriptions	File attachments, action descriptions, and detailed descriptions for requests are also replicated.
Design Part to Item Usage Relationships and Owner Design Part Rule	Design part to item usage relationships are not replicated. Ownership relationship is, however, replicated. If the owner design part does not exist on the recipient, then optionally the recipient's top product design part is used to own the item, or by default the item fails to replicate with an appropriate error message.
Checked out and Offline Items	Items that are checked out or offline at the time replication occurs are not replicated.
Requests with checked out items	Requests that are related to checked out items are not replicated.
Item Formats	Formats of replicated items are validated against those formats defined on the recipient base database. For example, if an item is assigned the format `LIB (BINARY)` on the primary DB site, but has been defined on the recipient database as `LIB` (text), the item is rejected.
Item to Item Relationships	Item to item relationships are not replicated.
Item User Defined Report Relationships	Item user defined report relationships are not replicated.
Library Filename	Is retained if no conflict occurs. Otherwise, a new library filename is generated automatically.
Lifecycle State	If the lifecycle state of the item on the sender is not defined on the recipient's lifecycle, the initial lifecycle state is used.
Revision Pedigree	Only revision pedigree information relevant to the replicated item revision is replicated (see "Item Revision 'Holes'" on page 330 for details).
Role Assignment and Inbox	Role assignments on an item or request are not replicated. The replicated item or request inherits the role assignment of the recipient base database. The item or request is displayed in the user inboxes according to its lifecycle state and the user role assignment on the recipient.
Update History	Update history for the item revision or request is replicated.
User-Defined Attributes and Attribute Update Rules	Only those user-defined attributes that are common to the sender and the recipient are replicated. Attribute update rules for item creation in the recipient are bypassed.

Project Filename	Is retained on the recipient. If a project filename with the same name exists, this is treated as an error.
Project Filename Changes	It is possible that the sender changes the project filename of an item that has already been replicated. Replicator propagates such a change to each recipient.
Project Relationships	Any relationships the replicated item has to projects other than that into which it is being replicated are ignored.

Integrity Constraints on Replicated and Transferred Version Branches

When a named branch is replicated from one base database site to another, the recipient base database registers the branch as owned by the sender base database and flags the branch as LOCKED so that its users cannot modify the branch. More specifically, the recipient enforces the following constraints on the remotely owned branch.

- The following Dimensions functions are not permitted on an item revision belonging to a remote branch:
  - Actioning the item to a new state.
  - Editing the item's attributes.
  - Deleting the item revision.
  - Changing the project filename.
  - Editing the item revision at the initial lifecycle state.
  - Creating a new revision of the item on that branch by checking it out.
- The get/print/browse functions are supported for any item revision on the branch.
- Checking out a new revision from a revision already on the remote branch is permitted, provided that the new revision is on a local branch.

When an item revision is replicated from a base database site to another base database site, the sender DB site imposes the following constraints and rules with respect to the Dimensions functions that may be performed against the replicated revision.

- The following functions are not permitted on the item revision:
  - Deleting the item revision.
  - Editing the item revision at initial lifecycle state.

All other Dimensions functions are allowed where applicable.

Data Integrity and Extending Replication Configurations

When request replication is established and in use, you may need to add additional replica databases. It is important that these nodes are replicated to before new requests are entered into their system by users. This ensures all nodes in the configuration are synchronized prior to the creation of any new requests.

Chapter 29

Running Replicator

Overview	380
Launching Replicator from the Command Line	380

Overview

Once Replication Configurations have been defined as discussed in Chapter 26, "Preparing for Replication", you can begin to initiate the replication process for a particular replication configuration. The replication process is initiated from the command line using the standalone `replicator` command.

IMPORTANT!

- Replicator does not support the use of Dimensions streams.
- Replication of design parts is not currently supported. A separate data transfer utility, "PDIFF Import/Export", is available for transferring design parts between databases. Please refer to Chapter 16, "Data Interchange File Format (PDIFF)" and Chapter 17, "Using PDIFF to Import/Export Data" for details.
- Replication of requirements information from one Dimensions server to another is not currently supported in Dimensions. However, by carefully coordinating Dimensions requests, you can still fully track the fulfillment of requirements in a replicated Dimensions environment. See, "Integrating Dimensions RM and CM in a Replicated Environment" on page 313.

Launching Replicator from the Command Line

The replication process is initiated by issuing the `replicator` command at the operating system command prompt using the syntax as described below. This can be a manual process or automated through a scheduler, for example, UNIX `cron`.

NOTES

- You must have `ADMIN_REPL` privilege for the base database in order to execute `replicator`.
- Unless you use the `replicator -p <database>` parameter, described below, you must ensure that the `DMDB` symbol is set, as described for `pdiff`, see "General Description" on page 236.
- The Dimensions Replicator command `replicator` utilizes "local connection", that is, it expects the network service name to be the default `pcms_sdp` — it does not function if you set an alternative network service name as described in Appendix C, "Using a Non-Default Service Name and Port Number". Additionally, all sites involved in replication must use the same port number for Dimensions CM.

Replicator Syntax

```
replicator -c <replication_id>

   [ -a ] [ ]-allow-overwrite-latest] [ -b ]
   [ -baseline_id< baseline_id> ] [ -c <replication_id> ][ -d ]
   [ -defer_connect] [ -dont_check_branches ] [ -dont_query_logs ]
   [ -dont_match_part]
   [ -dont_replicate_blns ] [ -dont_replicate_chdocs ]
   [ -dont_replicate_item_all_rels] [ -dont_replicate_item_item_rels ]
   [ -dont_replicate_item_part_rels]
   [ -dont_replicate_item_request_rels(1) ] [ -dont_replicate_items ]
   [ -dont_rereplicate_to_sender ] [ -dont_replicate_ws ]
   [ -dont_update_baseline ] [ -dont_update_chdoc ]
   [ -dont_update_items ] [ -dont_validate_user ] [ -dump <directory> ]
   [ -e ] [ -f ] [ -file <filename> ] [-force_baseline]
   [ -h <host_name> ] [ -help ] [ -ignore_related_baseline ]
   [ -ignore_related_errs ] [ -l ][ -load <directory> ]
   [ -load_logs <directory> ] [ -m number ] [ -n number ]
   [ -no_bln_branch_check ] [ -no_branch_check ] [ -no_replicate_mwsd ]
   [ -o <filename> ] [ -p <database> ] [ -prune_logs ]
   [ -prune_old_logs <date> ] [ -q ] [ -r ][ -redo ]
   [ -replicate_offnorm ] [ -replicate_remote_branches ][ -s ] [ -sc ]
   [ -t <item types> ] [ -u ] [ -use_old_attr_format ] [ -x ]
```

(1) This is a valid option for item or baseline replication, but is not a valid non-option for requests replication—where it is automatically enforced.

- **-c <replication_id>**

  (Mandatory) The Replication_Id of the configuration to be used for the replication process.

- **-a**

  Files are to be transferred to and from replica DB sites. This is the default case.

  See, also, the `-r` and `-s` flags.

- **-allow-overwrite-latest**

  When replicating items back to the their owning site, allow the incoming items to overwrite any local updates. (This option is required to disable the default replicator behavior of not overwriting an object when the update date of the current object is greater than that of the incoming POID data.)

- **-b**

  Items built/derived from Dimensions Make and Dimensions Build are to be considered for replication (this does not include build dependency information). By default derived items are ignored.

- **-baseline_id <baseline_id>**

  The Baseline_Id of the configuration to be replicated.

- **-d**

  Disable integrity constraint checks on branch ownership on the primary and instead defer them to the replica. The default is for the ownership of the branches to be checked on the primary site. This option reduces the initial startup time required by `replicator`.

- -**dont_match_part**

  Do not change an item's owning part on the receiving site.

  Default: change the part if necessary to match that of the previous revisions.

- **-defer_connect**

  Defer connection to the replica site until replication candidates have been found.

IMPORTANT! This option does not defer connection for back replication.

- **-dont_check_branches**

  Do not check the ownership of a version branch. The default behavior is to only replicate branches owned by the site.

  This option disables the default behavior of preventing the back replication of version branches that are not owned by the replica site. It disables part of the configuration validation process which takes place before any objects are transferred.

NOTE This option is not valid for issue replication.

- **-dont_query_logs**

  Do not use previous logs to identify replication candidates. This improves performance, but may result in necessary repeat replication.

- **-dont_replicate_blns**

  When replicating requests, do not replicate any of their related baselines.

- **-dont_replicate_chdocs**

  When replicating requests, do not replicate any of the related requests.

- **-dont_replicate_item_all_rels**

  Do not replicate relationships to items, requests, and design parts performing project (item) replication.

- **-dont_replicate_item_item_rels**

  Do not replicate items related to items when performing project (item) replication.

- **-dont_replicate_item_part_rels**

  Do not replicate relationships to design parts performing project (item) replication.

- **-dont_replicate_item_request_rels**

  Do not replicate requests related to items performing project replication.

IMPORTANT! This is a valid option for item or baseline replication, but is not a valid non-option for requests replication—where it is automatically enforced.

- **-dont_replicate_items**

  When replicating requests, do not replicate any of the related items.

- **-dont_rereplicate_to_sender**

  Do not resend objects already replicated.

- **-dont_replicate_ws**

  When replicating requests, do not replicate project to request relationships.

- **-dont_update_baseline**

  Disable the re-replication of request metadata. This is analogous to the `-dont_update_items` qualifier for item replication.

NOTE This option is only valid for Dimensions CM 10 or later.

- **-dont_update_chdoc**

  Disable the re-replication of request metadata. This is analogous to the `-dont_update_items` qualifier for item replication.

- **-dont_update_items**

  Disable the re-replication of item metadata, as described in "Replication of Updates to Item Metadata" on page 330.

- **-dont_validate_user**

  Do not perform any validation of the operating system user-ids when requests are loaded into the database. This is to support the concept of guest or change management users.

- **-dump <directory>**

  When using "air-gap" replication, dump Dimensions objects to an offline site using the specified directory. `-dump` and `-load` or `-load_logs` are mutually exclusive.

IMPORTANT!

- The directory referred to above must be a either a full physical UNIX or a full physical Windows pathname (UNC pathnames are *not* supported).
- If a directory is not specified, an error is displayed if you try to replicate to an "offline" subordinate site. Directory paths are required for this qualifier.

- **-e**

  Specifies that the removal of items from the primary project (through RIWS) is not propagated to any replicas. The default is for this propagation to occur.

  In addition to the replication of new objects and the replication of metadata updates, the default behavior of Replicator includes removing item revisions from remote worksets if the same item revisions have been removed from the corresponding workset of the sending site. The `-e` option disables this behavior.

- **-f**

  Force items to be loaded where the owning design part does not exist on the replica (the default is to stop an item being replicated if a part is not present). This option is only applicable for item replication.

- **-file <filename>**

  The name of a file that contains a comma-separated list of request IDs to be replicated instead of replicating the requests specified by the replication configuration.

- **-force_baseline**

  By default, if a baseline fails to replicate due to problems encountered during the replication process, Dimensions Replicator enforces that the baseline concerned is replicated once the issues causing the failure have been resolved. Attempts to replicate a different baseline following a failed baseline replication generates the following error message:

  ```
  The replica is still expecting items from baseline X
  ```

  To remove this restriction in the above situation, and permit the replication of a different baseline, specify the `-force_baseline` flag.

- **-help**

  Output a concise version of the information documented here.

- **-ignore_related_baseline**

  When replicating baselines, do not check whether predecessors to merged baselines have already been replicated.

- **-ignore_related_errs**

  When replicating requests, ignore any error due to related objects not existing on the receiving database.

- **-l**

  Run in verbose mode. This is essentially a debug option.

- **-load <directory>**

  When using "air-gap" replication, load Dimensions objects from an offline site using the specified directory. `-load` and `-dump` or `-load_logs` are mutually exclusive.

IMPORTANT!

- The directory referred to above must be a either a full physical UNIX or a full physical Windows pathname (UNC pathnames are *not* supported).
- If a directory is not specified, an error is displayed if you try to replicate from an "offline" subordinate site. Directory paths are required for this qualifier.

- **-load_logs <directory>**

  When using "air-gap" replication, load Replicator logs from an offline site using the specified directory. `-load_logs` and `-dump` or `-load` are mutually exclusive.

IMPORTANT!

- The directory referred to above must be a either a full physical UNIX or a full physical Windows pathname (UNC pathnames are *not* supported).
- If a directory is not specified, an error is displayed if you try to replicate from an "offline" subordinate site. Directory paths are required for this qualifier.

- **-m number**

  The lower limit for the number of items to be replicated. Replication occurs only if the number of items awaiting replication is greater than or equal to this number. The default number is 1.

- **-n number**

  The number of items to be submitted to the replication process at any one time. There may be a considerable number of items that require replication; therefore, to even out network traffic this option "chunks" the replicated items into groups of "n", and does not replicate a chunk until the preceding group has completed replicating. This chunking is performed automatically until all those items to be replicated have thus been processed. The default number is 100.

- **-no_bln_branch_check**

  When replicating baselines, do not perform strict branch validation.

- **-no_branch_check**

  When replicating baselines, items, or request, allow import of items without a branch ownership check. This enables the import of items on branches that are not owned by the sending site.

  This flag should be used if the following error is encountered:

  ```
  Error: Branch id "branch" is not owned by the sender site "SITE"
  ```

- **-no_replicate_mwsd**

  When replicating items, do not replicate those that were moved back to their owning site using directory rename.

NOTE The back replication of these items is not supported.

- **-o <filename>**

  Redirect all output messages to this the specified file.

- **-p <database>**

  The Dimensions database to connect to, for example, `qlarius_cm@dim10`. By default the value of DMDB environment variable is used.

- **-prune_logs**

  Remove all deleted replication logs from the Dimensions database.

- **-prune_old_logs <date>**

  Remove all replication logs older than the specified date from the Dimensions database.

  The date must be specified in the Dimensions standard format of `'DD-MON-YYYY HH24:MM:SS'` or `'DD-MON-YYYY'`.

- **-q**

  Generate a "what if" report of the operations that would have been undertaken if replication had been performed.

- **-r**

Items or requests are only to be received from the replica DB sites; no items or requests are to be sent.

See, also, the `-a` and `-s` flags.

- **-redo**

  Ignore the replication logs and resend everything. This option could be used to re-sync all the sites.

- **-replicate_offnorm**

  Allow the replication of items at an off normal/rejected status.

- **-replicate_remote_branches**

  Allow item revisions owned by a replica site to be replicated from the primary site to another replica site. This can be applied to both project and request replication when the requests have related items that are not owned by the sending site.

NOTE Baseline replication supports "foreign" item replication using the `-no_branch_check` option.

- **-s**

  Items or requests are only to be sent to the replica DB sites; no items or requests are to be received.

  The `-a`, `-r`, and `-s` flags can be used to further control the replication process. When `-r` is specified, only back-replication takes place, that is, the replication of objects from the replica sites to the primary. When `-s` is specified, only forward replication takes place, that is, the replication of objects from the primary to the replicas. The `-a` flag enforces the default behavior, which is to perform both back and forward replication.

- **-sc**

  Requests from the secondary catalog are also to be replicated.

- **-t <item_types>**

  A comma-separated list of item types that can be used as a filter to scope those items that are to be replicated. For example, `-t "SRC,TEXT,WORD"` only replicates items of types SRC, TEXT, and WORD. The default is all types.

NOTE Ensure that this list is enclosed within double quotation characters (" ").

- **-u**

  Resend the details on the replication configuration (specified through `-c`) to all replica DB sites. The default is to re-send whenever a configuration is updated.

  Using this option guarantees that the updated configuration is replicated to all replica sites.

- **-use_old_attr_format**

  Dump object attributes using pre Dimensions 9 format. Prior to Dimensions 9, attributes were dumped by attribute number; whereas, from Dimensions 9 onward they are dumped by attribute name unless this flag is set to revert to the old mechanism.

- **-x**

  Only create a log entry in the base database when there is an attempt to replicate one or more items. An attempt is made to replicate items only when the number of items to be replicated is equal to or greater than the lower limit set by the `-m` option. The default is for logging to occur whenever a replication process is undertaken regardless of whether there is any item to replicate or not.

  Use this option if replication is performed with a high frequency as it reduces the number of logs committed to the database.

Part 6

Archive, Retrieval, and Transfer Facilities

Introduction to Archive, Retrieval, and Transfer Facilities	391
ART Command Mode Applications	405
Viewing Archives and Transfers	421

Chapter 30

Introduction to Archive, Retrieval, and Transfer Facilities

Dimensions ART Overview	392
Access to Dimensions ART Facilities	393
Baselines for Archives	393
Creating an Archive	394
Deleting an Archive	394
Removing Online Archived Material	394
Restrictions	395
Transferring and Receiving Baselines	395
Transfer of Requests	398
Transfer of User-Defined Attributes	399
Transfer Baseline Identity	399
Tokens and Reports	400
Change Management Rules	401
Archive and Retrieval Across Dimensions Networks	401
Magnetic Tape Utility	402
Compatibility with Earlier Versions of Dimensions ART	402
Incompatibility Between MBCS and UTF-8 Database Character Sets	403

IMPORTANT! ART is supported with a Dimensions CM 8.0.1.6 or later server installation. However, archives created with earlier releases of Dimensions CM can be processed by Dimensions ART.

Dimensions ART Overview

The Archive, Retrieval, and Transfer (ART) facilities enable you to archive the contents of a baseline and store a copy on designated archive media. These media are referred to as archive volumes and each has an identity known to Dimensions ART as a Volume-id. An archive, which is effectively a copy, may be required for various reasons including satisfying procedural or contractual obligations. A special Release-baseline may be taken using the *ALL rule which includes in the baseline all revisions of each relevant item of the type specified regardless of lifecycle status.

Once a baseline is archived and a copy of this material is held on an archive volume, Dimensions ART provides facilities to remove certain material from Dimensions item libraries that are not delta-libraries. The effect of this is to reduce the amount of ONLINE disk space required. The removal is done by deleting archived material that is held ONLINE in Dimensions item libraries. Because this material is held on an archive volume it can be readily, securely, and selectively retrieved. The removal of ONLINE items frees up disk storage and the archive status of such items is known as OFFLINE. The RAMA and RAMP commands that perform this, and are described later in Chapter 31, "ART Command Mode Applications", apply certain rules to ensure that the archived material is not active and required by other users and baselines.

Dimensions ART also provides commands to retrieve such information back into the system and restore their status to ONLINE. The retrieval is based on material that has been taken OFFLINE and held on an archive volume such as a magnetic tape. Each such tape is identified by a Tape No and a Volume-id which is written physically on to the tape by Dimensions ART. The system permits retrieval of such items from a specified Volume-id, on a selective basis, the Tape No only being used for mounting purposes.

Incorporated within Dimensions ART are two commands that provide features to enable the contents of a baseline to be transferred to an archive area. This area can then be read by another Dimensions system as an incoming transfer of that baseline. There are conventions and standards that need to be agreed and adhered to, which enable two systems to transfer baselines. The concept here is that one Dimensions system is a donor, transferring a baseline and the other a recipient, accepting an incoming transfer. This means that information can be transferred between systems. The information about the project—at particular transfer-states defined for the lifecycles shared by the two systems involved in the transfers—can therefore be updated, given various checks provided by the system. The transfer-states are lifecycle states that are shared between the two systems and at which such transfers can occur.

Dimensions records the archive status of items in terms of OFFLINE and ONLINE and also ensures that an item's history is updated. This includes items that have been introduced into the system by an incoming transfer baseline.

Dimensions ART ensures that the volumes it creates cannot be overwritten by itself. A level of security is administered by Dimensions ART to ensure that an archive volume has not been tampered with and it uses checksums for this purpose. It is strongly recommended that adequate backups and security measures be taken of all archive volumes. The remove functions are dependent on being able to retrieve from these very

same volumes. Dimensions ART has a magnetic tape utility (MTU) that handles archive volumes and this is used to create and read archive and transfer volumes.

Access to Dimensions ART Facilities

IMPORTANT! To use the Dimensions ART functionality on Windows, you need to install a Windows version of the UNIX `tar` utility. To get a freeware version with associated source, see, for example, the Cygwin website: https://www.cygwin.com/

IMPORTANT! When installing Cygwin, the following packages are required: cygiconv-2.dll, cygintl-3.dll, `cygwin1.dll`, `cygintl-8.dll`, and `tar.exe`. Copy them to `%DM_ROOT\prog`.

All users have access to the viewing facilities of Dimensions ART available through the Administration Console. However, only users with General Administration Privileges are allowed to execute the Archive and Remove functions associated with the reading and the writing of archive and transfer volumes, which are available only from the `dmcli` command line interface as specified below.

The command...	is defined as...	and is available on...
CAR	Create Archive	UNIX and Windows
DAR	Delete Archive	UNIX and Windows
RAMA	Remove Archived Material selected by Archive	UNIX and Windows
RAMP	Remove Archived Material selected by Product	UNIX and Windows
RAI	Remove Archived Item	UNIX and Windows
RAT	Read Archive Tape	UNIX and Windows
ROA	Retrieve Offline Archive	UNIX and Windows
TBO	Transfer Baseline Out	UNIX and Windows
TBI	Transfer Baseline In	UNIX and Windows

Please refer to Chapter 32, "Viewing Archives and Transfers" for detailed descriptions of the ART viewing functions using the Administration Console**.**

Please refer to Chapter 31, "ART Command Mode Applications" for detailed descriptions of the functions using command mode.

Baselines for Archives

Design-baselines cannot be used for archives because the item revisions therein may continue to be modified, but all release-baselines are valid inputs for Dimensions ART. A special kind of baseline has been provided (the `*ALL` baseline), which contains all

revisions of the item-type specified regardless of lifecycle state. This *ALL baseline cannot be used for configuration-build or for release-control purposes.

Creating an Archive

The Create Archive (CAR) command enables items that have been baselined to be copied to a designated archive volume for storage. The process involves the selection of a baseline that is required to be archived.

An archive is not created if any item in a baseline that is required to be archived is OFFLINE at the time the archive is to be made.

A working space on disk called "pcmsartwork" is required by Dimensions ART to perform its archive operations and consequently has to be specified. This area, assigned by the system manager, is usually a temporary area that also has enough disk space to complete the operation. The default is the current working directory. Archiving together with removing archive material should not be done to free up disk space when the disk is nearly full, as there may not be sufficient disk work space to perform the appropriate functions. We recommend that this is done on a regular basis.

An archive volume is identified by a Volume-id and a tape number or Tape No. The Volume-id is a user-specified identification physically written on the archive volume as an identification that is read to ensure that it is the correct archive or transfer volume. The Tape No is effectively the human-readable "sticky label" that is affixed to the outside of the volume and is normally the identification that a tape librarian uses to fetch it from a rack of tapes.

It is essential that the "ART Work Area" is large enough to hold the entire contents of a baseline. Dimensions ART does not support multiple-volumes.

All the management of archive devices and media is done by the Dimensions MTU (magnetic tape utility) facility. For details, see "Magnetic Tape Utility" on page 402.

Deleting an Archive

The Delete Archive (DAR) command enables an archive to be deleted. All items contained in the archive and within the scope of the owning product need to be ONLINE before a DAR may be performed. All information relating to the archive, its volume and Tape No is removed from the Dimensions database. All history of the items such as status change from OFFLINE and ONLINE is retained.

Removing Online Archived Material

There are two remove-archive-material commands, and they enable certain items contained in archives to be deleted from ONLINE disk space. This is performed by the two commands RAMA and RAMP. The first, RAMA (Remove Archived Material selected by Archive), operates across a single specified archive; and the other, RAMP (Remove Archive Material selected by Product), operates across all archives within the specified product.

The removal is achieved by deleting archived material from Dimensions item libraries that are not delta libraries, and these items are then given a status of being OFFLINE. These items can then be brought ONLINE by retrieving them from an appropriate archive volume. The rules governing the removal of such items and placing them in an OFFLINE status are:

- Item revision should be recorded in one of the archives belonging to the product.
- Item revision should not belong to another product.
- Item revision should not be referenced by an open request.
- Item revision should not be the latest.

An option is provided for both RAMA and RAMP to operate without imposing the last two of the above conditions so that items can be placed OFFLINE even if they are the latest, or are referenced by an open request.

RAMP operates across all archives within the specified product, and RAMA operates on a single archive. Neither RAMP nor RAMA removes archive material belonging to another product.

Restrictions

Dimensions ART uses product-ids and item types to form subdirectory names, and these must conform to the same restrictions as the operating system imposes on directory names.

The TBI command does not work correctly for any design parts that have been SUSPENDED either in the receiving product or in the product from which the TBO medium was prepared. Users are recommended not to suspend any such design parts.

In the TBI command, if errors are detected during an inward transfer (such as disk full), the entire transfer is "rolled back", and any files involved are removed from item libraries.

Transferring and Receiving Baselines

Two commands are provided by Dimensions to transfer the contents of a baseline and to receive this baseline by another Dimensions system. The TBO – 'Transfer Baseline Out' command provides features that enable the contents of a baseline to be transferred to a TBO area. This area can then be read by another Dimensions system as an incoming transfer of that baseline. There are conventions and standards that need to be agreed and adhered to, in order to enable two systems to transfer baselines. The concept here is that one Dimensions system is a donor, transferring a baseline, and the other a recipient, accepting an incoming transfer by the use of the TBI – 'Transfer Baseline In' command. This means that information can be transferred between systems. The information about the project, at particular crossover states defined for the item lifecycles and request lifecycles shared by the two systems involved in the transfers, can therefore be updated, given various checks provided by the system.

The intended use of TBO and TBI is to provide commands to enable transfers. The concept of transferring a baseline between two systems is a complex one involving, for example, design part categories and structures, usage links, item-type, and associated item

lifecycles, building and release procedures. Two systems that are intended to exchange baseline information need clearly identified objectives and procedures that are to be followed. The design part structure should be cognizant of this, particularly regarding the usage links that are to be used. A baseline strategy also needs to be carefully considered as well, clearly defining whether the transfers are in one direction or whether they are to be bi-directional. The design part structure of the donor system and the receiving system are very important considerations.

As an example, the top part of the baseline has to have a design part in the receiving system to be its parent. Clearly, neither this top part nor any other design part can have the same `Part-id` as the receiving parent design part to which it attachs. In addition, if the incoming top part is also the top design part for the product, it has a category of `PRODUCT`. Therefore, if the transfer is permitted, there is a conflict, as there are two design parts having a category of `PRODUCT`, which is not permitted. When a TBI is performed, a replacement for the category `PRODUCT` has to be selected from the range of categories that are defined for the receiving product. Otherwise, the transfer is not permitted. An additional check is also made to ensure that an incoming `Part-id` is not the same as the `Product-id` of the receiving product. Otherwise, there is a serious conflict and a transfer is not permitted.

An incoming design part breakdown structure takes precedence over the existing breakdown structure of the receiving product, and so the structure of the receiving product is changed to reflect the structure of the incoming transfer. Usage relationships are merged in the incoming transfer, and any unwanted links have to be changed manually. If a donor system contains a design part in the transfer baseline that has no parent design part as a result of it being in the baseline through, for example, a cross-product `USAGE` relationship, then the transfer is not permitted, as it does not have a parent design part to which to attach in the receiving system. In a simple case where the `USAGE` relationship is within the product, you may need to take a baseline at a higher design part to retain the relationships to be transferred.

Incoming attributes for design parts and product items are checked by `TBI` for variable names known to the receiving system, and to ensure compatible data types. No check is now performed for missing mandatory attributes: These are detected on actioning an item and can be rectified at that time. Also incoming attribute values are not checked for length or against any applicable valid set.

Incoming attributes are accepted only in `TBI` for new design parts, for part variants, or for new product item revisions that are being added to the receiving product. The attributes in the receiving product for existing design part variants and product item revisions are not changed by `TBI`.

The design part structure, item relationships, and at which design part a baseline is performed, all need to be considered as part of a product's design, its build, and its release, as well as to establish a control structure to manage the project.

If a `USED` item belongs to a design part not in the baseline, the transfer is permitted with an exception in the report identifying the item that cannot be transferred, together with the owner design part. An issue like this can be handled again by perhaps taking a baseline at a different level and including the owning design part. This may not be possible and therefore it may be necessary to transfer a separate baseline containing the owning design part. At the receiving end, the exception is known. It is necessary to take the second transfer to receive the owning design part, and then to establish the item's `USAGE` relationship manually. It may not be the general case, but if the receiving system is to do a build, it needs to be modified to accommodate the change in this relationship. In addition, consideration needs to be given to avoid the potential for conflicts resulting from

identical item-ids and filenames in both donor and recipient systems. After these sorts of changes are established, a transfer procedure is in place that can be used effectively.

In TBI, any branches applicable to incoming items are checked against the branches defined as valid on the receiving system. Items are accepted (subject to the other checks) if they have either no branch or a valid branch. The branch for an item is given by the leading characters in the revision field, up to (but not including) the first '#' character. If there is no '#', the item has no branch.

Transfer of all user-defined attributes, including those for the design parts and items in the baseline, is described in a separate topic below.

CAUTION! The TBO command fails if the donor Product Baseline type has associated user-definable attribute definitions.

Product structures and item relationships can be complex, by their very nature, and it is essential that the transfer facilities be applied at a practical level to meet the objectives of the project. Conventions and standards need to be agreed.

Enhancements to Early Versions of TBO/TBI

The following TBO/TBI enhancements were first introduced with Dimensions 8:

- TBO and TBI support using a directory as a source and target for ART operations, in addition to the previous support of archive files.
- TBI no longer imports a baseline unless it has at least one item that would have been successfully created. Previously, baselines were imported even if no items would have been successfully created. Additionally, enhancements have been made to the way data is cleaned up after a failed TBI operation. These enhancements provide better error diagnostics as to why an import may have failed.

Cross-Platform Transfers

You can share archives between Windows and UNIX systems and import the archive on UNIX or Windows regardless of origin. To perform a cross-platform transfer, understand the following considerations:

- **Using the TBO on UNIX**

  When using the TBO command on UNIX, you must store the archive in the disk area. Set the /DEVICE parameter to NONE and include a path to the target location in the /DIRECTORY parameter (/VOLUME and /TAPE are not required when /DEVICE is set to NONE).

- **Using the TBO on Windows**

  When using TBO on Windows, the archive always goes to the path indicated in the /DIRECTORY parameter (/DEVICE, /VOLUME, and /TAPE are not required).

- **Transfer the baseline directory to the target system**

  After you set up a directory, transfer the baseline directory to the target system using DVD/CD, FTP, file copy, or similar method. You may want to use a compression utility as archives can be quite large. When you transfer the archives, use a binary transfer to exclude conversion of text documents since ART expects to handle this itself.

- **Using TBI on UNIX**

When using TBI on UNIX, set the /DEVICE parameter to NONE and the /DIRECTORY parameter should contain the path of the incoming archive (/VOLUME and /TAPE are not required).

- **Using TBI on Windows**

  When using the TBI on Windows, the /DIRECTORY parameter should contain the path of the incoming archive (/VOLUME, /DEVICE, and /TAPE are not required).

- **Transferring older archives across platforms**

  When performing a cross-platform transfer with an archive created with an earlier version of Dimensions ART (version 1.5 or earlier), you need to include the /SOURCE_OS parameter on the TBI command to ensure that items are converted properly. If you do not include this parameter with older archives, TBI produces an error message and aborts processing.

Routing Errors Accepted by TBI

If the TBI command attempts to place incoming items in the pending lists for appropriate users, as determined by the process model for the receiving product, any routing errors are reported to the user, but the TBI command continues to run. The user should correct the cause of any such routing error (often by assigning roles to suitable users), and then use the Dimensions PEND CHDOC dmcli command to route the items involved appropriately.

Transfer of Requests

An option is provided in the TBO and TBI commands to allow Dimensions CM requests to be transferred between systems, in association with baseline transfers.

Dimensions CM requests may be selected for transfer if they are related to any item in the baseline being transferred. On receipt, a request is assigned a new request identity with a serial number following on from the existing requests on the receiving product. The incoming request is routed to users in accordance with the current role assignments on the receiving product. If it is impossible to route any incoming request, the entire TBI transaction is rolled back and any files involved are removed from the libraries.

In the receiving system, requests may be accepted by the TBI command, if they have not previously been transferred into the receiving product.

Further checks are performed on incoming requests, similar to those for incoming items. The request type must be defined in the receiving product. The request must be at a crossover lifecycle state—one that is mapped in the receiving product. No check is made to see whether the receiving product has a request template defined at TBI time, though one is needed before the document can be browsed, printed, or gotten. (This is to allow templates and requests to be transferred simultaneously.)

When a request is accepted on the receiving product, it is assigned a new request identity, as if it were a freshly created request. A report is produced at the conclusion of the TBI command showing the original and the newly assigned request identifiers for all requests transferred into the receiving product. The incoming request is slotted into the appropriate lifecycle for its type, at the same state as it had in the sending system, and it is routed to users with the appropriate roles. If no such users can be found, an error message is returned, and the entire TBI transfer is aborted.

The request is also assigned, where possible, relationships to design parts, and items corresponding to all such relationships that it had in the sending system. Any part or item relationships that cannot be maintained are reported. Relationships to other requests are not transferred to the receiving product.

Transfer of User-Defined Attributes

Transfers between systems include user-defined attributes for the design parts, items, and requests involved.

Any archive generated by the TBO command includes the current set of attributes for each design part and item revision in the baseline being transferred, and for each request transferred. As the attribute values are not recorded in a baseline, the values transferred are those at the time that the TBO command is performed.

In the TBI command, attributes are set up from the archive for any new incoming design part, item revision, or request. But no action is taken in respect of the attributes applicable to any design part variant, item revision, or request already present in the receiving system.

Attribute definitions are not transferred as such between systems. Instead it is for the receiving system to ensure that the process model used for the receiving product has appropriate definitions to enable it to accept all required incoming attributes.

The checks on attributes performed by the TBI command are based on the variable name associated with each attribute in the sending system. The same variable name must be defined for an attribute belonging to the relevant design part category, item type, or request type on the receiving product. The data types also must match.

No checks are made for the presence or absence of any mandatory attributes. Nor are the values of incoming attributes checked against valid sets. Any incompatibilities in these areas become apparent when an item or request is actioned, and can easily be corrected at that time. It is quite possible that the corrective action may be to modify the receiver's process model in the light of the incoming data.

Transfer Baseline Identity

A "transfer baseline out" is known by its TBO-id. This TBO-id is a concatenation of "Location", "Date/time", and an Identity. We recommend that you agree on a standard for Location to ensure that two sites, or two products, do not use the same "Location". The TBO-id remains unique, but from a management point of view it may be clearer, in terms of reports that are generated by Dimensions, to have a simple differentiation. The Date/time is added to ensure that the same baseline can be transferred as many times as necessary while maintaining a unique TBO-id. A default for the "Identity" is the baseline-id itself, as it describes the contents of the transfer. However it is user-definable and so the default can be changed.

Tokens and Reports

Token checking can be performed between two systems that are to donate and receive transfer baselines. The intent behind this type of checking is that, ahead of the actual transfer of a baseline, a Token is generated using the TBO command and this Token is read by the receiving Dimensions system. The Token establishes whether a potential incoming transfer may be accepted. Exception reports are produced enabling the management of the two transfer systems to discuss and take corrective actions to ensure a successful Token check before the actual transfer. In effect the Token check performs a trial run of the transfer.

Token checking for an incoming transfer depends on what could be included at the time the Token was created. The checks include: whether item types are known; whether an item library exists; and whether, for new design parts, a category exists. In addition, a check is made to ensure that incoming lifecycle states exist. Otherwise, it is not possible to receive the item. The generated report is dependent on what was included in the Token when it was taken. A Token transfer is permitted even if items are OFFLINE. In the normal transfer case, this would be stopped, as a committed transfer is not permitted if items are OFFLINE.

Trial runs may be performed by selecting the REP option on TBO and TBI. This runs the actual job without committing the results. However, all normal checks are conducted as if it were a real execution. In the case of a TBI, we strongly recommend that a trial run is performed using the REP option before an actual run is committed.

Composite reports are produced that have a similar structure for the three options of Token, REP, and Commit. The following type of information is provided for all options:

- **Information on design parts:**
  - Existing design parts detailing new variants and new breakdown relationships.
  - Design parts with unrecognized categories.
  - Undefined attributes for design parts.
  - Attributes of incompatible data types for design parts.
  - New design parts and their owners.
- **Information on items:**
  - Undefined item types.
  - Undefined item libraries.
  - Unmatched lifecycle states for items.
  - Undefined attributes for items.
  - Attributes of incompatible data types for items.
  - Items already present.
  - Items changing ownership.
  - Items to be transferred.
  - Items that were OFFLINE in a token report.
- **Information on requests:**
  - Undefined request types.

- Unmatched lifecycle states for requests.
- Undefined attributes for requests.
- Attributes of incompatible data types for requests.
- Requests not related to incoming items.
- Requests to be transferred.

Change Management Rules

All Dimensions ART operations are carried out independently of any change management rules that may currently be in force for the product concerned. No requests are required to create (or delete) an archive, to remove `ONLINE` items, or to retrieve items from an archive. Dimensions ART does not remove any `ONLINE` archived items from the libraries related to any open request.

Similarly, `TBO` and `TBI` are independent of change management rules. The preparation of a transfer by `TBO` is similar to the creation of an archive, and does not affect the structure or content of the sending product.

An incoming transfer through `TBI` involve the creation of new items or new revisions of existing items in the receiving product. These may be at any lifecycle state (so long as the state name is used in both sending and receiving systems). It may well be that the receiving product uses rules requiring the existence of related requests for such items. In such a case, it may well be advisable for the transfer to include requests of appropriate types.

It is strongly recommended that any incoming transfer should first be checked with the `TBI /REPORT` option. Plans can then be made for any requests or relationships to items that are needed after performing the incoming transfer.

Archive and Retrieval Across Dimensions Networks

If Dimensions ART is being used in a homogeneous network configuration, where both the Dimensions database server and all its clients are using the same or compatible operating system types, then there are no special considerations that need to be addressed. The authorized user can use Dimensions ART without restrictions from any Dimensions node. All items can be archived from any client or server node, as the operating system recognizes and manages all the various file types consistently.

Magnetic Tape Utility

NOTE The Magnetic Tape Utility (MTU) can also be configured to use other devices such as writable CDs or DVDs if so desired.

Predefined example files dm_mtu.ucg (Windows) and dm_mtu.ucg.disk (UNIX) are provided (in the Dimensions "root" MMI subdirectory) to demonstrate how the MTU utility can be configured. For Windows, this example has been written to use zip files, but you can configure it to use any media-writing software.

MTU is invoked by Dimensions ART to read and write archive and TBO volumes. It is responsible for ensuring that the correct volumes are mounted, that the volume that is read is correct, and that the tape has not been interfered with since it was generated. This is done by the use of checksums.

The basic operations of MTU are to mount a tape, to initialize it, to read and write it, and to do basic mounts and dismounts. Before a tape is initialized to write an archive or a TBO, a check is made to ensure that it is not a Dimensions generated tape. MTU does not overwrite such a generated tape.

An archive volume is identified by a Volume-id and a tape number or Tape No. The Volume-id is a user-specified identification physically written on the archive volume as an identification that is read to ensure that it is the correct archive or transfer volume. The Tape No is effectively the "sticky label" which is human readable that is affixed to the outside the volume and is normally the identification that a tape librarian uses to fetch it from a rack of tapes.

MTU requests the appropriate tape by its Tape No, which is the label that the Tape Librarian may use to obtain it from a Tape Library. The Tape No is only used for this purpose and, apart from recording the single relationship between a Tape No and a Volume-id, no further use is made of it. Dimensions ART including its MTU utility uses the Volume-id for all other relevant purposes.

If the Volume-id check fails the user is invited to retry mounting a different tape. MTU does not check the Tape No and relies entirely on the Volume-id because the user may be using a copy of the primary archive or TBO Volume. If retrieving from several volumes, MTU and Dimensions ART process each volume in turn, inviting the user to mount the appropriate tape using the original Tape No supplied when the volume was originally created.

The Dimensions MTU dialog is described in "Magnetic Tape Utility" on page 419.

Compatibility with Earlier Versions of Dimensions ART

The current releases of TBO and TBI accept input from several (but not all) earlier releases of Dimensions ART. The general principle is that input is accepted from Dimensions ART 1.3 and later releases, but that output from the current release is not compatible with earlier releases. Removable media prepared by the current release include distinguishing data, and if they are presented to earlier versions of Dimensions ART, an appropriate error message is generated.

This means that users upgrading to the current release of TBO and TBI can continue to use existing archive removable media, and need take no special action with regard to

these, other than normal maintenance procedures (such as periodical rewinding and so on). The same applies to removable media from earlier releases, subject to special action with regard to request attributes transferred by Dimensions ART 1.3.x.

Dimensions ART 1.4.x does not offer compatibility with Dimensions ART 1.2.x or earlier versions.

In Dimensions ART 1.5, new information about the current baseline project and donor operating system is kept in the baseline files put there by the TBO command. When importing a baseline using TBI, it uses this information to import the baseline items properly. If you are importing a baseline created with an earlier version of ART, you need to specify project and source operating system on the command line to ensure that the import works properly.

Incompatibility Between MBCS and UTF-8 Database Character Sets

The current version of Dimensions ART does not support the transfer of data between systems that are running MBCS and UTF-8 database character sets. Any attempt to do so results in high order ASCII characters, such as umlauts, being incorrectly transferred.

Chapter 31

ART Command Mode Applications

Introduction	406
CAR - Create Archive	407
DAR - Delete Archive	409
RAMA - Remove Archived Material Selected by Archive	410
RAMP - Remove Archived Material Selected by Product	411
RAI - Remove Archived Item	412
RAT - Read Archive Tape	413
ROA - Retrieve Offline Archive	414
TBO - Transfer Baseline Out	415
TBI - Transfer Baseline In	417
Magnetic Tape Utility	419

Introduction

This section lists the Archive, Retrieval, and Transfer functions (ART) that are available in command mode together with the parameters and qualifiers.

The syntax conventions used here are the same as those for the general Command Mode Applications, and are explained at the start of the *Command-Line Reference*. The commands are listed in the order shown below

The command...	is defined as...	and is available on...
CAR	Create Archive	UNIX and Windows
DAR	Delete Archive	UNIX and Windows
RAMA	Remove Archived Material selected by Archive	UNIX and Windows
RAMP	Remove Archived Material selected by Product	UNIX and Windows
RAI	Remove Archived Item	UNIX and Windows
ROA	Retrieve Offline Archive	UNIX and Windows
RAT	Read Archive Tape	UNIX and Windows
TBO	Transfer Baseline Out	UNIX and Windows
TBI	Transfer Baseline In	UNIX and Windows

PRIVILEGES Only users with General Administration privileges can use these command-mode functions.

CAR - Create Archive

```
<archive-id>
   /BASELINE=<baseline-spec>
   /DEVICE=<device-id>
      or /DEVICE=NONE
   /TAPE=<tape no.> /VOLUME=<volume-id>
   [ /DESCRIPTION=<description>]
   [ /DIRECTORY=<directory>]
   [ /REPORT ]
   [ /FORCE ]
```

Example

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

- **<archive-id>**

  is the identity to be given to the archive.

- <**baseline-spec>**

  identifies the baseline from which the archive is to be created and comprises:

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

- **<device-id>**

  is the device used to write the tape on. If you are writing to disk, this should be set to `NONE` and a path should be specified in `/DIRECTORY`.

- **<tape no.>**

  This refers to a "sticky label" stuck to the volume, for example, a Tape number.

- **<volume-id>**

  This is the name of the volume being written.

- **<description>**

  is optional text describing the archive to be created. If this contains any spaces, it needs to be enclosed in double quotes (`"`).

- **<directory>**

  specifies a directory (to which the user has write permission) to be used as the `pcmsartwork` work space. This must be in the standard format for the operating system and it may be in absolute form or relative to the current default working directory.

  If omitted, the current default directory is used as work space.

- **/REPORT**

  specifies that only a predictive report is required, and no archive is to be produced. It is generated in the file `carrep.out`. This report also gets generated when `CAR` is run for real.

- **/FORCE**

  Befor Dimensions 2009 R1, if you ran the same CAR command twice:

  - The first time, the archive is created.
  - The second time, the message

    ```
    ART0005710E Duplicate Archive Identity
    ```

    is generated to indicate that the archive already exists, but the archive itself is deleted.

  Dimensions 2009 R1 and later: If a directory for the archive already exists, it is not deleted (by running a second CAR command). To revert to the old behavior where the archive is deleted, you need to specify the `/FORCE` qualifier.

DAR - Delete Archive

```
<archive-id>
```

Example

```
DAR AA12AB
```

- **<archive-id>**

  is the identity of an archive which is to be deleted.

Additional Restriction

All items in the archive to be deleted, and belonging to the product owning the archive, must be ONLINE in their item libraries.

RAMA - Remove Archived Material Selected by Archive

```
<archive-id>
   [ /NOCHECK ]
```

Example `RAMA ARCH_BL5`

- **<archive-id>**

  specifies the archive from which archived items belonging to the specified product are to be removed from the item libraries that are not delta-libraries.

- **/NOCHECK**

  specifies that only a relaxed set of rules is used to govern the removal of items (see below).

Item Removal Rules

The rules governing the removal of such items and placing them in an `OFFLINE` status are:

- Item revision should be recorded in one of the archives belonging to the product.
- Item revision should not belong to another product.
- Item revision should not be referenced by an open request.
- Item revision should not be the latest.

If `/NOCHECK` is specified, the last two of these rules are not applied, so that item revisions can be placed `OFFLINE` even if they are the latest or are referenced by a request. In the absence of this qualifier the command behavior is unchanged, and product item revisions are not placed OFFLINE if they are the latest revision or are related to an open request.

RAMP - Remove Archived Material Selected by Product

```
<product-id>
   [ /NOCHECK ]
```

Example

```
RAMP PRODX
```

- **<product-id>**

  specifies the product for which archived items belonging to the product are to be removed from the item libraries that are not delta-libraries.

- **/NOCHECK**

  specifies that only a relaxed set of rules is used to govern the removal of items (see below).

Item Removal Rules

The rules governing the removal of such items and placing them in an OFFLINE status are the sane as described for RAMA on page 410.

RAI - Remove Archived Item

```
<item-spec>
   [ /FILENAME=<filename>]
   [ /NOCHECK ]
```

Example `RAI "PRODX:DECODER.AAAA-SRC;1"`

- **<item-spec>**

  specifies an item that is to be removed from its item library and comprises

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

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

- **<filename>**

  specifies the name of the file containing the item in the item library.

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

- **/NOCHECK**

  specifies that only a relaxed set of rules is applied to determine whether the item may be removed from its library (see below).

Item Removal Rules

The rules governing the removal of such items and placing them in an `OFFLINE` status are the same as described for `RAMA` on page 410.

RAT - Read Archive Tape

```
<archive-id>
   /DEVICE=<device-id>
      or /DEVICE=NONE
   /TAPE=<tape no.> /VOLUME=<volume-id>
   [ /DIRECTORY=<directory>]
```

Example

```
RAT AA12AB /DEVICE="/dev/rmt0h" -
   /TAPE="aa100"/VOLUME="bb100"
```

- **<archive-id>**

  specifies the identity of the archive to be read from tape.

- **<device-id>**

  specifies the device used to read the tape on. If you are reading from disk, this should be set to `NONE` and a path should be specified in `/DIRECTORY`.

- **<tape no.>**

  refers to a "sticky label" stuck on the volume, for example, a Tape number.

- **<volume-id>**

  is the name of the volume being read.

- **<directory>**

  specifies a directory (to which the user has write permission) to be used as the `pcmsartwork` work space. This must be in the standard format for the operating system and it may be in absolute form or relative to the current default working directory.

  If omitted, the current default directory is used as work space.

Purpose

To read an archive tape containing a specified archive. The specified archive identity is checked against the list of known archives held in the database and against the archive identity recorded on the tape.

ROA - Retrieve Offline Archive

```
<archive-id>
   [ /DIRECTORY=<directory>]
```

Example `ROA AA12AB`

- **<archive-id>**

  specifies the archive from which archived items are to be retrieved to the item libraries.

  It must be preceded by `RAT` on the appropriate tape.

- **<directory>**

  if you are not in the directory where the archive is located when running `ROA`, then you must specify the correct directory with this qualifier.

Purpose To retrieve an entire offline archive.

TBO - Transfer Baseline Out

```
<tbo-id>
   /BASELINE=<baseline-spec>
   /DEVICE=<device-id>
     or /DEVICE=NONE
   /TAPE=<tape no.> /VOLUME=<volume-id> ]
   [ /DESCRIPTION=<description> ]
   [ /DIRECTORY=<directory> ]
   [ /REPORT
     or /TOKEN ]
   [ /CHANGE_DOC_IDS= (<chdoc-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 & chdocs"
```

Example

```
TBO TB12AB /BASELINE="PRODX:BL12AB" -
   /DIRECTORY="c:\usr\smith\work"" -
   /CHANGE_DOC_IDS=(PR,CR) -
   /DESC="12AB transfer - sources & chdocs"
```

- **<tbo-id>**

  is the identity to be given to the transfer record.

- **<baseline-spec>**

  identifies the baseline which is to be transferred to another system and comprises

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

CAUTION! The TBO command fails if the donor Product Baseline type has associated user-definable attribute definitions.

- **<device-id>**

  is the device used to write the removable media on. If you are writing to disk, this should be set to `NONE` and a path should be specified in `/DIRECTORY`.

- **<tape no.>**

  This refers to a "sticky label" stuck to the volume, for example, a Tape number.

- **<volume-id>**

  This is the name of the volume being written.

- **<description>**

  is optional text describing the transfer record to be created.

- **<directory>**

  specifies a directory (to which the user has write permission) to be used as the `pcmsartwork` work space. This must be in the standard format for the operating system and it may be in absolute form or relative to the current default working directory.

  If omitted, the current default directory is used as work space.

- **/REPORT**

  specifies that only a predictive report is required, and no data is to be output. The report is generated in the file `tborep.out`.This report also gets generated when `TBO` is run for real.

- **/TOKEN**

  specifies that a transfer of tokens is required, rather than a full baseline transfer. The recorded data includes a contents list and database information but does not include any item or request files. (This facility enables users to check on the acceptability of a future baseline transfer without needing to produce an archive containing large quantities of data from item and request files.)

NOTE `/REPORT` and `/TOKEN` may not be specified at the same time.

- **/CHANGE_DOC_IDS=** (<request-type>, ...) or **/CHANGE_DOC_IDS= ***

  specifies that requests are to be included in the outward transfer. For a request to be included, at least one of the item revisions included in the transferred baseline must be related to it.

  If a `<request-type>` list is given, included requests are restricted to the types specified. If * is specified instead of a list, all qualifying requests are included.

  If the `/CHANGE_DOC_IDS` qualifier is omitted, or if `TBO` is invoked in interactive mode, no requests are transferred.

TBI - Transfer Baseline In

```
<tbo-id>
   /PART=<part-spec>
   /DEVICE=<device-id>
     or /DEVICE=NONE
   /TAPE=<tape no.> /VOLUME=<volume-id> ]
   /CATEGORY=<replacement-category>
   [ /DIRECTORY=<directory> ]
   [ /REPORT
     or /TOKEN ]
   [/CHANGE_DOC_IDS= (<chdoc-type>, ... )
     or /CHANGE_DOC_IDS= * ]
   [/WORKSET=<project-id> ]
   [ /SOURCE_OS= WINDOWS
     or UNIX]
```

Example

```
TBI TB12AB /PART="PRODY:P123" -
   /CATEGORY=MODULE /CHANGE_DOC_IDS=* -
   /DEVICE="/dev/rst0" /TAPE="ta100" /VOLUME="tb100" -
   /DIRECTORY="/usr/jones/work"
```

Example

```
TBI TB12AB /PART="PRODY:P123" -
   /CATEGORY=MODULE /CHANGE_DOC_IDS=* -
   /DIRECTORY="c:\usr\smith\work"
```

- **<tbo-id>**

  The identity given to the transfer record when it was created on another system.

- **<part-spec>**

  Identifies the design part in an existing product that is to be the parent part for the top part of the baseline being transferred, and comprises:

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

- **<device-id>**

  The device used to read the removable media on. If you are writing to disk, this should be set to `NONE` and a path should be specified in `/DIRECTORY`.

- **<tape no.>**

  Refers to a "sticky label" stuck to the volume, for example, a Tape number. Not required if `/DEVICE` is set to `NONE`.

- **<volume-id>**

  The name of the volume being read. Not required if `/DEVICE` is set to `NONE`.

- **<replacement-category>**

  A design part category defined on the receiving product that replaces the design part category for incoming design parts when this would otherwise be `PRODUCT`.

- **<directory>**

  Specifies a directory (to which the user has write permission) to be used as work space. This must be in the standard format for the operating system and it may be in absolute form or relative to the current default working directory.

If omitted, the current default directory is used as work space.

- **/REPORT**

  Specifies that only a report is required, and no items or requests are to be moved into libraries and no changes are to be made to the database. This option may be used only in connection with a full transfer record (one that was prepared to include item and/or request files). We strongly recommend to use it as a final check before a full baseline is transferred to the receiving system. The report is generated in the file `tbirep.out`.This report also gets generated when `TBI` is run for real.

- **/TOKEN**

  Specifies that a transfer of tokens is required, rather than a full baseline transfer. This option may only be used in connection with a token transfer record (one that was prepared with the `/TOKEN` option). (This facility enables users to check on the acceptability of a future baseline transfer without needing to produce removable medium containing large quantities of data from item and request files.)

NOTE `/REPORT` and `/TOKEN` may not be specified at the same time.

- **/CHANGE_DOC_IDS=** (<request-type>, ...) or **/CHANGE_DOC_IDS= ***

  Specifies that the inward transfer (or the checks done for `/REPORT` or `/TOKEN`) is to include requests from the transfer record.

  If a `<request-type>` list is given, only requests of the listed types are transferred. If the list is simply *, all requests from the transfer record are transferred regardless of type.

  If the `/CHANGE_DOC_IDS` qualifier is omitted, or if `TBI` is invoked in interactive mode, no requests are transferred.

  For tapes on UNIX, `TBI` must be preceded by a `RAT` on the relevant tape.

- **/SOURCE_OS**

  Specifies the OS version of the donor system during the import. It tells Dimensions which operating system the file is transferred from, UNIX or Windows.

NOTE Entering VMS results in an error.

- **/WORKSET**

  Specifies the project on the target system to import the baseline into. This project must already exist and be capable of accepting the incoming baseline. This parameter is required only when using archives created with an earlier version of Dimensions ART. If the `/WORKSET` parameter is not specified, the incoming baseline is tied to `$GENERIC:$GLOBAL`.

Dimensions 8, or later Enhancements

`TBI` does not import a baseline unless it has at least one item that would have been successfully created. Previously, baselines were imported even if no items would have been successfully created.

Additionally, enhancements have been made to the way data is cleaned up after a failed `TBI` operation. These enhancements provide better error diagnostics as to why an import may have failed.

Magnetic Tape Utility

Purpose

This is the utility program (DM_MTU) used by Dimensions ART to Write and Read Archive and TBO volumes.

Description

The utility has a dialog with the user in order to ensure that correct archive and TBO volumes have been mounted for reading and writing.

To copy an MTU tape, use the normal UNIX tape copying method (dd).

MTU dialog

- When MTU initializes a tape:

  It first checks that the tape was not created by MTU. Under some circumstances, as it checks, it sends error messages to the screen (such as "File not found") before the tape is written to. These should be ignored, and MTU outputs a specific error message if there is a problem.

  The list of files sent to the tape is displayed on the screen, as they go onto the tape.

- When MTU reads a tape:

  the filenames being read are displayed on the screen.

  MTU checks Dimensions ART markers on the tape to ensure that the tape (or a copy) is one that MTU expects.

Messages from MTU

1. MTU: Operation completed.
2. MTU: Operation has failed.

   If the operation has failed, a message: "FAILED would you retry the operation? (Y/N)" is displayed and you are invited to respond by either retrying the volume or with another volume or terminating with N.

3. An MTU operation is already in progress

   You cannot have two MTU operations in progress in a single directory.

4. Cannot create file <filename> on device name
5. Cannot write file <filename> on device name
6. The symbol DM_MMI is not defined

   This indicates that the configuration file is not present and this should be referred to the system manager of your Dimensions system.

7. Cannot read file <filename> on device <device name>
8. There is a syntax error in the <filename> file.

   The configuration file has an error within it. Refer to the system manager.

9. The variable <variable> is not valid in the config file

   There is an unknown replacement variable in the configuration file. Refer to the system manager.

10. Tape in device <device name> is a Dimensions ART created tape, please re-initialize it if you do not want it.

    MTU was requested to write to a tape, but the drive held a tape that was previously created by MTU, and so MTU does not proceed with the transaction.

11 Tape in device <device name> is NOT a valid Dimensions tape.

MTU was requested to read a tape, but it was not a valid MTU-generated tape, and so cannot proceed.

12 Directory specification <directory spec> must be relative to working directory

The directory to transfer to the tape must be relative to the current working directory.

13 The tape read has a volid of <Volume_id> and is the wrong tape.

MTU was requested to read a tape but the volid (volume identification) of the tape was not the one that was requested.

Dimensions 8, or later Changes

Two predefined example files dm_mtu.cfg (Windows) and dm_mtu.ucg.disk (UNIX) are provided (in the Dimensions "root" MMI subdirectory) to demonstrate how the MTU utility can be configured. For Windows, this example has been written to use zip files, but you can configure it to use any media-writing software.

Unlike earlier releases of Dimensions, the DM_MTU utility is no longer called interactively. When an ART command is submitted via the command line, the command runs immediately, and there is no prompting to determine if the ART device is ready.

Chapter 32

Viewing Archives and Transfers

About Archives and Transfers 422
Archives and Transfers Main Window 422
Viewing Archives and Transfers 424

About Archives and Transfers

Purpose

Archives and Transfers in the Administration Console allow you to view the details and contents of:

- **All:** All the following.
- **Archive:** A baseline that has been copied to an archive volume for storage purposes.
- **Token Transfers:** A trial run of a baseline transfer that is used to identify compatibility problems before a full baseline transfer is performed.
- **Transfer Baselines:** A baseline that has been transferred via the `TBI` or `TBO` functions.

Invocation

Dimensions Administration Console | Distributed Development | Archives & transfers

Archives and Transfers Main Window

The Archives and Transfers main window consists of the following:

- **Menu area:** Displays the toolbar, see "Archives and Transfers Menu Area" on page 422.
- **Navigation area:** Enables you to view, filter, and select from a list of archives and transfers. See "Archives and Transfers Navigation Area" on page 422.
- **Content area:** Displays details about the selected archive or transfer, or a summary of all archives or transfers if you selected the top-level icon in the navigation area. See "Archives and Transfers Content Area" on page 423.
- **Status area:** Displays login details. See "Archives and Transfer Status Area" on page 423.

Archives and Transfers Menu Area

The menu are displays the following toolbar button:

Button	Function
	View the contents of a selected archive or baseline using the Show Contents Dialog Box.

Archives and Transfers Navigation Area

The navigation area displays the archives and transfers that have been created from the current product. You can:

- Filter the display of archives and transfers by selecting a value from the list: **All, Archives**, **Transfer Baselines**, or **Token Transfers**.
- Select an archive or transfer to view associated details in the content area.

- Select the top-level icon: to view a summary of all archives or transfers in the content area.

Archives and Transfers Content Area

When you select an archive or transfer in the navigation area, the content area displays the following details:

Field	Description
Archive ID	The ID of the archive or transfer.
Description	The description of the archive or transfer.
Product	The ID of the product from which the archive was created or the baseline was transferred.
Baseline	The ID of the baseline that was archived or transferred.
Tape Label	The "sticky label" that is attached to the outside of a tape.
Volume ID	The ID that is physically written on the volume.
Created	The date when the archive was created or the baseline was transferred.
User	The user ID who created the archive or transferred the baseline.
Archive Type	The archive type, Token or Transfer.

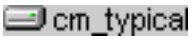
When you select the top-level icon: in the navigation area, you can:

- View a summary of all archives or baselines with a subset of the details explained above.
- Print the list of archives or transfers by clicking .
- Save the list of archives or transfers as comma-separated values by clicking .
- Sort the archives or transfers by clicking the column headings. The sort order is indicated by or .
- Filter the list of archives or transfers by clicking the **Filter** link.

Archives and Transfer Status Area

The status area contains the following icons:

Icon	Description
DMSYS	User ID of the currently logged in user.
Preferences	Click this link to display the Set User Preferences dialog box.
	Name of the current product. You can click the icon to change the current product. **NOTE** When you log in to the Administration Console under your user, the current product is the last one you have previously set under that user.

Icon	Description
	Name of the current base database.
	Authenticate with a remote host or change a remote host password.

Viewing Archives and Transfers

How to View Archive and Transfer Contents

Purpose Follow this procedure to view the items contained in a selected archive or transfer baseline. You can filter items by offline or online archive status.

To view the contents of an archive or transfer:

1. From the Archives and Transfers main window, select an archive or transfer baseline.
2. Click the Show Contents button. The "Viewing Archives and Transfers" on page 421 appears.
3. On the **General** tab, you can view details of the archive or transfer.
4. Click the **Online** or **Offline** tab to view the items by status:
   - **Offline:** The items have been deleted from the item library.
   - **Online**: The items are still stored in the item library.

Part 7

OpenText PulseUno

PulseUno Installation Files 427

PulseUno: Mapping Plugin Severity 431

Chapter 33

PulseUno Installation Files

Common Tools Directory 428

PulseUno Data Directory 428

Common Tools Directory

The Common Tools directories are:

- **Windows:** `%Program Files%\OpenText\common\`
- **UNIX:** `$DM_ROOT/../common/`

The latest version of Dimensions CM includes Tomcat 9.0 and Java 11.

Tomcat webapps Directory

The Tomcat webapps directory contains:

- The `pulse.war` web application
- The `pulse\` subdirectory containing the expanded web application.

The directories are:

- **Windows:** `%Program Files%\OpenText\common\tomcat\<tomcat-version>\webapps\`
- **UNIX:** `$DM_ROOT/../common/tomcat/<tomcat-version>/webapps/`

Tomcat SSO Configuration Directory

The Tomcat SSO configuration directory contains the `gatekeeper-services-config.xml` configuration file. Configuration of the SSO Gatekeeper is required by PulseUno for customers who are using it, however this is handled by the Common Tools installer and you do not need to change it.

The directories are:

- **Windows:** `%Program Files%\OpenText\common\tomcat\<tomcat-version>\alfssogatekeeper\conf\`
- **UNIX:** `$DM_ROOT/../common/tomcat/<tomcat-version>/alfssogatekeeper/conf/`

PulseUno Data Directory

The PulseUno data directory contains program data for the web application and includes the following subdirectories:

- `conf/`: PulseUno server configuration
- `_logs/`: PulseUno server logs
- `_caches/`: application caches
- `_secrets/`: secrets created in PulseUno

The directories are:

- **Windows:** `%ProgramData%\OpenText\Dimensions CM\Pulse\`

  On Windows, `%ProgramData%` is typically `C:\ProgramData` and is a hidden folder.

- **UNIX:** `$DM_ROOT/pulse_data/`

The PulseUno web application running in the Tomcat `webapps` directory finds the data directory by consulting the `dataDir` context parameter in this configuration file:

`pulse\WEB-INF\web.xml`

This is the only parameter in `web.xml` that you need to edit.

Configuration Settings

You can edit the configuration settings stored in the `conf/` subdirectory:

`startup.properties`	Database connection details (required for startup). Example:
	`authentication.dcm.server=cm-server.example.com`
	`database.type=oracle`
	`database.jdbcurl=jdbc:oracle:thin:@//db-server.example.com:1521/DIM14`
	`database.username=pulse`
	`database.password=...`
	`pulse.session.timeout=<integer in minutes>`

`logback.xml`	Logging configuration file.
`autoRunOnce-!firstRun.js`	Script that runs the first time PulseUno runs. The script is deleted after it completes successfully.
`backupOnce-!firstRun.js`	Backup copy of `autoRunOnce-!firstRun.js`
`installer.properties`	Records information about the installer settings and is used only for diagnostics.

If you are using a remote database you may need to create the database user and edit the property `database.jdbcurl` (in `startup.properties`) to point to your database. For details, see the PulseUno online help.

Chapter 34

PulseUno: Mapping Plugin Severity

You can configure the severity mapping between plugin findings and PulseUno.

Checkstyle Plugin

The default mapping between the findings of the Checkstyle plugin and PulseUno is:

```
# Checkstyle = Pulse
IGNORE  = IGNORE
INFO    = INFO
WARNING = WARNING
ERROR   = ERROR
```

To override the defaults create the following file on the PulseUno server:

```
${dataDir}/conf/experts/com.serena.starlight/checkstyle/
    severity-map.properties
```

For example, to set all Checkstyle findings to ERROR:

```
# Checkstyle = Pulse
IGNORE  = ERROR
INFO    = ERROR
WARNING = ERROR
ERROR   = ERROR
```

SonarQube Plugin

The default mapping between the findings of the SonarQube plugin and PulseUno is:

```
# SonarQube = Pulse
BLOCKER     = FATAL
CRITICAL    = ERROR
MAJOR       = WARNING
MINOR       = INFO
INFO        = INFO
ERROR       = ERROR
```

To override the defaults, create the following file on the PulseUno server:

```
${dataDir}/conf/experts/com.serena.starlight/sonarqube/
    severity-map.properties
```

For example, to set all SonarQube findings to ERROR:

```
# SonarQube = Pulse
BLOCKER  = ERROR
CRITICAL = ERROR
MAJOR    = ERROR
MINOR    = ERROR
INFO     = ERROR
```

PMD Plugin

The default mapping between the findings of the PMD plugin and PulseUno is:

```
# PMD = Pulse
HIGH        = FATAL
MEDIUM_HIGH = ERROR
MEDIUM      = WARNING
MEDIUM_LOW  = INFO
LOW         = INFO
```

To override the defaults create the following file on the PulseUno server:

```
${dataDir}/conf/experts/com.serena.starlight/pmd/
    severity-map.properties
```

For example, to set all PMD findings to ERROR:

```
# PMD = Pulse
HIGH        = ERROR
MEDIUM_HIGH = ERROR
MEDIUM      = ERROR
MEDIUM_LOW  = ERROR
LOW         = ERROR
```

Part 8

Appendices

Configuring Centralized Network Authentication	435
Secure Sockets Layer Support	491
Using a Non-Default Service Name and Port Number	501
Replicator Utilities	507
Replicator Error and Warning Messages	511
Using and Configuring Library Cache Areas	537
Executing Operations on Remote Nodes	551
Schedule Job Utilities and Configuration Parameters	557
Migrating MBCS Data to UTF-8 Data	565
Logging and Tracing Facilities	569
VRS Data Caching	575

Appendix A

Configuring Centralized Network Authentication

Overview	436
Authentication Architecture	441
Dimensions Support for LDAP	451
Dimensions LDAP Examples	454
Synchronizing CM and LDAP Groups	455
Dimensions Support for PAM	459
PAM Example: Configuring Solaris to use CM with Active Directory	460
Dimensions Support for SSO	467
Dimensions Support for SSO with CAC	485

Overview

Introduction

Many organizations require centralized networked user log in authentication of distributed software clients, often across heterogeneous operating systems. This is required either to overcome the maintenance overheads or perceived security concerns of local user login authentication.

Such centralized authentication is commonly implemented in the following scenarios:

- A user account does not exist on a local machine.
- A user account does exist on a local machine, but its password is read from a centrally located repository.
- A user account does exist on a local machine, but various user attributes are located or managed centrally, for example, the user's home directory, UNIX `uid`, or UNIX `gid`.

The Dimensions® CM Server provides authentication of Dimensions CM user log ins by using one or more of the following:

- Native local authentication.
- Centralized authentication through the industry standard Pluggable Authentication Modules (PAM) mechanism on UNIX (which can be made to work against Active Directory and the Light Directory Access Protocol (LDAP)) and LDAP on both UNIX and Windows.
- Starting with Dimensions CM 2009 R1, support for clients Single Sign-On (SSO) authentication was made optionally available by editing various Dimensions CM configuration files as a post-installation activity. However, that support for SSO was limited to utilizing various files and services of an existing SSO server, for example, an SSO-enabled ® Business Manager (SBM) server installation. It was, therefore, a prerequisite of an SSO-enabled Dimensions CM 2009 R1 or 2009 R1.01 installation that an existing SSO server be installed on the network or same node as the Dimensions CM server.

  Starting with Dimensions CM 2009 R2, the installer now optionally offers you the choice of:

  - Installing a Dimensions CM SSO server along with the Dimensions CM server.
  - Continuing to use an existing SSO server such as an SSO-enabled SBM server installation.

  In both cases, the Dimensions CM installer performs most of the necessary configuring of Dimensions CM to enable it to work with SSO. However, to do this, the installer requires you to provide certain prerequisite details, some of which require advanced administrator knowledge. For details, see "Dimensions Support for SSO" on page 467.
- **Dimensions CM 2009 R2 and later:** The installer optionally offers the further choice of installing, along with the SSO and Dimensions CM server, smart card reader configuration software. The only supported smart card reader is the Common Access Card (CAC), a United States Department of Defense (DoD) smart card issued as standard identification for logging in to DoD hosted software.

  The Dimensions CM installer performs most of the necessary configuring of Dimensions CM to enable it to work with SSO plus CAC, with the exception that the

trusted certificate authorities must be manually configured as a post-installation activity (see "Dimensions Support for SSO" on page 467). To do this, the installer requires you to provide certain prerequisite details, some of which requires advanced administrator knowledge. For details, see "Dimensions Support for SSO" on page 467.

The installer also offers an option to add an SSO server or SSO server plus the configuring of CACs. But the installer does not let you configure CACs subsequent to the installation of a Dimensions CM server plus SSO server. You need to do this as a post-installation activity, as explained in the installation guides and "Implementing CAC Support after Installing CM Server and SSO" on page 489.

About LDAP

LDAP (Lightweight Directory Access) is an open internet protocol that, for example, email programs use to look up contact information from a server. A Lightweight Directory is a specialized database that is optimized for lookups. Unlike a traditional RDBMS, LDAP is not designed to show complex relationships between relations. LDAP excels in situations where you want anyone, anywhere to be able to do simple queries over the Internet.

To see how LDAP is supported in Dimensions CM to access authentication information from a central authentication server, see "Native, LDAP, and PAM Authentication Methods" on page 441and then proceed to "Dimensions Support for LDAP" on page 451.

About PAM

PAM (Pluggable authentication modules) are a mechanism to integrate multiple low-level authentication schemes into a high-level API, which allows for programs that rely on authentication to be written independently of the underlying authentication scheme. With a PAM module it is possible to specify different authentication methods for different system applications without having to recompile these applications. PAM is also useful for implementing a particular policy for authorization.

For example, a system administrator may only allow console log ins from users stored in the local password file, but only allow users resolved from a NIS database to log in over the network. The UNIX `winbind` utility uses the authentication management and password management PAM interface to integrate Windows Server Active Directory users into a centralized UNIX system. This enables Windows Server Active Directory users to:

- Log in to a UNIX machine and be authenticated against a suitable Domain Controller.
- Change their passwords and have this change take effect directly on the Domain Controller.

PAM is configured by providing control files in various UNIX directories, as explained later.

PAM is currently supported in AIX, FreeBSD, HP-UX, Linux, Mac OS X, NetBSD and Solaris. PAM was later standardized as part of the X/Open UNIX standardization process, resulting in the X/Open Single Sign-on (XSSO) standard. As the XSSO standard differs from both the original RFC, Linux and Sun APIs, and also from most other implementations, PAM implementations do not all operate in the same manner. For this and other reasons, OpenBSD has chosen to adopt BSD Authentication, an alternative authentication framework that originated from BSD/OS

To see how PAM is supported in Dimensions CM, see "Native, LDAP, and PAM Authentication Methods" on page 441and then proceed to "Dimensions Support for PAM" on page 459.

About SSO

Single Sign-On (SSO) is a method of access control that enables a user to login once and gain access to the resources of multiple software systems without being prompted to login again. The following are examples of the use of this mechanism for an SSO-enabled Dimensions CM environment:

- By taking advantage of the SSO token that is generated by the Security Server and passed with ALF events by an SSO-enabled Dimensions CM, a triggered orchestration can invoke Dimensions CM or Solutions Business Manager (SBM) web services without having to explicitly specify the user/password. These web services use the same credentials as the user whose action triggered the ALF event.
- When using SBM as a request provider, Dimensions CM can invoke SBM web services using the SSO token with the same credentials as the originally logged-in user.

If you run a second instance of Dimensions CM, or the desktop and web clients, you are authenticated in both. You can also use SSO-enabled tools such as SBM without additional authentication.

SSO includes the following components:

- **SSO Gatekeeper**

  The Gatekeeper is a servlet filter that rejects incoming messages that lack proper user credentials. The Gatekeeper component acts an agent that sits in front of applications, such as Dimensions CM and the SBM Application Engine, allowing requests to pass through if they have the proper authentication, or rejecting them if they do not.

- **Security Server Identity Provider (IDP)**

  The Security Server Identity Provider presents a login page that asks the user for their credentials. Every login request comes through the Security Server to challenge the user for credentials. The Security Server also receives requests for security tokens and issues them if the requesting user is properly authenticated. The Security Server generates and signs security tokens (also known as SAML tokens) with its own private key for authenticated users. There are two entities in the Security Server:

  - **Identity Attribute Service**

    The Identity Attribute Service (IDAS) is an abstraction layer that contacts the authentication source (in conjunction with the Context Provider) to authenticate a user's credentials and return verification to the Security Server. The IDAS allows the Security Server to operate with identity stores on a high level, so that the Security Server does not have to know a particular LDAP directory and its structure or the SBM Application Engine and its structure.

  - **Context Provider**

    The Context Provider is an adapter to an authentication source such as an LDAP directory. The authentication source could be LDAP or SBM internal passwords in the SBM Application Engine database. SBM supplies LDAP and SBM Application Engine context providers as part of the installation.

  IMPORTANT!

  In CM 14.2.0.1 the SSO Login Application (Federation Server) was merged with the SSO Security Token Service (STS) into the Security Server. The `ALFSSOLogin` and `TokenService` directories were merged and replaced with a new `idp` directory on the SSO server.

If you are upgrading from a version prior to CM 14.2.0.1 and have created custom SSO integrations, you must review all URLs and calls to ensure that they use the new directory names. For example, if your existing integrations call the pre-14.2.0.1 Security Token Service (STS), you must ensure that the new `idp` directory is used (instead of `ALFSSOLogin` or `TokenService`). The endpoints of the SSO services in your custom integrations must be changed accordingly. The relative URLs are the same and the new login application entry point is:

```
http(s)://host[:port]/idp/login
```

For the STS the entry point is:

```
http(s)://host[:port]/idp/services/Trust
```

- **Identity Store**

  The Identity Store is the actual LDAP directory or database that holds the user identity (ID and password).

NOTE IDP contains JAVA API and code samples that illustrate how to implement various authenticators. Please see the following files:

- Windows: `<Tomcat-Root>\webapps\idp\samples`
- UNIX: `<Tomcat-Root>/webapps/idp/samples`

About CAC

Dimensions CM can be configured to authenticate users through a Common Access Card (CAC) that utilize the Single Sign-On (SSO) framework. A CAC is a Java Smart Card (issued by a US Department of Defense certificate authority) that is a secure token carrier containing X.509 v3 certificates (with software from middleware enabling an operating system to interface with the card through a hardware reader).

CAC authentication is a secure and reliable authentication method that enables users who have a current CAC (containing valid certificates and identity information) to gain access to a CAC-enabled Dimensions CM system once the proper PIN is provided.

The following provide a high-level overview of the Dimensions CM CAC authentication process from beginning to end:

- Extracting and Submitting Authentication Materials

  User authentication materials are extracted from the CAC when the user opens a Dimensions CM enabled client while the card is inserted in the card reader. Once the user provides a valid PIN and selects the appropriate identification certificate, the X.509 certificate information is extracted from the card and sent to the SSO server for validation within the login request.

  As part of the SSO token generation process, the gatekeeper redirects the incoming login request to the Login Application since the request does not contain a valid SSO token. Once the Login Application receives the request, it passes the authentication materials into the Security Server (IDP) through a WS-Trust Request Security Token request. Upon receiving the certificate, the IDP extracts the certificate in order to validate it. The security of this call is ensured by the SSL tunnel established by the HTTPS protocol, which requires a client certificate.

- Validating the Authentication Materials

  Prior to authenticating the user against the Identity Store, the IDP validates the certificate against one or more Certificate Revocation Lists (CRLs), a separate identity

source, or a combination of CRL and identity source. A CRL is a Certificate Authority (CA)-managed list of certificates that have been revoked or are no longer valid. If certificate validation is not performed against a CRL, the certificate can be validated by direct authentication against an existing LDAP directory or another identity system. You can choose to validate certificates against one or more sources.

Certificate validation techniques are available through extension points (also known as exits) in the IDP. These validation techniques are only required when authentication is presented through the X.509 certificate.

- Identity Mapping

  Once the authentication materials have been validated, an optional identity transformer extension is called to transform the current user ID to an ID contained in the Identity Store (in the event that the IDs are not identical).

  For example, the IDP validates the X.509 certificate and passes the certificate information to the Identity Store. The certificate contains a common name (CN) in the form CN=LASTNAME.FIRSTNAME.MIDDLE.10DIGITNUMBER. If the user ID does not match the common name, then authentication fails unless the IDs are mapped. Therefore, the IDP provides an exit in which you can establish your own ID mappings.

- Identity Resolution

  Once the authentication materials are validated, the IDP goes through the identity mapping exit, then through the IDAS and the Context Provider to talk to the Dimensions CM Identity Store to resolve the user identity and generate SSO tokens. Once the token is received, the user is logged in to Dimensions CM.

- Logging Out

  SSO maintains the user's session as long as the browser is open. If the browser is closed, the entire authentication sequence must be repeated in order to access Dimensions CM again. If the user exits the browser, they are returned to the SSO login page. Depending on the settings in CAC client software (ActivClient), when the card is removed from the reader, the user is either:

  - logged out of the Dimensions CM (requiring a subsequent log in), or the machine is locked with the login credentials preserved in the browser session, or
  - the user is logged off the operating system, terminating the SBM browser session.

To see how CAC is supported in Dimensions CM, see: "Web Tools Authentication with SSO/CAC" on page 449 and "Desktop Client, Eclipse, and Visual Studio Authentication with SSO/CAC" on page 450; and then proceed to "Dimensions Support for SSO with CAC" on page 485.

Password Security Operating System Limitations

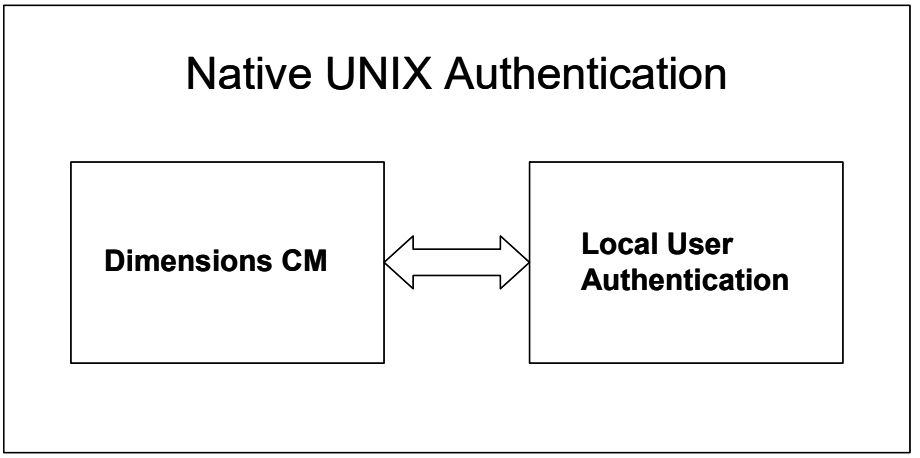
Dimensions CM login password security is ultimately dependent on the operating system password security, so this should always be borne in mind. For example:

- Most vanilla UNIX operating systems consider only the first eight characters of a password to be significant. Therefore, for a particular Dimensions CM server, client or agent login user identity, if the password was, for example, `qwerty99_zap`, only `qwerty99` is significant. In this case, if you enter `qwerty99fred` or `qwerty99jill`, your password is still accepted.
- Unless the operating system is set up to lock out users after a specific number of failed login attempts, you can always gain entry to Dimensions CM by eventually entering the correct operating system password after numerous wrong attempts.

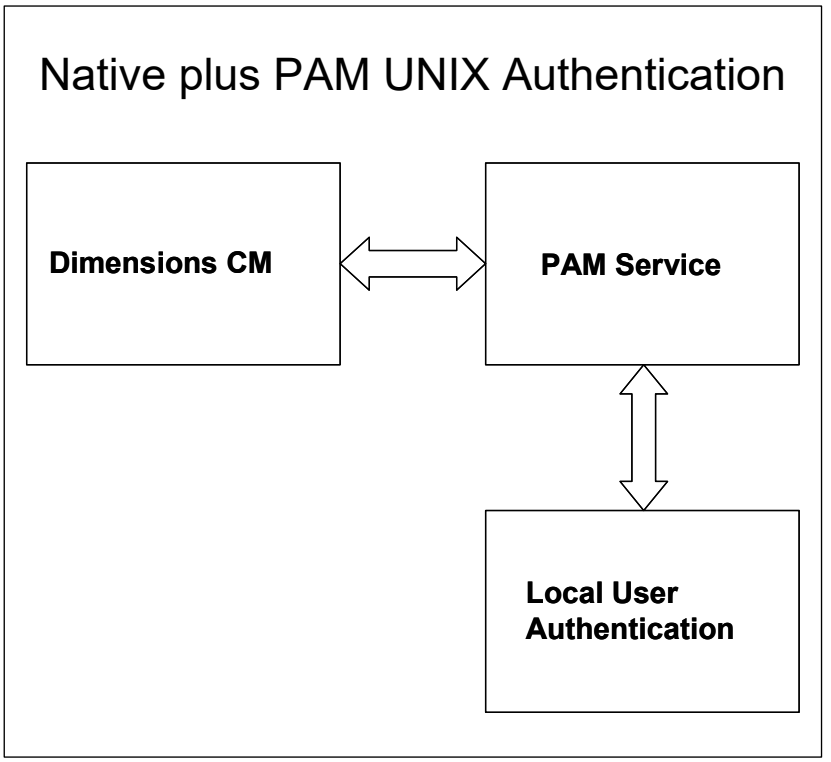
Authentication Architecture

Native, LDAP, and PAM Authentication Methods

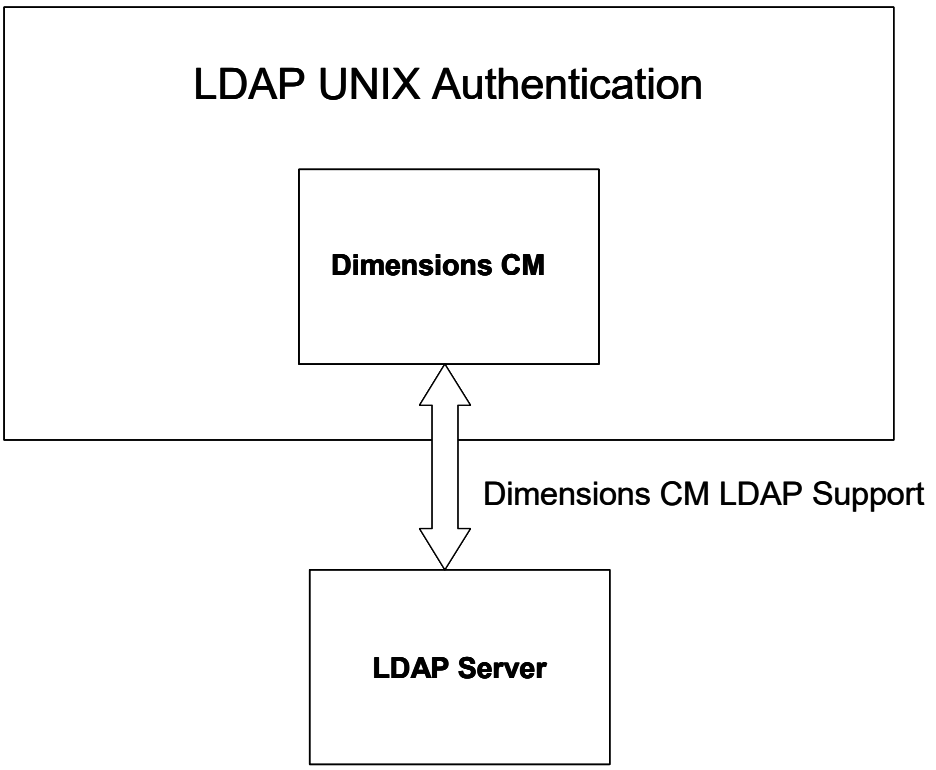
The native, LDAP, and PAM authentication methods supported are as follow:



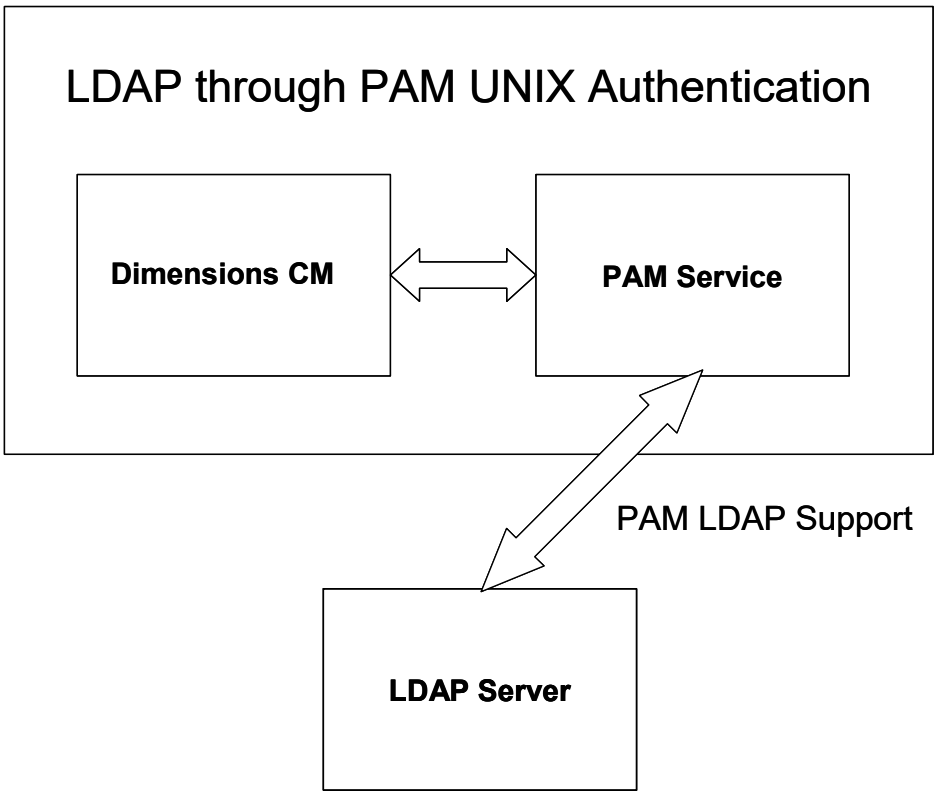
Native Authentication for a Dimensions CM for UNIX Server



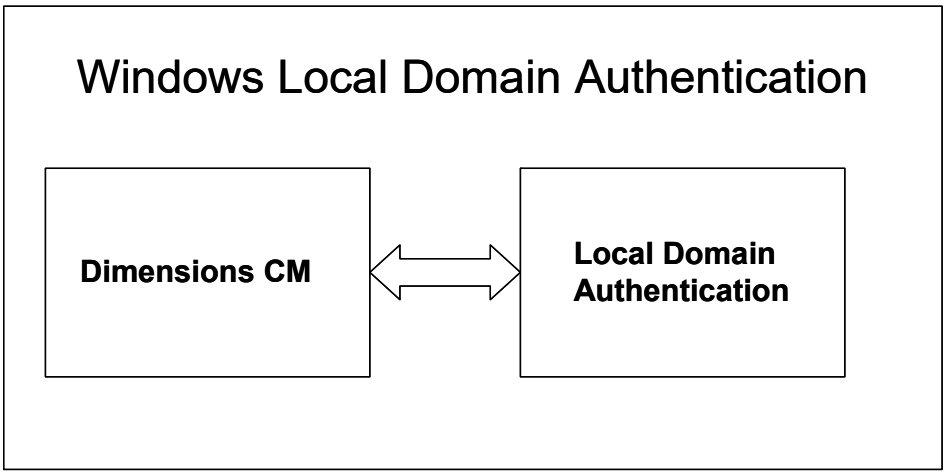
Native Authentication through PAM for a Dimensions CM for UNIX Server



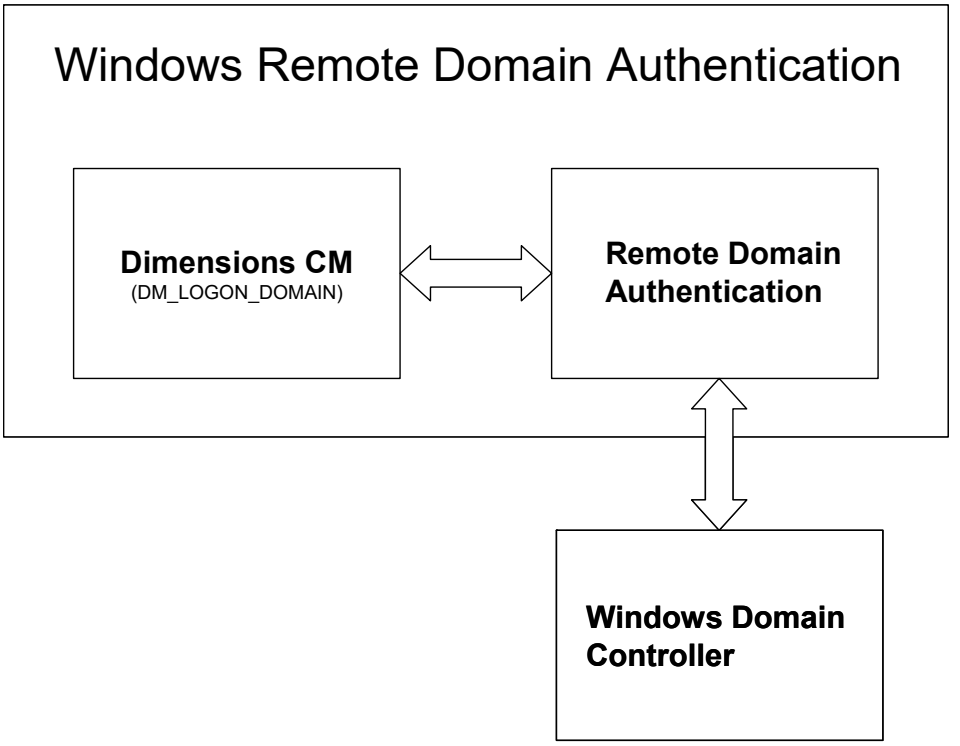
LDAP Authentication for a Dimensions CM for UNIX Server



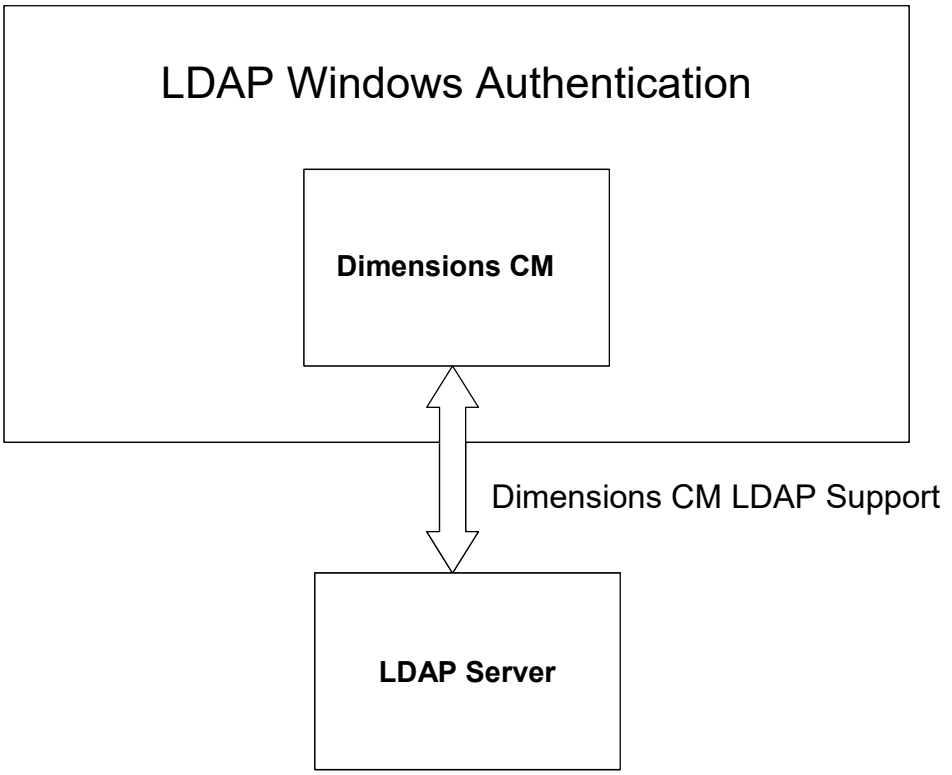
PAM LDAP Authentication for a Dimensions CM for UNIX Server



Local Domain Authentication for a Dimensions CM for Windows Server



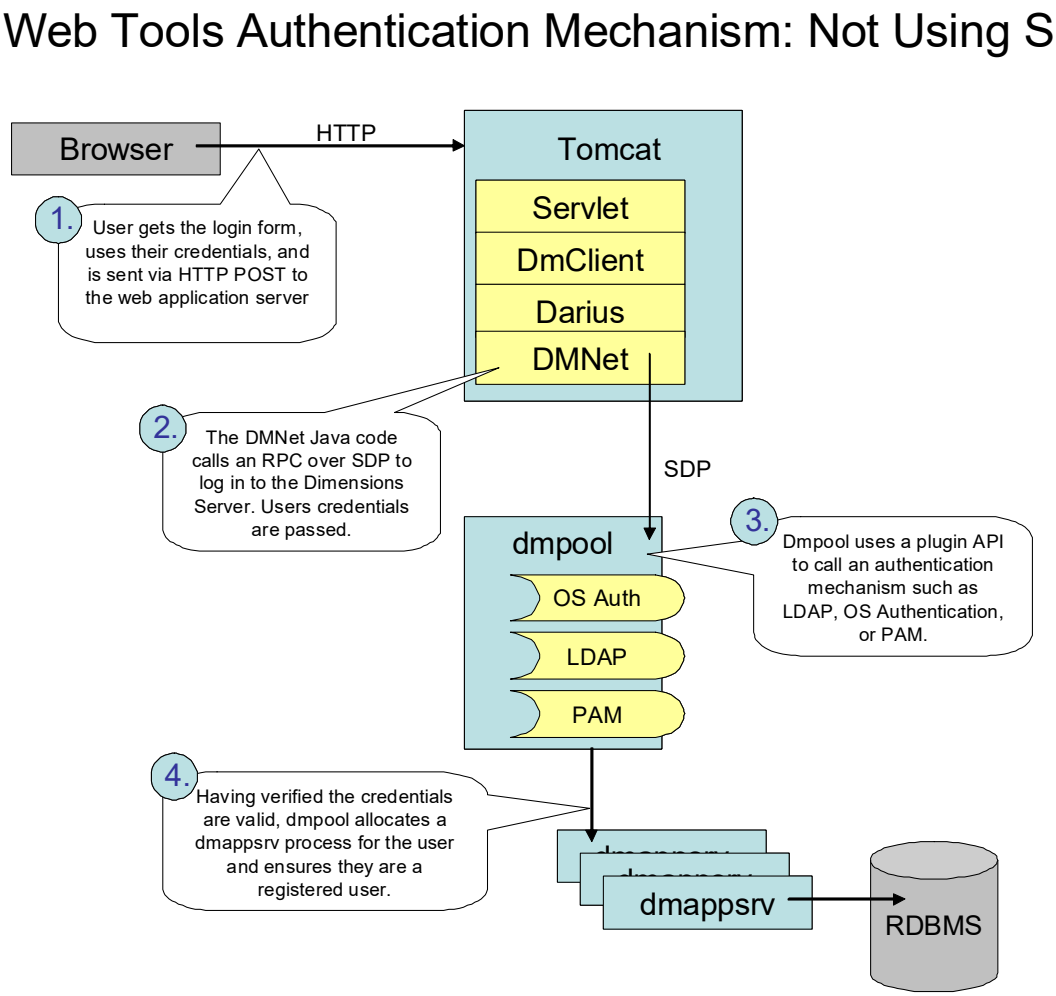
Remote Domain Authentication for a Dimensions CM for Windows Server



LDAP Authentication for a Dimensions CM for Windows Server

Web Tools Authentication without SSO

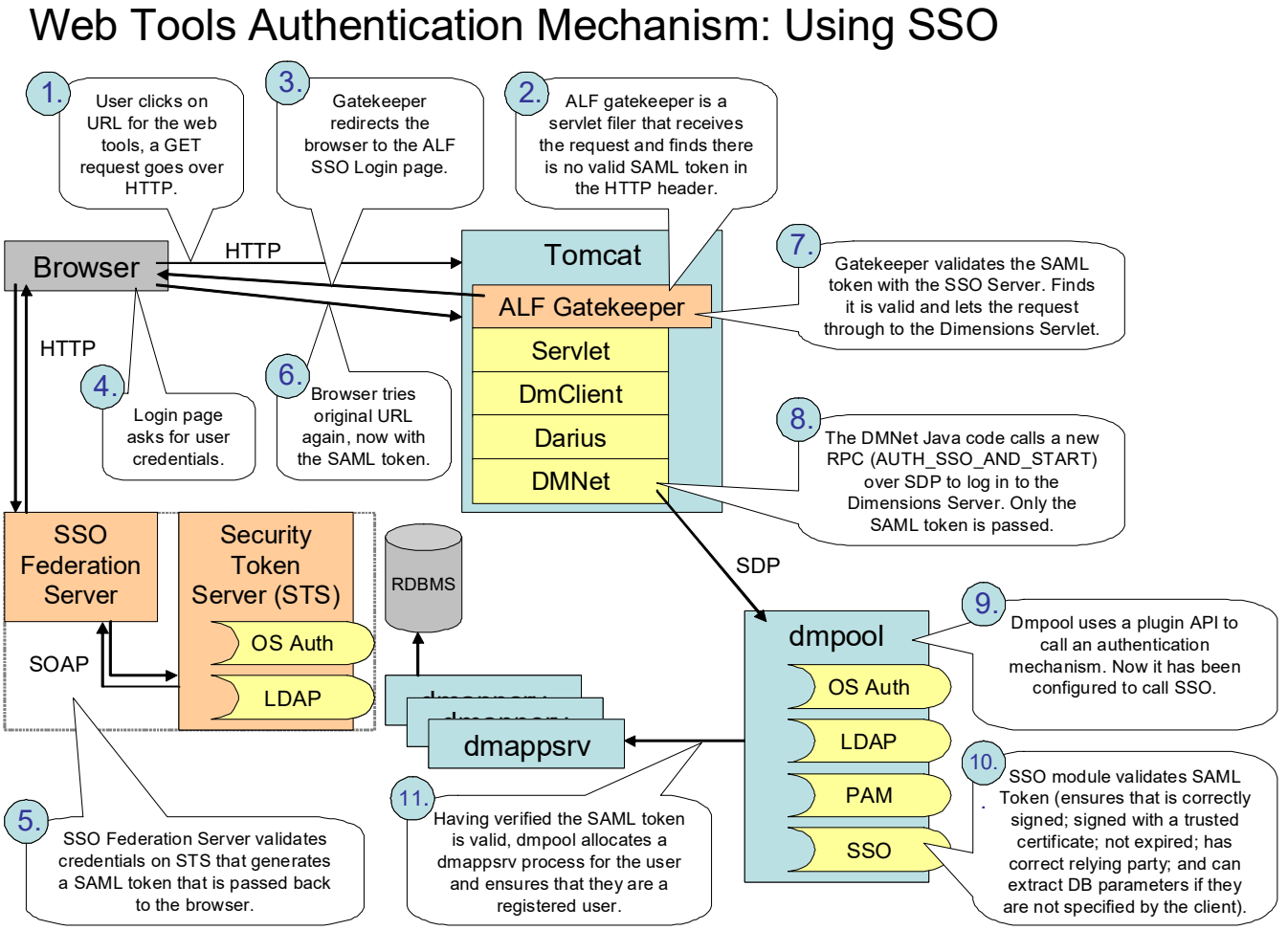
The low-level Dimensions CM web tools authentication mechanism without Single Sign-On (SSO) is as follows:



Web Tools Authentication Mechanism: Not Using SSO

Web Tools Authentication with SSO

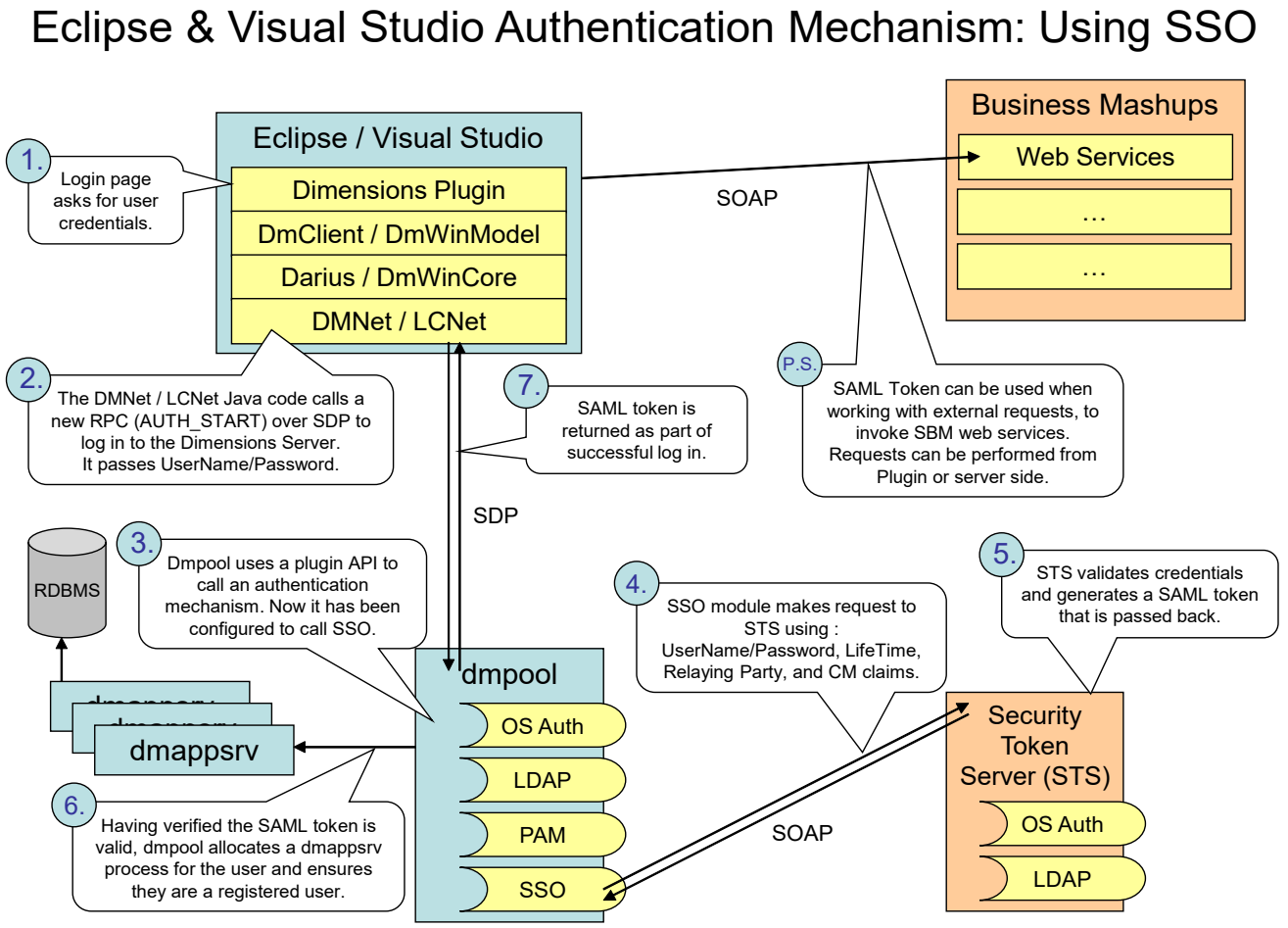
The low-level Dimensions CM web tools authentication mechanism with Single Sign-On (SSO) is as follows:



Web Tools Authentication Mechanism: Using SSO

Eclipse and Visual Studio Authentication with SSO

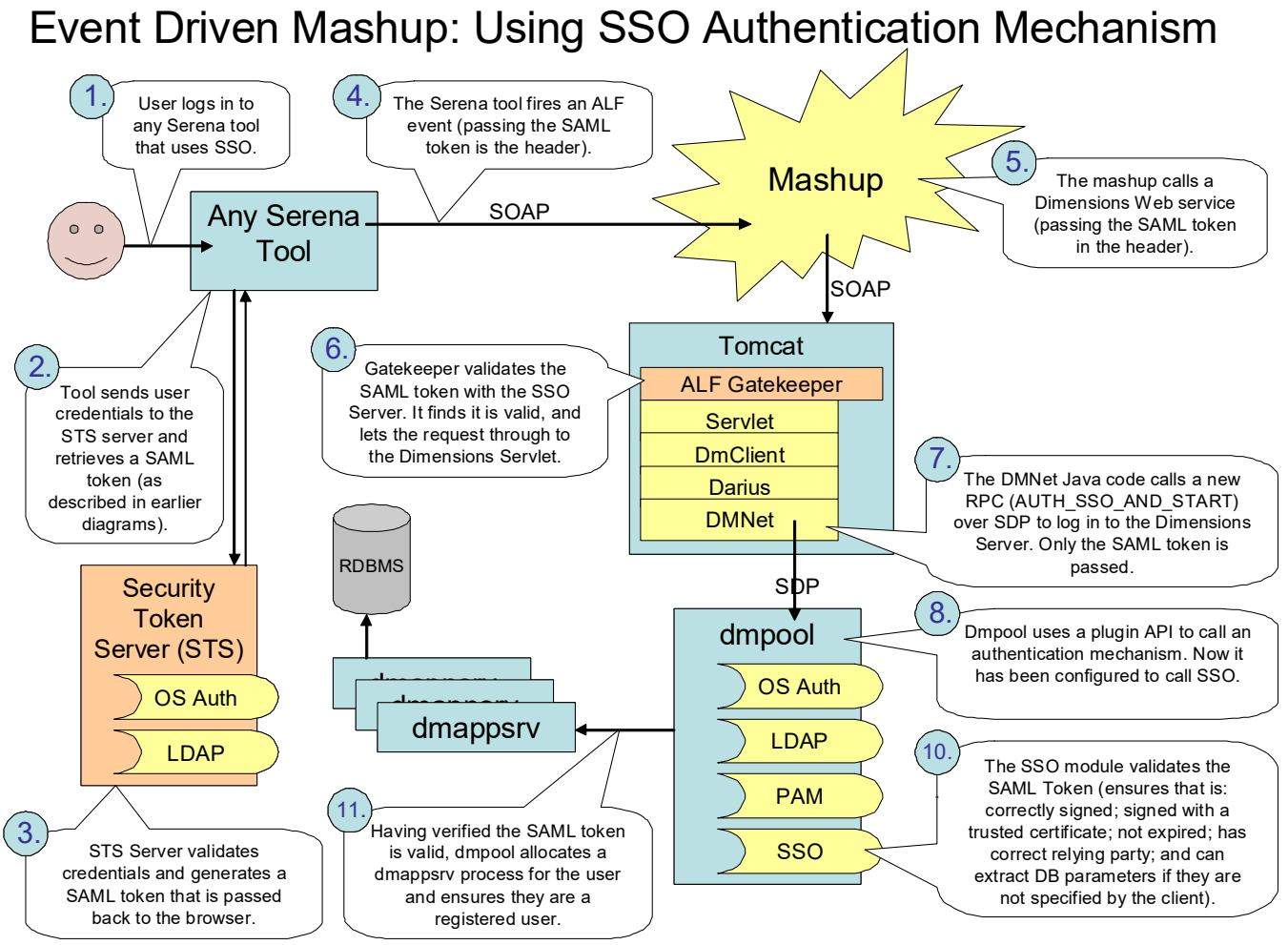
The low-level Dimensions CM Eclipse and Visual Studio rich integration authentication mechanism with Single Sign-On (SSO) is as follows. Where Eclipse and Visual Studio components differ, they are shown as `<Eclipse_component> / <Visual_Studio_component>` in the diagram below, for example, `DmClient / DmWimModel`:



Eclipse and Visual Studio Rich Integration Authentication Mechanism: Using SSO

Event Driven Authentication with SSO

The low-level Dimensions CM event driven process app authentication mechanism with Single Sign-On (SSO) is as follows:

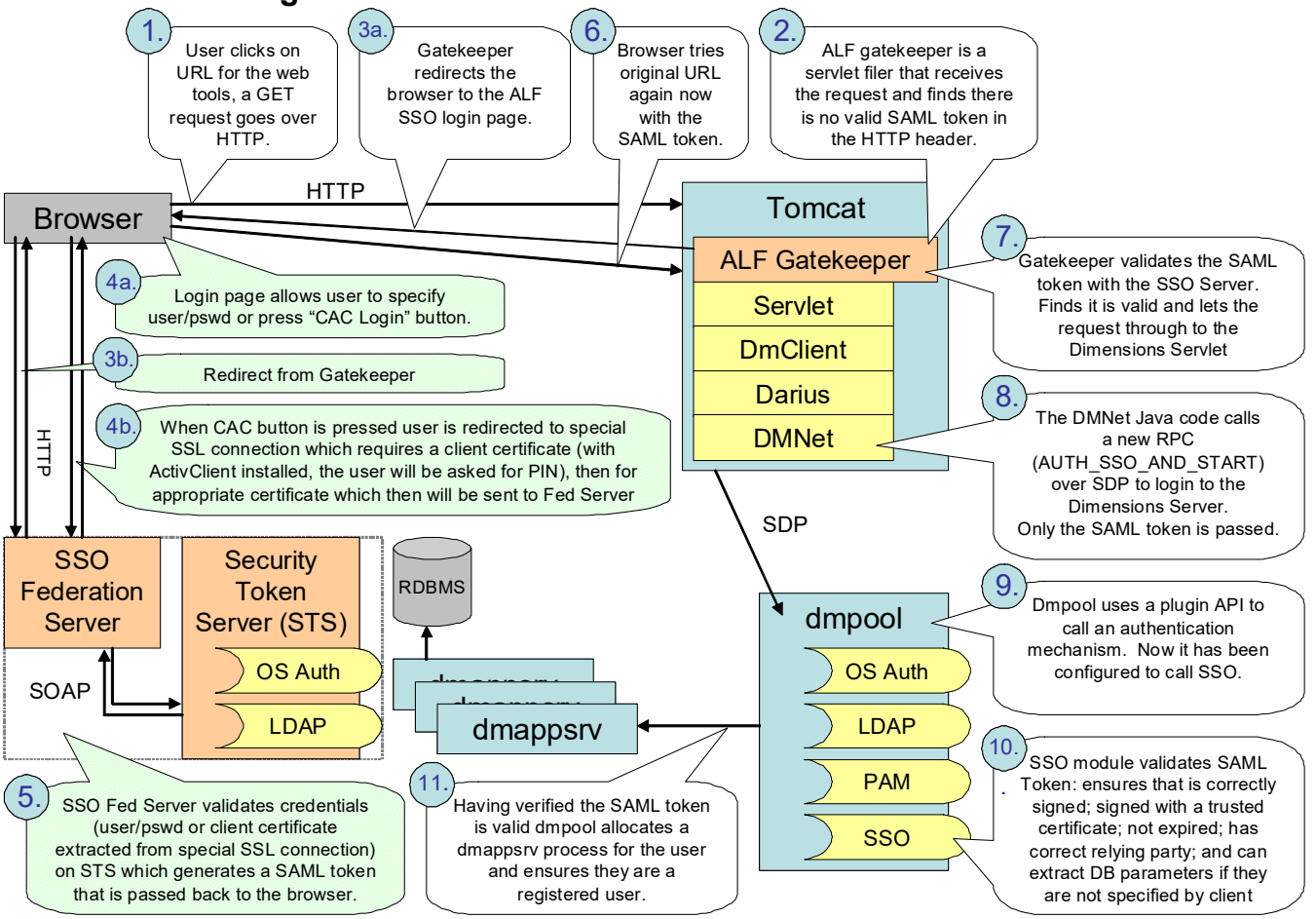


Event Driven Process App Authentication Mechanism: Using SSO

Web Tools Authentication with SSO/CAC

The low-level Dimensions CM web tools authentication mechanism with Single Sign-On (SSO) plus Common Access Card (CAC) is as follows:

Web Tools Using CAC within SSO Authentication Mechanism

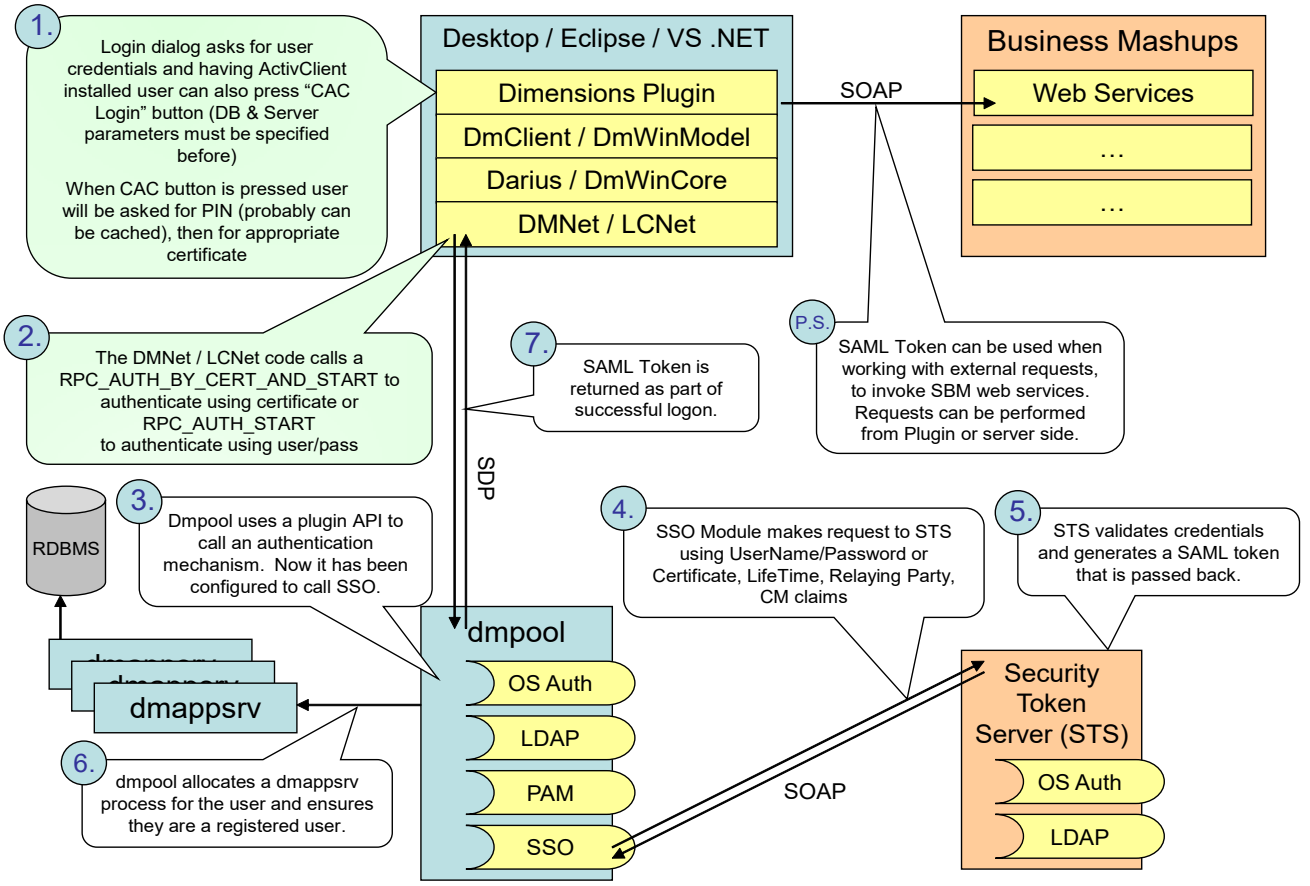


Web Tools Authentication Mechanism: Using SSO Plus CAC

Desktop Client, Eclipse, and Visual Studio Authentication with SSO/CAC

The low-level Dimensions CM desktop, Eclipse and Visual Studio rich integration authentication mechanism with Single Sign-On (SSO) is as follows (where desktop, Eclipse and Visual Studio components differ, they are shown as `<Desktop component>/ <Eclipse_component> / <Visual_Studio_component>` in the diagram below:

Desktop Client, Eclipse, and Visual Studio Using CAC within SSO Authentication Mechanism



Desktop client, Eclipse, and Visual Studio Rich Integration Authentication Mechanism: Using SSO Plus CAC

Dimensions Support for LDAP

Introduction

A Dimensions CM client tool can authenticate against a LDAP server for an operating system username and password when connecting to a Dimensions CM server. To enable LDAP authentication, the following Dimensions CM configuration variables in the file `dm.cfg` on the Dimensions CM server need to be set as the following example shows:

```
DM_AUTH_TYPE_DBS        LDAP
LDAP_HOST               ldap.mycompany.com
LDAP_PORT               389
LDAP_ORG                My Company
LDAP_ORG_UNIT           _ABViews_
```

IMPORTANT!

- Dimensions CM works with LDAP servers that support version 3 of the LDAP protocol.
- In a pure LDAP environment, Dimensions CM commands requiring authentication, such as AUTH, CA, and DPL, require a local administrator operating system account.

In addition to the standard LDAP protocol, the Dimensions CM product also supports:

- LDAPS (Secure Sockets Layer encrypted protocol typically run over TCP/IP port number 636). This is a popular LDAP configuration, but is not described in any IETF standard.
- LDAP with TLS extensions (TLS encrypted protocol typically run over the standard LDAP TCP/IP port number 389). This internet standard protocol enables secure communication to an LDAP server without using an additional TCP/IP port. It is described in RFC 2830 (https://www.ietf.org/rfc/rfc2830.txt).

For details on configuring Dimensions CM for secure LDAP, please see the next section.

Configuring CM for Secure LDAP

If you plan to use LDAPS (LDAP using Secure Sockets Layer) or LDAP with TLS extensions, proceed as follows:

1. Ensure that your LDAP server is already configured for secure connections (consult the vendor documentation for your specific LDAP server for details).
2. Set the `LDAP_SECURE` variable in `dm.cfg` to indicate which secure protocol you want to use (TLS or SSL).
3. Install a copy of your LDAP server's client certificate on the Dimensions CM server. This enables the LDAP server to recognize and trust the Dimensions CM server—without the presence of the certificate the Dimensions CM server cannot connect. The vendor documentation for your LDAP server should contain details on how to obtain the certificate. Details on how Dimensions CM enables access to LDAP servers is given below.

Dimensions CM makes use of the OpenLDAP software library to access LDAP servers, this library uses a file called `ldap.conf` to indicate where the certificate is installed. For details, see the OpenLDAP documentation.

The ldap.conf file is located in the Dimensions CM distribution as follows:

- **Windows:** OpenLDAP by default expects the configuration file to be called ldap.conf and exist in the folder C:\OpenLDAP\sysconf.

  If you want to change that location, then set the environment variable LDAPCONF in the environment of the account running the Dimensions CM Network Listener service. This variable must be set to the location (including name) of the ldap.conf file, for example:

  ```
  C:\Program Files\OpenText\Dimensions <version>\CM\dfs\ldap.conf
  ```

  After setting the environment variable, do not forget to restart the service.

- **UNIX:** OpenLDAP by default looks in /usr/local/etc/openldap/ldap.conf, but it also looks at ldap.rc or .ldaprc in the home directory of the user who starts the Dimensions network listener (typically root).

In the ldap.conf file Dimensions CM only requires the TLS_CACERT variable to be set to the filesystem location of the certificate, an example might be:

```
TLS_CACERT C:\OpenLDAP\sysconf\cacert.pem
```

LDAP Authentication and Tertiary Node Access

LDAP authentication is only used to validate credentials of users logging in to the Dimensions CM server.

For tertiary node operations such as remote file access and build, a valid set of operating system user credentials for the remote node must be provided. These credentials are required because the filesystem needs to be accessed in the context of the user. This is not possible with LDAP credentials.

Dimensions LDAP Configuration Variables

The first set of variables to consider are:

- DM_AUTH_TYPE_DBS <auth_type>

  This is a local instruction to the server or tertiary node, for example, a server or tertiary node does LDAP authentication only if it is locally switched on, on that node. When DM_AUTH_TYPE_DBS is set to LDAP, Dimensions CM uses LDAP for authentication instead of operating system or PAM authentication when a user logs in to a Dimensions CM server through a Dimensions CM client. Operating system or PAM authentication are still used for all tertiary node operations, such as building and getting files to a remote network node.

- LDAP_HOST <host_name>

  This is the host name of the LDAP server, for example, ldap.mycompany.com.

- LDAP_PORT <port_no>

  The TCP/IP port number that the LDAP server is listening on. Most sites set this to 389.

- LDAP_SEARCH 1

  This instructs Dimensions CM to search for the user instead of binding to the user.

- LDAP_SECURE <TLS|SSL>

  This variable indicates if secure LDAP is required and enables the security protocol to be chosen. If not present a normal non-secure LDAP connection is made.

  The value can be one of the following:

  - SSL – This enforces the use of LDAPS (secure SSL LDAP connection).
  - TLS – This enforces the use of LDAP with TLS extensions.

- LDAP_NO_REFERRALS 1

  This variable instructs Dimensions CM to ignore any referral requests resulting from an LDAP directory search.

  By default, when Dimensions CM searches the LDAP directory, if it comes across a referral request, then the search fails and, therefore, the Dimensions CM log in fails. Setting this variable to 1 ensures that the search and log in continues regardless of referral requests.

The second set of variables to consider affect the LDAP Distinguished Name (DN) that you pass to the ldap_bind call. You start by adding cn=userid to the DN string then you evaluate the following variables adding as you go:

- LDAP_ORG <organization_segment>

  If this optional variable is set, you use its value as the organization o= segment of the DN.

- LDAP_ORG_UNIT <organization_unit>

  If this optional variable is set you use its value as the organization unit ou= segment of the DN.

- LDAP_BASE_NAME <base_name>

  If this optional variable is set, its value is added to the end of the DN.

- LDAP_USER_RDN cn

  This is used as the attribute type for the user-id for the search filter. If not specified, cn is used.

- LDAP_BIND_USER cn=fred,cn=Users,dc=staldev,dc=com

  This is used to bind to the LDAP server for executing searches. If not specified, an anonymous connect is made to the LDAP server.

  The password for this user must be registered with Dimensions CM using the following:

  - **Windows:** %DM_ROOT%\prog\dmpasswd.exe
  - **UNIX:** $DM_ROOT/prog/dmpasswd

  as follows:

  ```
  dmpasswd cn=fred,cn=Users,dc=staldev,dc=com -add -pwd <password>
  ```

Dimensions LDAP Examples

Example 1

```
LDAP_HOST        server1
LDAP_PORT        389
LDAP_ORG         serena
LDAP_BASE_NAME   sn=Raymond,dc=serena
```

If user `fred` logs in, Dimensions CM tries to bind to the LDAP host `server1` on port `389` with a `DN` of

```
cn=fred,o=serena,sn=Raymond,dc=serena
```

`fred` does not require an operating system login account on the Dimensions CM server or LDAP server, the authentication is completely performed by the `ldap_bind`.

Example 2

```
LDAP_HOST       devlogin
LDAP_PORT       389
LDAP_ORG        serena
LDAP_ORG_UNIT   dev
```

If user `jim` logs in, Dimensions CM tires to bind to the LDAP server `devlogin` on port `389` with a `DN` of

```
cn=jim,o=serena,ou=dev
```

`jim` does not require an operating system login account on the Dimensions CM server or LDAP server, the authentication is completely performed by the `ldap_bind`.

Example 3

An example of configuring Dimensions CM to search LDAP is as follows:

```
LDAP_HOST       server2
LDAP_PORT       389
LDAP_SEARCH     1
LDAP_BASE_NAME  ou=Users,dc=serena,dc=com
LDAP_BIND_USER  cn=dmsys,ou=Users,dc=serena,dc=com
LDAP_USER_RDN   SAMAccountName
```

In the above configuration if user `bob` logs in, Dimensions CM binds to the LDAP server using the `DN` specified in

```
LDAP_BIND_USER (cn=dmsys,ou=Users,dc=serena,dc=com)
```

using the password set through the `dmpasswd` utility.

Dimensions CM then searches the directory to find a `DN` whose `SAMAccountName` attribute (as specified in `LDAP_USER_RDN`) matches the value `bob` (the name of the user logging in). If found, Dimensions CM binds against that `DN` using the password as supplied by the user logging in to Dimensions CM. If no matches or more than one match is found, connection fails.

Example 4

This is an example of configuring Dimensions CM to use LDAPS (secure SSL LDAP connection):

```
LDAP_HOST       safeserver.serena.com
LDAP_PORT       636
LDAP_SECURE     SSL
LDAP_BASE_NAME  ou=Users,dc=serena,dc=com
```

In the above configuration, the LDAP connection is made using Secure Socket Layer LDAP (the LDAPS protocol) to port 636 of the machine `safeserver.serena.com`. Note that in this example the fully qualified domain name for the server was specified because the name *must* match the name specified in the certificate.

For this to be successful you must have first copied your server's client certificate (public key) onto your Dimensions server machine (see "Configuring CM for Secure LDAP" on page 451).

Example 5

```
DM_AUTH_TYPE_DBS  LDAP
LDAP_SEARCH       1
LDAP_HOST         ord-dc.acme.com
LDAP_PORT         389
LDAP_USER_RDN     sAMAccountName
LDAP_BIND_USER    cn=scmadmin,ou=New York Lab,ou=Acme Service
    Accounts,dc=acme,dc=com
LDAP_BASE_NAME    dc=acme,dc=com
```

Synchronizing CM and LDAP Groups

Introduction

When you authenticate against an LDAP server you can optionally synchronize Dimensions CM groups and LDAP groups. Synchronizing groups has these advantages:

- Group members are automatically synchronized from LDAP to CM.
- Groups are automatically created in CM if they do not exist.
- Users are automatically added to CM groups. Users are only removed from groups they were added by the synchronization.
- Synchronization frequency can be set by a CM administrator.
- LDAP group attributes are configurable in CM.
- A single server process can synchronize multiple CM base databases.
- Synchronization runs as a background task to minimize performance when logging in.

Synchronization is run by the *task server* (`dmtasksrv.exe`), a Dimensions CM server component started by the pool manager. The task server reads the configuration file (`DM_ROOT/dfs/dmtasksrv_config.dat`) and runs the tasks for each base database.

Configuring Synchronization

1 To configure the synchronization edit this configuration file:

`DM_ROOT/dfs/dmtasksrv_config.dat`

By default the configuration file is set up by the installer to monitor the default database in the listener configuration file: `DM_ROOT/dfs/listener.dat`

2 Add parameters to suit your environment (see below).

Global Parameters

Global parameters apply to all base databases.

Parameter	Description	Default Value
`log_dir`	Specify this parameter to: ■ Enable logging. ■ Redirect the log to a file in the specified directory.	None, logging is off.
`idle_timeout`	Specifies the period after which the task server closes unused application server connections. If you specify 0 unused connections are kept in the connection pool until the deployment server terminates.	60

Base Database-Specific Parameters

Parameter	Description	Default Value
database	Specifies a single database to be monitored.	None
database_[n]	Specifies multiple databases to be monitored where: [n] = 1, 2, 3, ... You must specify at least one database.	None
host	Specifies a single application server host and port for the database specified in the parameter database.	Local host name
host_[n]	Specifies multiple application server hosts and ports for the databases specified in the parameter database_[n].	Local host name
dmuser dmuser_[n]	Specifies registered Dimensions CM users to be used for performing tasks in the databases stipulated above. Use the dmpasswd utility to register the passwords corresponding to the Dimensions users in: DM_ROOT/dfs/registry.dat	dmsys
threads threads_[n]	Specifies the number of threads to be used to execute tasks.	20
task_thread_timeout	Specifies the interval in seconds between each synchronization.	60

Enabling Synchronization

To enable LDAPSYNC, the LDAP synchronization, add these variables to the dm.cfg configuration file on the Dimensions CM server.

Variable	Description
LDAP_HOST LDAP_PORT LDAP_BASE_NAME LDAP_SEARCH LDAPCONF LDAP_SECURE LDAP_BIND_USER LDAP_NO_REFERRALS LDAP_USER_RDN LDAP_FILTER	Used to bind to the LDAP server and to filter searches.
LDAP_GROUP_RDN	Used as the attribute type for the group ID in the search filter. If not specified, cn is used.
LDAP_GROUP_MEMBER_RDN	Used as the attribute type for the group member in the search filter. If not specified, member is used. Only used for *OpenLdap* servers.
LDAP_SEARCH_CHUNK_SIZE	Splits large LDAP search queries into chunks to limit the number of LDAP users and groups per query.

Variable	Description
LDAP_GROUP_OBJ_CLASS	Used as the attribute type for the group object class in the search filter. If not specified, groupOfNames is used. Only used for *OpenLdap* servers.
LDAP_USER_MEMBEROF_RDN	Used as the attribute type for the group member in the search filter. Only used for *Microsoft Active Directory*.
LDAP_GROUP_CREATE_OFF	Disables creation of new groups in CM.

Example 1

In the following example LDAPSYNC:

- Binds to the LDAP host server1.
- Uses the Dimensions CM user ID admin with a DN of:
  - cn=dmsys
  - o=serena
  - sn=Raymond
  - dc=serena
- Searches for users and groups. User ID is represented with the attribute cn and user group member with memberOf. Group ID is represented with the attribute sAMAccountName.

```
LDAP_HOST               server1
LDAP_ORG                serena
LDAP_BASE_NAME          sn=Raymond,dc=serena
LDAP_GROUP_RDN          sAMAccountName
# Active Directory
LDAP_USER_MEMBEROF_RDN  memberOf
```

Example 2

In the following example LDAPSYNC:

- Binds to the LDAP host server2 using the DN specified in LDAP_BIND_USER (cn=ted,ou=Users,dc=serena,dc=com) and the password set with the dmpasswd utility.
- Searches for users and groups using the filter specified in LDAP_FILTER. User ID is represented with the attribute SAMAccountName and group member with memberOf. Group ID is represented with the attribute cn.

```
LDAP_HOST               server2
LDAP_BASE_NAME          ou=Users,dc=serena,dc=com
LDAP_SEARCH             1
LDAP_BIND_USER          ted
LDAP_USER_RDN           SAMAccountName
LDAP_FILTER             (&(objectClass=user)(sAMAccountName=%s))
# Active Directory
LDAP_USER_MEMBEROF_RDN  memberOf
```

Example 3

In the following example LDAPSYNC:

- Binds to the LDAP host `server3` using the DN specified in LDAP_BIND_USER (`cn=Manager,ou=Users,dc=serena,dc=com`) and the password set through the `dmpasswd` utility.
- Searches for users and groups using the filter specified in `LDAP_FILTER`. Splits the query into chunks to search a maximum of 10 users and groups per chunk. User ID is represented with the attribute `cn`. Group is represented with the object class `groupOfNames` and ID with `cn`. Members of groups are represented with `member`.

```
LDAP_HOST                server3
LDAP_BASE_NAME           ou=Users,dc=serena,dc=com
LDAP_SEARCH              1
LDAP_SEARCH_CHUNK_SIZE   10
LDAP_BIND_USER           cn=Manager,ou=Users,dc=serena,dc=com
LDAP_FILTER              (&(objectClass=person)(cn=%s))
# OpenLdap
LDAP_GROUP_MEMBER_RDN    member
```

Dimensions Support for PAM

Introduction

IMPORTANT!

- PAM is only available on Dimensions for UNIX servers but can be made to work in conjunction with both UNIX LDAP and Windows Active Directory.
- As well as setting certain Dimensions server configuration symbols, as described in "PAM Configuration Variables and Files" on page 459, some general PAM configuration of the host machine may be needed before it can be used with a Dimensions server. See your operating system documentation for guidance.

Dimensions provides support, as described below, for the industry standard PAM mechanism, which enables programs that rely on authentication to be independent of the underlying authentication scheme.

PAM Support

The Dimensions server PAM implementation only supports password authentication. This means that more elaborate authentication schemes and configurations requiring more than password authentication fail, as the additional information required has not been requested from the user.

PAM Configuration Variables and Files

A Dimensions CM client tool can authenticate against a PAM server for operating system username and password when connecting to a Dimensions server. To enable PAM authentication, the following Dimensions configuration variable in the file `dm.cfg` on the Dimensions server needs to be set as follows:

```
DM_AUTH_TYPE_DBS        PAM
```

In certain circumstances (as described below), the following variable also needs to be set:

```
DM_PAM_SERVICE_NAME      <PAM_service_name>
```

The configuration variables and files to consider are:

- DM_AUTH_TYPE_DBS <auth_type>

  This is a local instruction to the server or tertiary node, for example, a server or tertiary node does PAM authentication only if it is locally switched on, on that node. When DM_AUTH_TYPE_DBS is set to PAM, Dimensions uses PAM for authentication instead of operating system or LDAP authentication when a user logs in to a Dimensions server through a Dimensions client. Operating system or PAM authentication is also used for all tertiary node operations, such as building and fetching files to a remote network node.

- DM_PAM_SERVICE_NAME <PAM_service_name>

  This is *only* used in those circumstances where several Dimensions CM server instances are co-located on the same UNIX host (see "Running Multiple CM Instances on a UNIX Host Machine" on page 53) and each has a different authorization scheme in PAM. The default value of `dimensions_cm` is used if no entry for this configuration variable exists in `dm.cfg`.

- PAM Configuration File

  The PAM configuration file contains details on all PAM authentications, including the default authentication scheme, which is labelled `"other"`. You can change the default (called `"other"`) or add entries specific to Dimensions CM (called "`dimensions_cm`").

  On most UNIX systems, the file is `/etc/pam.conf` and contains a default configuration to work for all PAM services. You can add specific alterations to the PAM authentication used by Dimensions CM, by adding or changing configuration keyed by "`dimensions_cm`".

  On Linux, there are many configuration files in the `/etc/pam.d` directory, one for each PAM service, with each having the name of the service. To start with, you can copy the `login` or `su` service configuration file to `dimensions_cm`.

PAM Example: Configuring Solaris to use CM with Active Directory

Introduction

This section of the manual describes a demonstration prototype implementation of PAM that took place at an internal site. It describes the steps required to configure a collection of Solaris 10 machines running Dimensions CM authenticating through PAM user log ins against Windows Active Directory.

Acknowledgment

The implementation was established by following (and modifying as appropriate) the instructions in this article:

https://blog.scottlowe.org/2006/08/15/solaris-10-and-active-directory-integration

The referenced article describes in detail all the necessary procedures and processes required to set up LDAP and Active Directory on Solaris. Reference should be made, in

the most part, to the instructions in the referenced article, with this section primarily intended to help summarise, supplement, and comment on those instructions. For comments on issues arising from the referenced article, see "Possible Issues" on page 463.

The Active Directory setup used in the implementation was: a Windows 2003 R2 Active Directory in native mode, containing a single root domain with test users created in the default users organizational unit.

Overview of CM Authentication via Active Directory

It is possible to configure the Solaris operating system to authenticate against Active Directory using LDAP. This is achieved by configuring standard PAM technology on the Solaris machine such that when authentication requests are received they are routed through PAM and LDAP to use a specified Active Directory server.

When an authentication request is logged against a PAM-enabled Dimensions CM server (see "PAM Configuration Variables and Files" on page 459), the request is routed through to Active Directory to be serviced. Any authentication errors that might occur—such as invalid user name or invalid password—are routed back to Dimensions CM and displayed to the end-user. This means that the Dimensions CM authentication request is being serviced by Active Directory.

In this scenario, Dimensions CM passes control to the PAM programming interface to perform authentication. The PAM authentication library and engine are controlled by suitably editing the system configuration file `/etc/pam.conf` on most UNIX flavors or the appropriate system configuration file in the set of such files in the directory `/etc/pam.d` on Linux.

The PAM configuration file enables a choice of various authentication schemes including third-party and customer developed schemes. This is all independent of the Dimensions CM application and you can, in fact, set different authentication schemes for different PAM-enabled applications on your system, the default being normal operating system authentication. After an authentication scheme is chosen, you can configure it through the PAM configuration file or additional associated configuration files.

To summarize:

1. Dimensions CM can be switched to use PAM authentication.
2. PAM authentication can then be configured to use LDAP (which is supplied native to UNIX machines).
3. LDAP can then, in turn, be configured to work with an existing Active Directory installation instead of an LDAP server.

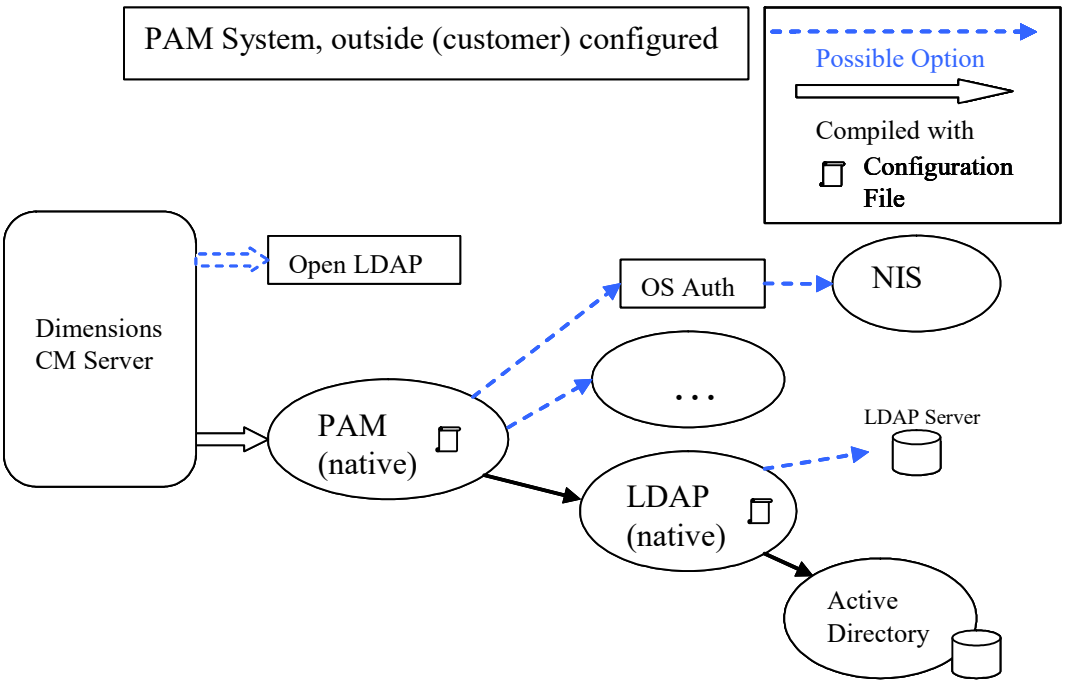


Figure A-6. PAM Configuration for Active Directory Through LDAP

Implementing the Scheme

The following steps need to be performed to implement the scheme:

1 Preparing Active Directory

The following steps need to be sequentially preformed on the Active Directory to allow it to work with the UNIX servers.

a Once Only

Make sure Active Directory has the extensions for UNIX and create an account to link things together. In the referenced online article, this is documented in *"Preparing Active Directory (One Time)"*.

b For Each User

Supply each Active Directory account with associated UNIX account attributes such as "UID", "login Shell", and "Home Directory". This is done through a table installed earlier. In the referenced online article, this is documented in *"Preparing Active Directory (Each User)"*.

c For Each Solaris Server

Creating a user for Kerberos to authenticate to Active Directory for each Solaris 10 Server. In the referenced online article, this is documented in *"Preparing Active Directory (Each Solaris Server)"*.

2 Configuring Each Solaris Server

Each Solaris server must be configured to authenticate against Active Directory.

a Configuring Kerberos

Edit the Kerberos configuration files to allow it to use the Active Directory server. In the referenced online article, this is documented in *"Configuring Kerberos"*.

b Configuring LDAP

Configure LDAP to use Active Directory by utilizing the `ldapclient` utility. In the referenced online article, this is documented in *"Configuring LDAP"*.

c Configuring PAM

Change the default PAM authentication route to use Kerberos authentication when retrieving information through LDAP. In the referenced online article, this is documented in *"Configuring PAM"*.

3 Testing the Configuration

Test the configuration using `getent`, `kinit`, and `id` UNIX utilities. In the referenced online article, this is documented in *"Testing the Configuration"*.

After you complete the earlier steps, all operating system authentication are driven through Active Directory. To reconfigure certain elements of the operating system to use different authentication models, you must appropriately modify the PAM configuration file or files.

Possible Issues

When implementing this configuration, some issues were encountered—these may impact other customers who want to implement a similar PAM-LDAP scheme and are consequently documented here:

- Almost all issues were related to issues of case sensitivity with respect to entries in the configuration file `/etc/krb5.conf` and to the order in which the `ldapclients` variables were set up.
- Other issues were related to:
  - The `/etc/pam.conf` configuration file, which needed some correcting.
  - The `/etc/krb5/krb5.conf` configuration file, which needed an additional entry.

Once all of these issues were corrected LDAP and Kerberos Authentication started working.

In more detail, the remedial actions that needed to perform comprised:

1 Creating an installation-specific configuration script (called `/etc/krb5/ldap_config.sh`). It was found to be easier and less error prone to do this rather than try and compose a single, complex command to be entered through the command-line interface.

2 Correcting the `/etc/pam.conf` configuration file `auth` and `account` entries to reflect the correct authentication method and order.

3 Correcting the `/etc/krb5/krb5.conf` configuration file for case sensitive syntax and to add one additional entry for remote LDAP login authentication.

4 Changing Active Directory user `"dummy1"` password to `"<a password>"` to verify correct log in. This password still remains (no quotes).

5 Working around a known Solaris issue, for which a patch can be obtained from Sun. The issue and workaround prior to getting the Sun patch is as follows:

There is a known issue with Solaris that the administrator needs to be aware. The "name-service-cache daemon" (`nscd`) has a known bug that hangs the service from time-to-time. This stops the authentication process from occurring.

If you encounter this issue, use this quick workaround before obtaining the relevant Solaris patch:

- Perform a UNIX `ps` on `nscd`.
- Find the process id.
- Kill the process.

This automatically re-spawns the process and starts the authentication again, resolving the issue.

The UNIX terminal output described above is similar to the following:

```
svcs -a | grep name
online         17:58:55 svc:/system/name-service-cache:default

ps -ef | grep nscd
    root 15319     1   0 17:58:56 ?           0:01 /usr/sbin/nscd
```

Example Configuration Files

Configuring LDAP to Use Active Directory: /etc/krb5/ldap_config.sh

NOTES

- As mentioned earlier, this was a script developed for internal use. It is offered here only as an example. You need to tailor it to your unique installation environment.
- Variables in <> need to be manually configured to match your own domain names and user details.
- Domain names are shown in the case that they should be substituted in.
- Care must be taken to ensure that the syntax and case sensitivity used is correct.
- The order of entries is important.

```
ldapclient manual -v \
-a credentialLevel=proxy \
-a authenticationMethod=simple \
-a proxyDN=cn=<client machine>,cn=Users,dc=<domain name>,dc=com \
-a proxyPassword=<a password> \
-a defaultSearchBase=dc=<domain name>,dc=com \
-a domainName=<domain name>.com \
-a defaultServerList=<xx.xx.xx.xxx> \
-a attributeMap=group:userpassword=userPassword \
-a attributeMap=group:memberuid=memberUid \
-a attributeMap=group:gidnumber=gidNumber \
-a attributeMap=passwd:gecos=cn \
-a attributeMap=passwd:gidnumber=gidNumber \
-a attributeMap=passwd:uidnumber=uidNumber \
-a attributeMap=passwd:homedirectory=unixHomeDirectory \
-a attributeMap=passwd:loginshell=loginShell \
-a attributeMap=shadow:shadowflag=shadowFlag \
-a attributeMap=shadow:userpassword=userPassword \
-a objectClassMap=group:posixGroup=group \
-a objectClassMap=passwd:posixAccount=user \
-a objectClassMap=shadow:shadowAccount=user \
-a serviceSearchDescriptor=passwd:dc=<domain name>,dc=com?sub \
-a serviceSearchDescriptor=group:dc=<domain name>,dc=com?sub
```

Configuring PAM: /etc/pam.conf

NOTES

- Ensure that the `auth` and `account` entries in the `/etc/pam.conf` file reflect the correct authentication method and order to use, as illustrated in the partial file listing below.
- An additional entry should also be provided to allow for remote log in (for example, `telent`, `ssh`, and `su`).

```
#
# <domain name>
# Default definition for Authentication management #
other   auth requisite          pam_authtok_get.so.1
other   auth required           pam_dhkeys.so.1
other   auth sufficient         pam_krb5.so.1
other   auth required           pam_unix_cred.so.1
other   auth required           pam_unix_auth.so.1
#
# Default definition for Account management #
other   account requisite       pam_roles.so.1
other   account sufficient      pam_unix_account.so.1
other   account required        pam_ldap.so.1
```

Configuring Kerberos/LDAP: /etc/krb5/krb5.conf

NOTES

- Ensure that the entries in the `/etc/krb5/krb5.conf` file have been edited correctly for case sensitivity, as illustrated in the partial file listing below.
- Add an additional entry for remote LDAP login authentication to the `/etc/krb5/krb5.conf` file.
- Variables in <> need to be manually configured to match your own domain names and user details.
- Domain names are shown in the case that they should be substituted in.

```
[libdefaults]
        default_realm = <DOMAIN NAME>.COM
        dns_lookup_kdc = true
        dns_lookup_realm = true
        verify_ap_req_nofail = false

[realms]
        <DOMAIN NAME>.COM = {
        kdc = <first server>.<domain name>.com
        kdc = <second server>.<domain name>.com
        admin_server = <second server>.<domain name>.com
        }

[domain_realm]
        .<domain name>.com = <DOMAIN NAME>.COM
        <domain name>.com = <DOMAIN NAME>.COM

[logging]
        default = FILE:/etc/krb5/kdc.log
        kdc = FILE:/etc/krb5/kdc.log
        kdc_rotate = {
        period = 1d
        version = 10
        }


[appdefaults]
        kinit = {
        renewable = true
        forwardable= true
        }
```

Dimensions Support for SSO

Introduction

When SSO is enabled in a Dimensions CM server, authentication is performed using the `UserName/Password` or the SSO token. In the first case, the Dimensions CM server obtains the SSO token using user credentials from the Security Server Identity Provider (IDP). In the second case, the Dimensions CM server directly validates the SSO token (validating attributes such as the Digital Sign, Expiration, and Relaying Party).

When the user logs into a Dimensions web client the normal login page is displayed if SSO is not enabled. If SSO (or SSO plus CAC) is enabled, the SBM login screen is displayed.

After the user enters their credentials, the login application passes the information:

- If the user's credentials are recognized by the authentication server, the user receives a security token and is redirected back to the desired destination (in this example, Dimensions web client). When the user subsequently tries to log into the web interface of another SSO-enabled tool (such as Solutions Business Manager), they are redirected to the destination without having to log in again. This is true as long as the user keeps the original browser session.
- If the user's credentials are not recognized by the authentication server, the Gatekeeper component of SSO intercepts the login request and an error message is displayed.

Prerequisites for SSO Authentication

Client Prerequisites

The SSO software is all server side and there are no client prerequisites.

Server Prerequisites For an Existing SSO Server

The following information is requested by the Dimensions CM installer if you choose to use an existing SSO server (for example, an SSO-enabled SBM server installation):

Existing SSO Parameter Required	Description
Hostname	Host name of the existing SSO Server.
SSO Port	HTTP or HTTPS TCP port used by the existing SSO server. If the port is not an HTTPS port, then the **Secure (https) Connection** checkbox parameter (see below) *must* remain in its default unchecked state.
Secure (https) Connection	Checkbox (unchecked by default) to be checked to let the installer know that Secure Socket Layer (SSL) communication is required.

The SBM software and documentation, if required, can be downloaded from Support. For enabling an installed SBM server for SSO, see the *SBM Administrator Guide*.

Server Prerequisites For a New SSO Server

To create a new SSO server, the following details are requested:

Base Authentication Method	SSO Parameter Required	Description
Light Directory Access Protocol (LDAP)	Hostname	The host name of the Domain Controller (Active Directory), or the machine that serves LDAP requests (usually the former).
	Port	TCP port (by default 389) to be used by the new SSO server.
	Base DN	The LDAP base DN to be used by the SSO server. The base DN is the top level within the LDAP directory tree below which the search for the user should be performed—it should look like: `CN=Users,DC=your,DC=domain,DC=com`

Base Authentication Method	SSO Parameter Required	Description
Light Directory Access Protocol (LDAP) (continued)	Search Filter	The LDAP search filter to be used by the new SSO server. The installer prepopulates this with a default search filter. LDAP search filters function within a framework. The framework includes what attributes you are searching on and the value or range of values that you are trying to match. Each search filter involves at a minimum of three components: ■ The attributes to search for, called the attribute data type. ■ The search filter operator that determines what to match, sometimes called the match operator. ■ The actual value of the attribute you are searching for. Each search needs to have at least one of each of the components. You can create compound search filters by connecting two or more search filter modules. They are enclosed in parentheses to clarify filter content, and include one or more compound search filter operators (AND, OR, NOT). You can add multiple compound and wildcard filters as long as you have the correct number of matching parentheses. The actual search filter in the case of Microsoft Active Directory (Domain Controller) should look like: `(&(objectClass=user)(sAMAccountName={0}))` where `{0}` is the actual username that is logging in. For details about LDAP search filters and a mechanism for representing them as strings, see the LDAP RFC 4515 documentation.
	Bind User DN	The LDAP bind user DN to be used by the new SSO server. The bind user DN is the user on the external LDAP server permitted to search the LDAP directory within the defined search base. Most of the time, the bind DN is permitted to search the entire directory. The role of the bind DN is to query the directory using the LDAP query filter and search base for the DN (distinguished name) for authenticating users. When the DN is returned, the DN and password are used for authentication.
	Password	The LDAP password to be used to be used in conjunction with the bind user DN by the new SSO server.

Supported SSO Scenarios

The following SSO scenarios are supported:

- An orchestration can receive ALF events from Dimensions CM and talk to Dimensions CM through web services without the use of a password—that is, both Dimensions CM web services and ALF events can work with SSO tokens.
- After logging in to the Dimensions CM web client, if you use the same browser session you can use the web interface of another SSO-enabled tool (such as Solutions

Business Manager, Dimensions RM, or the Dimensions CM Administration Console) without having to log in again.

NOTE The default configuration of Dimensions CM stores the current user in temporary cookies, that is, it doesn't store it permanently. But it is possible to configure the federation server to store the user permanently.

- As the external request integration supports SSO, the Dimensions CM Eclipse and Visual Studio integrations are able to interact with an SSO-enabled SBM without explicit authorization. This enables Dimensions CM.
- Other Dimensions CM tools are unaffected by the use of SSO. When SSO is enabled, dmcli, local connect, and others work normally and the user still needs to log in explicitly (if appropriate).

Configuring a Dimensions Server for SSO

Configuration Entries for SSO Authentication

Provided a new SSO server is installed using the Dimensions CM installer or an existing SSO server is nominated to the Dimensions CM installer, the Dimensions CM server `dm.cfg` file is automatically configured by the installer with the requisite symbols (and default values) for use with SSO authentication. If any of these values require editing (for example, `SSO_TOKEN_LIFE_TIME`), you need to restart the Dimensions CM server for those edits to take effect.

In addition to the installer-generated `dm.cfg` symbol entries, you need to manually configure trusted Certificate Authorities to enable SSO (and SSO plus CAC) authentication. For details, see "Configuring Trusted Certificate Authorities for SSO Installations" on page 475.

`dm.cfg` contains the following SSO parameters:

- DM_SSL_TRUST_CERTS_FILE

  Specifies the location of a .pem based certificates store, for example:

  `%dm_root%\dfs\cacerts.pem`

  For more details see page 493.

- DM_AUTH_TYPE_DBS <auth_type>

  This is a mandatory parameter. It is used to specify the authentication type that Dimensions CM uses. To enable SSO authentication, it needs to be set to `SSO`.

- SSO_STS_ENDPOINT

  This is a mandatory parameter. It needs to contain the IDP endpoint used to obtain the SSO token, and is of the form:

```
http://<SSO_Host>:<port_number>/idp/services/Trust
```

IMPORTANT! For security purposes, we recommend that a Secure Sockets Layer (SSL) connection to the IDP be employed, in which case the appropriate secure port number should be used. If a non-SSL connection is used instead, the default port is the default Tomcat port `8080` for Dimensions CM and `8085` for SBM. For more information on SSL, see "SSL Connection Considerations for an SBM SSO Server" on page 482.

NOTE The IDP endpoint also needs to be configured for Dimensions web client. For details, see "Configuring the Dimensions Web Client for SSO" on page 480.

- SSO_TRUST_CERTIFICATE

  This is a mandatory parameter. It needs to contain the path to the trusted certificate file (*.PEM), which is used to validate the SSO token. By default this is `%DM_DFS%` (Windows) and `$DM_DFS` (UNIX). A default certificate is supplied in this directory, but it needs to be regenerated at every site for security purposes. Also, by default, Dimensions CM is configured to use the same certificate as SBM.

  Please contact Support for details on changing a certificate:

- SSO_RELYING_PARTY

  This is a mandatory parameter. It needs to contain SSO "Relying Party" information, which is used to validate and request the SSO token. an example could be:

  ```
  uri:org:eclipse:alf:sso:relyingparty:anonymous:anonymous:anonymous
  ```

- SSO_TOKEN_LIFE_TIME

  This is an optional parameter. It contains the time in seconds for which the requested SSO Token is valid. If this parameter is not specified, it is set by default to `28800` (eight hours).

IMPORTANT! The Dimensions CM 2009 R2 installer assigns a value of `28800` seconds (eight hours) to `SSO_TOKEN_LIFETIME` in `dm.cfg`. The value is consistent with the parameter value assignment in the XML file detailed below (and if this value is altered, it should be consistently altered in both dm.cfg and the XML file):

- Windows

  ```
  <Tomcat-Root>\alfssogatekeeper\conf\gatekeeper-core-config.xml
  ```

- UNIX

  ```
  <Tomcat-Root>/alfssogatekeeper/conf/gatekeeper-core-config.xml
  ```

parameter-value assignment:

```
<parameter name="SecurityTokenTTL" Type="xsd:decimal">28800
</parameter>
```

NOTE IDP can ignore this parameter if it greater than MaxTokenLifetime parameter in the IDP configuration.

- SSO_CLOCK_TOLERANCE

  This is an optional parameter. It contains the "Expiration Tolerance" time in seconds, used to validate the SSO token. If this parameter is not specified, it is set by default to `0`.

  This parameter allows for possible "clock slew" in those circumstances where the server and relying party clocks are not perfectly aligned. For example, a SSO token might be issued, say, at 12:00:00 but the relying party (library) clock is 2-3 mins behind, so it thinks it is 11:57:00. In this case, the SSO token is rejected even though it is valid. To cater for such situations, this parameter lets you assign some small (configurable) amount of time accounting to allow for "clock skew."

- SSO_DONT_USE_BEFORE_EXPIRATION_TIME

  This is an optional parameter. It enables the user to ensure that an SSO token does not expire in the middle of an operation (for example, an ALF event). If this parameter is not specified, it is set by default to `300` seconds.

  At the start of a long running orchestration, the SSO token is validated and it needs to remain valid during the life of that orchestration, or it fails.

  This parameter checks that an SSO token is valid for the specified period (seconds). If the token is:

  - Valid: the operation continues.
  - Invalid: the operation does not start. It requires that the user logs in again.

- SSO_ENDORSING_USER
- SSO_ENDORSING_USER_PASSWORD

  These parameters endorse a user name and password with the requisite permissions to get an SSO token on behalf of another user.

  If `$$DMSECURE$$` is specified in the `SSO_ENDORSING_USER_PASSWORD parameter`, then the password should be specified using `dmpasswd` for the user specified in the `SSO_ENDORSING_USER` parameter.

  These parameters are mandatory, but are only used in the case when Dimensions CM should obtain the token for a user for which it does not know the password. (`dmcli` can be used in such a way, when it uses the current Windows user to authenticate to Dimensions CM.)

Changing SSO Endorsing User Credentials in Dimensions CM and SBM

Dimensions CM or SBM is configured with two SSO endorsing users to establish trust between certain IDP clients and the IDP:

- SSO Endorser User for the Application Engine

  An SSO Endorser user account is automatically added to Dimensions CM or SBM and is configured to communicate with the Security Token Server. This user account is assigned API/Script product access. This product access type does not consume seat or named licenses and users assigned API/Script access cannot log into the web interface. Privileges are not assigned to the SSO Endorser user account.

- SSO Endorser User for Special Requests

  These credentials do not necessarily correspond to a user identity and do need to exist in Dimensions CM, SBM, or the underlying identity store, but they are used by certain clients to make requests to the IDP. The IDP honors these requests only

if they come from a trusted entity. You can change the default trust relationship and trusted user credentials after installation by following the steps below:

To change the SSO Endorser User for the Application Engine:

a If using an SBM SSO, open SBM Administrator and modify the password for the provided SSO Endorser account. If you choose not to use default account for the SSO Endorser, create a dedicated user account for this purpose and assign this user API/Script access.

b On the server where SSO is installed, navigate to the following directory:

- Dimensions CM (Windows)

  ```
  <Tomcat-Root>\webapps\idp\WEBINF\conf\Configuration.xml
  ```

- Dimensions CM (UNIX)

  ```
  <Tomcat-Root>/webapps/idp/WEBINF/conf/Configuration.xml
  ```

- SBM

  ```
  <install_directory>\Common\Tomcat
   <version>\server\default\deploy\
   idp.war\WEB-INF\conf
  ```

c In an editor, open the Dimensions CM `Configuration.xml` file or the SBM `configuration.xml` file.

d Search for the string containing `"MasterUsername"`.

e Change the values of `MasterUsername` and `MasterPassword` as needed.

f Save and close the file.

g Change the SSO_ENDORSING_USER and SSO_ENDORSING_USER_PASSWORD parameters in dm.cfg to be consistent.

To modify credentials for the SSO Endorser User for special requests, you must first establish new endorsing credentials in the IDP:

a Navigate to:

- Dimensions CM (Windows)

  ```
  <Tomcat-Root>\webapps\idp\WEBINF\conf\Configuration.xml
  ```

- Dimensions CM (UNIX)

  ```
  <Tomcat-Root>/webapps/idp/WEBINF/conf/Configuration.xml
  ```

- SBM

  ```
  <Installation Directory>\
   Business Manager\Common\Tomcat <version>\
   server\default\deploy\idp.war\WEBINF\conf\Configuration.xml
  ```

b In the following section, change the `Username` setting to a new name and modify the password setting:

```
        <Setting Name="TrustedDelegators" Type="htf:namedlist">
          <Setting Name="UsernameSubject" Type="htf:map">
            <Setting Name="Username" Type="xsd:string">endorsing-user</Setting>
            <Setting Name="Password" Type="xsd:string">changeit</Setting>
          </Setting>
        </Setting>
```

NOTE The user name and password do not need to correspond to a Dimensions CM user, SBM user, or a user on your network. The credentials are used solely for establishing trust between the IDP and the Dimensions CM or SBM clients.

c Using the credentials established in step b, change the following files to establish new endorsing credentials in SSO gatekeepers and the Federation Server:

- Dimensions CM (Windows)

  `<Tomcat-Root>\alfssogatekeeper\conf\gatekeeper-core-config.xml`

  `<Tomcat-Root>\idp\WEBINF\conf\fedsvr-core-config.xml`

- Dimensions CM (UNIX)

  `<Tomcat-Root>/alfssogatekeeper/conf/gatekeeper-core-config.xml`

  `<Tomcat-Root>/idp/WEBINF/conf/fedsvr-core-config.xml`

- SBM

  ```
  <Installation Directory>\Business Manager\
   Application Engine\alfssojavabridge\
   WEB-INF\alfssogatekeeper\conf\gatekeeper-core-config.xml
  ```

  ```
  <Installation Directory>\Business Manager\Common\Tomcat
   <version>\server\default\alfssogatekeeper\conf\gatekeeper-
   core-config.xml
  ```

  ```
  <Installation Directory>\Business Manager\Common\Tomcat
   <version>\server\default\fedserver\conf\
   fedsvr-core-config.xml
  ```

```
<parameter name="PrincipalEndorsment">
     <PrincipalEndorsmentMethods>
       <UsernameCredential>
         <Username>endorsing-user</Username>
         <Password>changeit</Password>
       </UsernameCredential>
       <X509CertificateSignature>
         <ApplicationSignature/>
       </X509CertificateSignature>
     </PrincipalEndorsmentMethods>
   </parameter>
```

d If using an SBM SSO, change the following file to establish new endorsing credentials in the SBM administration tool.

```
<Installation Directory>\Business Manager\Common\Tomcat
    <version>\server\default\deploy\mashupmgr.war\
    WEB-INF\web.xml
```

```
<!-- Endorsing username to be used for insufficient credentials token requests. e.g.
           "endorsing-user" -->
       <context-param>
              <param-name>STS_USER</param-name>
              <param-value>endorsing-user</param-value>
       </context-param>
<!-- Endorsing password to be used for insufficient credentials token requests. e.g.
           "secret" -->
    <context-param>
       <param-name>STS_PASS</param-name>
       <param-value>changeit</param-value>
    </context-param>
```

e If using an SBM SSO, stop and start the Application Engine web server and the SBM Tomcat service.

f Change the SSO_ENDORSING_USER and SSO_ENDORSING_USER_PASSWORD parameters in dm.cfg to be consistent.

Configuring Trusted Certificate Authorities for SSO Installations

Introduction

For SSO and SSO plus Common Access Card (CAC) installations, the most important part of authentication by certificate is checking that the certificate for user, services, and other (different) purposes was issued by a trusted Certificate Authority (CA). To configure CAs correctly you should have certificate of your certificate authority (it can be CA on a Microsoft Domain Controller or externally based on OpenSSL).

Storing/Adding a Certificate into a Java Key Store

The standard Java tool "keytool" can be used to perform various operations with Java Key Store (*.JKS).

To create a new keystore or add a new certificate to existing keystore, the following command can be used:

```
"%JAVA_HOME%\bin\keytool" -import -keystore <your_keystore_file_name> -
            storepass <your_keystore_password> -file <cert_to_import> -
            alias <your_cert_alias>
```

where:

<your_keystore_file_name>	is the existing or new keystore file name to which the certificate should be added.
<your_keystore_password>	is the password for the keystore.
<cert_to_import>	is the certificate that should be added to the keystore. It can be *.PAM, *.CER (Base64 or DER encoded), or *.CRT.
<your_cert_alias>	is the alias of the certificate in the keystore.Each such certificate in the keystore has an unique alias/ name.

Configuring Truststore in the Security Server

To configure trusted CAs, you should specify one or more keystore and certificate aliases from the keystores in the X509-LDAP (or X509-BASE) authenticators of the IDP. This is configuration is achieved by appropriately editing (as explained below) the main IDP configuration file located at:

```
<TOMCAT_HOME>\webapps\idp\WEB-INF\conf\Configuration.xml
```

The following sample and template shows how to configure trusted CAs—please pay special attention to the "CertificateIssuerTrustMatcher" section:

```
<Setting Name="serena-ldap-authenticator" Type="htf:map">
  <Setting Name="Provider" Type="xsd:string">X509-LDAP</Setting>
  <Setting Name="CertificateMustExistInLDAP" Type="xsd:boolean">false</Setting>
  <Setting Name="CertificateAttributeName" Type="xsd:string"></Setting>
  <Setting Name="SearchFilter" Type="xsd:string">(objectclass=*)</Setting>
  <Setting Name="CompatibleRequestMatchers" Type="htf:namedlist">
    <Setting Name="CredentialsTypeMatcher" Type="xsd:string">X509</Setting>
    <Setting Name="AuthenticationTypeMatcher" Type="xsd:string">*</Setting>
    <Setting Name="CertificateIssuerDNMatcher" Type="xsd:string">*</Setting>
    <Setting Name="CertificateIssuerTrustMatcher" Type="htf:map">
      <!-- Sample Entry -->
      <Setting Name="serena-truststore" Type="htf:keystore">
        <Setting Name="Type" Type="xsd:string">JKS</Setting>
        <Setting Name="File" Type="htf:file">serenaca.jks</Setting>
        <Setting Name="Password" Type="xsd:string">changeit</Setting>
      </Setting>
      <Setting Name="serenaca" Type="htf:certificate">
        <Setting Name="KeyStoreName" Type="xsd:string">serena-truststore</Setting>
        <Setting Name="Alias" Type="xsd:string">serenaca</Setting>
      </Setting>

      <!-- Template Entry -->
      <Setting Name="[your_keystore_alias]" Type="htf:keystore">
        <Setting Name="Type" Type="xsd:string">JKS</Setting>
        <Setting Name="File" Type="htf:file">[your_keystore_file_name]</Setting>
        <Setting Name="Password" Type="xsd:string">[your_keystore_password]</Setting>
      </Setting>
      <Setting Name="[your_certificate_alias(2)]" Type="htf:certificate">
        <Setting Name="KeyStoreName" Type="xsd:string">[your_keystore_alias]</Setting>
        <Setting Name="Alias" Type="xsd:string">[your_certificate_alias]</Setting>
      </Setting>
    </Setting>
  </Setting>
 <Setting Name="JNDI.Environment" Type="htf:map">
    <Setting Name="java.naming.factory.initial"
             Type="xsd:string">com.sun.jndi.ldap.LdapCtxFactory</Setting>
    <Setting Name="java.naming.provider.url" Type="xsd:string">
             ldap://serena.com:389</Setting>
    <Setting Name="java.naming.security.authentication" Type="xsd:string">simple
           </Setting>
    <Setting Name="java.naming.security.principal" Type="xsd:string">ldapuser</Setting>
    <Setting Name="java.naming.security.credentials" Type="xsd:string">changeit
          </Setting>
  </Setting>
</Setting>
```

where:

[your_keystore_alias]	is any unique keystore name/alias, for example, my_company_ca_store.
[your_keystore_file_name]	is the existing keystore filename, full path, or relative path to the directory where Configuration.xml is located.
[your_keystore_password]	is the keystore password.
[your_certificate_alias]	is the existing certificate alias from **[your_keystore_file_name]**.
[your_certificate_alias(2)]	is any unique certificate name/alias, for example, my_company_ca-01. It can be the same as **[your_certificate_alias]**.

Establishing a Certificate Revocation List for CAC Certificates

Introduction

A Certificate Revocation List (CRL) is one of two common methods when using a public key infrastructure for maintaining user access to servers in a network. The other, newer method, which has superseded CRL in some cases, is Online Certificate Status Protocol (OCSP).

The CRL is exactly what its name implies: a list of subscribers paired with digital certificate status. The list enumerates revoked certificates along with the reasons for revocation. The dates of certificate issue, and the entities that issued them, are also included. In addition, each list contains a proposed date for the next release. When a potential user attempts to access a server, the server allows or denies access based on the CRL entry for that particular user. As part of CAC authentication, you have the option of comparing user certificates against one or more CRLs.

The main limitation of a CRL is the fact that updates must be frequently dowloaded to keep the list current. OCSP overcomes this limitation by checking certificate status in real time.

*Configuring Dimensions Security Server with a Locally Stored *.crl File*

1 Shut down Tomcat (see "Stopping Tomcat" on page 29).

2 Navigate to the following file and open in a text editor

- Windows

  ```
  <Tomcat-Root>\webapps\idp\WEB-INF\conf\Configuration.xml
  ```

- UNIX

  ```
  <Tomcat-Root>/webapps/idp/WEB-INF/conf/Configuration.xml
  ```

3 Comment out the following block of parameters:

```
<Setting Name="serena-crl-validator" Type="htf:map">
   <Setting Name="Provider" Type="xsd:string">X509-CRL</Setting>
   <Setting Name="CompatibleRequestMatchers" Type="htf:namedlist">
     <Setting Name="CredentialsTypeMatcher" Type="xsd:string">X509</Setting>
     <Setting Name="AuthenticationTypeMatcher" Type="xsd:string">*</Setting>
     <Setting Name="CertificateIssuerDNMatcher" Type="xsd:string">*</Setting>
   </Setting>
   <Setting Name="CRLDir" Type="xsd:string"></Setting>
   <Setting Name="CacheFileName" Type="xsd:string">CRLInfoCache.xml</Setting>
   <Setting Name="RefreshPeriod" Type="xsd:string">180</Setting>
</Setting>
```

4 Copy the *.crl file from the LDAP machine.

On Windows, the location is:

```
C:\WINDOWS\system32\certsrv\CertEnroll\
```

On UNIX, you have to locate the appropriate directory.

5 Restart Tomcat (see "Starting Tomcat" on page 27).

*Configuring Dimensions Security Server with a Remotely Stored *.crl File*

1 Shut down Tomcat (see "Stopping Tomcat" on page 29).

2 Modify the settings of the Tomcat service log on. It should run under an account that can access the directory containing the `*.crl` file on the remote computer, for example, `dmsys`.

3 Restart Tomcat (see "Starting Tomcat" on page 27).

Dimensions Utility for Generating Encrypted Passwords Strings for IDP

The following scripts:

- Windows

  ```
  <Tomcat-Root>\bin\sso_encstring.bat
  ```

- UNIX

  ```
  <Tomcat-Root>/bin/sso_encstring.sh
  ```

provide a utility for generating encrypted passwords strings for the IDP component of an SSO server. The IDP component utilizes the following XML file:

- Windows

  ```
  <Tomcat-Root>\webapps\idp\WEB-INF\conf\Configuration.xml
  ```

- UNIX

  ```
  <Tomcat-Root>/webapps/idp/WEB-INF/conf/Configuration.xml
  ```

which has parameter entries of the form:

```
<Setting Name="Password"
    Type="htf:encstring">enc:01089289f182aef03a6657af3f9f1e813718ce4
    7b224b78fc5ac</Setting>
```

The sso_encstring.bat (Windows) or sso_encstring.sh (UNIX) utility enables you to generate values for these Type="htf:encstring" parameters in the XML file.

NOTE If you do not want to use encrypted passwords (for example while testing), the Type="htf:encstring" parameter can be changed to Type="xsd:string" to utilize a plain text password string.

Configuring the Dimensions Web Client for SSO

Enabling SSO in Dimensions Web Client

In Dimensions CM 2009 R1.x, the comment tags (<!-- and -->) placed around the 'Serena SSO Gatekeeper Filter Configuration' section in the 'web.xml' file located in the following directory:

- Windows

  <Tomcat-Root>\conf

- UNIX

  <Tomcat-Root>/conf

had to be removed manually for an SSO installation.

Starting with Dimensions CM 2009 R2, when SSO or SSO plus CAC is installed the Dimensions CM installer automatically removes these comment tags for you.

Configuring SSO Gatekeeper

In Dimensions CM 2009 R1.x, the 'gatekeeper-core-config.xml' file located in the following directory:

- Windows

  <Tomcat-Root>\alfssogatekeeper\conf

- UNIX

  <Tomcat-Root>/alfssogatekeeper/conf

has unassigned parameter values for the 'SecurityTokenService' and 'FederationServerURL' nodes, and these had to be assigned manually for an SSO installation.

Starting with Dimensions CM 2009 R2, when SSO or SSO plus CAC is installed the Dimensions CM installer automatically makes these parameter assignments for you.

SSO-Enabled Dimensions Web Client Login

When an SSO-enabled Dimensions web client is opened for the first time under SSO authentication, the user sees the SSO-Federation Server login page instead of the standard Dimensions web client login page.

If the SSO user login credentials are successfully negotiated, the user either is:

- prompted for the usual Dimensions CM database parameters, or
- logged in using default DB configured on server side.

The `sso_use_default_db_for_login` parameter in the file

`<TOMCAT_ROOT>\webapps\dimensions\WEB-INF\web.xml`

defines which of the above mechanisms is used (the default value is **no**):

```
<init-param>
  <param-name>sso_use_default_db_for_login</param-name>
  <param-value>no</param-value>
  <description>
    If true, Login page with DB selection won't be shown and default DB parameters will
          be used
  </description>
</init-param>
```

An SSO-enabled Dimensions web client login page has the following differences to a standard Dimensions web client login page:

- Instead of the usual username/password fields, the following message is displayed: **Your user name and password has been validated with SSO**.
- An additional **Logout** link is available next to the Help and About links at the top right of the login page. This addition link performs an SSO Logout, after which the user is returned to SSO-Federation login page.

If the sso_use_default_db_for_login mentioned earlier is set to **true/yes** and the log in gets failed, then the additional **Server default** value of **Connect to** appears.

SSO-Enabled Dimensions Web Client Logout

When a Dimensions web client is SSO-enabled, two logout mechanisms are supported, full and partial.

Partial Log Out

Partial log out means that an authenticated user can come back to a Dimensions web client without entering login/password when they:

- Close a browser with Dimensions web client.
- Close a browser tab with Dimensions web client.
- Open another web application, site, etc in a tab within Dimensions web client.

In such cases, the SSO logout functionality:

- Closes the HTTP session.

- Releases Dimensions web client licenses.
- Closes the connection to the Dimensions CM server.

Full Log Out

An authenticated Dimensions web client user can do a full log out by first using the **Logout** button. They are then redirected to main SSO login page.

- In such a case, the SSO logout functionality:
- Closes the HTTP session.
- Releases Dimensions web client licenses.
- Closes the connection to the Dimensions CM server.
- Causes the Gatekeeper filter to invalidate the token and redirect the user to the main SSO Login page.

Other SSO Logout Functionality for Dimensions Clients

See the *Dimensions CM online help* for a discussion of the logout experiences for other SSO-enabled (and SSO plus CAC-enabled) Dimensions clients.

Adding a System Administrator as an SBM User

If you want to use a Solutions Business Manager SBM SSO server, you have to create, on SBM, a user with same identity and password as that of the Dimensions system administrator user on Dimensions CM. By default this user is `dmsys`. To do this proceed as follows (consult the SBM documentation if you need further details):

1. SBM Administrator | Users tabs.
2. Select **Admin** user.
3. Press **Copy**.
4. Enter a new user i/d with the same i/d as the Dimensions system administrator.
5. Assign the same password to the new the SBM user as that used for the Dimensions system administrator.

SSL Connection Considerations for an SBM SSO Server

If you are using an SSO-enabled SBM as the SSO server for Dimensions CM, SSO components on SBM trust to certificates from their JVM keystore, so the certificate file that Tomcat uses for Dimensions CM must be added to the Java Virtual Machine (JVM) that SBM uses.

Proceed as follows:

1. On the Dimensions CM server, configure SSL connection in Tomcat, see page 491.

2 Export the certificate to a "*.cer" file:

```
keytool -export -alias tomcat -file tomcat.cer
```

3 Place tomcat.cer in the following directory:

```
<sbm-install-dir>\Common\jdk1.8\jre\lib\security\
```

4 The certificate now needs to be imported into to the SBM JVM. Any imported Tomcat certificate (on the SBM machine) must be stored in the existing file in which the Root/System/Trusted certificates of JVM are currently stored, namely:

```
<sbm-install-dir>\Common\jdk1.8\jre\lib\security\cacerts
```

To add tomcat.cer to this store:

a Navigate to:

```
<sbm-install-dir>\Common\jdk1.8\jre\lib\security\
```

b Run the following command:

```
..\..\bin\keytool.exe -import -trustcacerts -keystore cacerts -storepass changeit
-noprompt -alias cm_tomcat -file tomcat.cer
```

5 Restart SBM Tomcat.

Check that can log into Dimensions web client through an SSL connection such as:

```
https://current_machine_name:8443/dimensions
```

Updating an IDP Certificate

We recommend that you generate new key pairs to secure your SBM installation. If you do not generate new key pairs, the default certificates that the IDP inherently trusts are used. These same certificates are available from any SBM installation and can be easily spoofed; therefore, to increase security, you should generate a new unique certificate for each SSO component from time to time.

IMPORTANT! When you update an IDP certificate on the SBM side, you must then also update the certificate on the Dimensions CM side.

To update an IDP certificate:

1 On the SBM machine, start SBM Configurator, navigate to Security | SSO Trust Keys | IDP, and then click **Generate Keypair** to regenerate the IDP certificate.

2 Click **Export Certificate** to export the newly generated certificate to a .pem file (for example, C:\Temp\sts.pem).

3 Copy the .pem file to the Dimensions CM machine (for example, in the C:\Temp directory).

4 On the Dimensions CM machine, navigate to <tomcat-install-dir>\alfssogatekeeper\conf, and make a backup copy of the truststore.jks file.

5 Navigate to <java-install-dir>\jre\<jre-version>\bin, and delete the existing IDP certificate:

```
keytool -delete -keystore "<tomcat-install-
```

```
    dir>\alfssogatekeeper\conf\truststore.jks"
    -alias sts
```

6 Import the certificate from the folder to which you copied it, for example: `C:\Temp\sts.pem` into `truststore.jks`:

```
keytool -import -keystore "<tomcat-install-
    dir>\alfssogatekeeper\conf\truststore.jks"
    -file "C:\Temp\sts.pem" -alias sts
```

7 Enter the default password **changeit** at the prompt.

8 Create a backup copy of `<dimcm-install-dir>\CM\dfs\sts.pem`.

9 Copy `C:\Temp\sts.pem` (if that is where you placed the file) to `<dimcm-install-dir>\CM\dfs\sts.pem`.

10 Restart the Dimensions Tomcat and Listener services.

Troubleshooting SSO

- We recommend that you do not use an URL like `127.0.0.1` or `localhost` in `dm.cfg`, `web.xml`, and `gatekeeper-services-config.xml`. Such SSO configuration works only when the client is on the same machine as the server. All Dimensions web client SSO connections from remote machines do not login correctly in such circumstances.
- It is generally recommended that you only use full DNS computer names in Dimensions CM `dm.cfg`, `web.xml`, and `gatekeeper-services-config.xml` files (for example, use `current_machine_name.acme.com`, not `current_machine_name`). There can be possible failures if you use partial DNS machine names in SSO redirections, and these can be very hard to diagnose. If in doubt, you can, however, use IP address in all three files instead of the machine name.

  However, the SBM installer does not use the full DNS machine name to configure endpoints, as can be seen from:

  ```
  <sbm-install-dir>\Common\Tomcat\server\default\
      alfssogatekeeper\conf\gatekeeper-core-config.xml
  ```

  Consequently, as SBM is being used to SSO-enable Dimensions CM, for correct SSO configuration the endpoints for the Dimensions CM SSO gatekeeper must be exactly the same as the endpoints for the SBM SSO Gatekeeper.

NOTE SBM uses two different gatekeeper instances (and separate gatekeeper configuration files): one for Tomcat and other for IIS.

- A temporary cookie is used for SSO, so SSO is quite sensitive to browser cookie settings. Issues have been reported with Internet Explorer Enhanced Security Configuration (which is enabled by default on Windows 2003 server). As we do not recommend to uninstall Internet Explorer Enhanced Security Configuration, to ensure that cookies are working as expected in such scenarios, we recommend that you proceed as follows:

  Modify the Internet Explorer setting on each client machine, by adding both Dimensions server site and SBM server site, for example:

  ```
  http://current_machine_name.serena.com
  ```

to the **Local Intranet** zone list:

```
Tools | Internet Options | Security | Local intranet | Sites |
    Advanced | Add this Web site to the zone
```

Dimensions Support for SSO with CAC

Introduction

SSO is a prerequisite for the use of Common Access Card (CAC). Familiarize yourself with "Dimensions Support for SSO" on page 467 before reading the following sections.

Prerequisites for SSO with CAC Reader Authentication

Windows Client Prerequisites

The following client side prerequisites are required:

- The installation of CAC ActivClient 6.1 or later software. All configuring of the ActivClient client, if necessary, should be performed as described in the vendor documentation. For details on how to log in using CAC and your PIN in the various Dimensions CM clients, see the *Dimensions CM online help*.
- That each user has a personal CAC.
- That a CAC Reader is attached to the client machine (either as a standalone reader or an integrated keyboard reader).

Server Prerequisites for an Existing SSO Server with CAC Reader

The following information is requested by the Dimensions CM installer if you choose to use an existing SSO server plus CAC reader (for example, an SSO-enabled SBM server installation):

Existing SSO Parameter Required	Description
Hostname	Host name of the existing SSO Server.
SSO Port	HTTP or HTTPS TCP port used by the existing SSO server. If the port is not an HTTPS port, then the **Secure (https) Connection** checkbox parameter (see below) *must* remain in its default unchecked state.
Secure (https) Connection	Checkbox (unchecked by default) to be checked to let the installer know that Secure Socket Layer (SSL) communication is required.

The SBM software and documentation, if required, can be downloaded from the Support website. For enabling an installed SBM server for SSO, see the *SBM Administrator Guide*.

Local or Remote Windows Server Prerequisites for a New SSO Server with CAC Reader

The following CAC and SSO information (in that order) is requested by the Dimensions CM installer if you choose to create a new Windows SSO server plus CAC reader setup software:

Base Authentication Method	CAC Parameter Required	Description
Light Directory Access Protocol (LDAP)	Hostname	The host name of either the Domain Controller (Active Directory) or the machine that serves LDAP requests. It is usually the former.
	Port	TCP port (by default 389) to be used by the new SSO server.
Light Directory Access Protocol (LDAP) (continued)	Bind User DN	The LDAP bind user DN to be used by the new CAC setup software. The bind user DN is the user on the external LDAP server permitted to search the LDAP directory within the defined search base. Most of the time, the bind DN is permitted to search the entire directory. The role of the bind DN is to query the directory using the LDAP query filter and search base for the DN (distinguished name) for authenticating users. When the DN is returned, the DN and password are used for authentication.
	Password	The LDAP password to be used to be used in conjunction with the bind user DN by the new CAC setup software.

Base Authentication Method	SSO Parameter Required	Description
Light Directory Access Protocol (LDAP)	Hostname	The host name of either the Domain Controller (Active Directory) or the machine that serves LDAP requests. It is usually the former. By default, the installer prepopulates this parameter field with the LDAP value supplied earlier for the CAC setup software.
	Port	TCP port (by default 389) to be used by the new SSO server. By default, the installer prepopulates this parameter field with the LDAP value supplied earlier for the CAC setup software.
	Base DN	The LDAP base DN to be used by the new SSO server. The base DN is the top level in the LDAP directory tree under which the search for the user should be performed. It look like this: `CN=Users,DC=your,DC=domain,DC=com`
	Search Filter	The LDAP search filter to be used by the new SSO server. The installer prepopulates this with a default search filter. LDAP search filters function within a framework. The framework includes what attributes you are searching on and the value or range of values that you are trying to match. Each search filter involves at a minimum of three components: ■ The attributes to search for, called the attribute data type. ■ The search filter operator that determines what to match, sometimes called the match operator. ■ The actual value of the attribute you are searching for. Each search needs to have a minimum of one of each of the components. You can create compound search filters by connecting two or more search filters modules. They are enclosed in parentheses to clarify filter content, and include one or more of three compound search filter operators (AND, OR, NOT). You can add as many compound and wildcard filters as needed, as long as you have the correct number of matching parentheses. The actual search filter in the case of Microsoft Active Directory (Domain Controller) should look like: `(&(objectClass=user)(sAMAccountName={0}))` where `{0}` is substituted by the actual user name that is logging in.

Base Authentication Method	SSO Parameter Required	Description
Light Directory Access Protocol (LDAP) (continued)	Search Filter (continued)	Please see LDAP RFC 4515 https://opends.dev.java.net/public/standards/rfc4515.txt for more information about LDAP search filters and a mechanism for representing them as strings.
	Bind User DN	The LDAP bind user DN to be used by the new SSO server. By default, the installer prepopulates this parameter field with the LDAP value supplied earlier for the CAC setup software.
	Password	The LDAP password to be used in conjunction with the bind user DN by the new SSO server. By default, the installer prepopulates this parameter field with the LDAP value supplied earlier for the CAC setup software.

Configuration Entries for SSO with CAC Authentication

Provided the Dimensions CM installer is used to:

- install a new SSO server and configure your CAC authentication details, or
- nominate an existing SSO server and configure your CAC authentication details

the Dimensions CM server `dm.cfg` file is automatically configured by the installer with the requisite symbols (and default values) for use with SSO plus CAC authentication. If any of these values require editing, you need to restart the Dimensions CM server for those edits to take effect.

The SSO symbols automatically added to `dm.cfg` by an SSO server only installation are described in "Configuration Entries for SSO Authentication" on page 470. These, of course, are also required for an SSO plus CAC installation.

The following sample shows the additional SSO symbols that the Dimensions CM installer automatically adds to `dm.cfg` together with explanations for each parameter:

```
########################################
# Smart Card (CAC) Config
########################################
# CM Server X.509 certificate in PEM format that is used for signing requests to IDP
# IDP must trust to this certificate to successfully issue SSO Tokens
# This certificate is provided for demo purposes and must be regenerated in real
# environment and # and correspondingly after this it should be added to IDP trust store
# if $$DMSECURE$$ is specified in password then password should be specified using
# DMPASSWD for SSO_SERVER_PRIVATE_KEY user
SSO_SERVER_CERTIFICATE                          %DM_DFS%cm.pem
SSO_SERVER_PRIVATE_KEY                          %DM_DFS%cm.pem
SSO_SERVER_PRIVATE_KEY_PASSWORD                 serena
#SSO_SERVER_PRIVATE_KEY_PASSWORD                $$DMSECURE$$
```

- `SSO_SERVER_CERTIFICATE`

The location of the PEM format demonstration server certificate.

- SSO_SERVER_PRIVATE_KEY

  The location of the PEM format demonstration server private key.

- SSO_SERVER_PRIVATE_KEY_PASSWORD

  Password for PEM format demonstration server private key.

IMPORTANT! If you install only the SSO component to work in conjunction with SSO and CAC located on an Solutions Business Manager (SBM) installation, then the above entries need to be manually added to the Dimensions CM server dm.cfg file.

Not Allowing Accounts for CAC-Only Authorized Users

Dimensions CM supports dual username/password and CAC authentication for certain power users, for example, administrators and those who require the running of unattended batch jobs.

For all other CAC users, it is often company best practice or mandated policy to ensure that such users do not have optional access to username/password authentication. In such circumstances, the operating system administrator should either:

- never assign such users username/password authentication in the first place (the recommended option); or
- ensure that username/password authentication is removed from all normal CAC users who happen to have such authentication (for example, users with usernames that existed before CAC authentication was introduced).

Implementing CAC Support after Installing CM Server and SSO

To implement CAC authentication support after installing Dimensions CM with SSO, complete the steps below.

1. Open

   <TOMCAT_HOME>\webapps\idp\WEB-INF\conf\fedsvr-core-config.xml

   in an XML or text editor.

2. Locate the AllowedPrincipalAuthenticationTypes parameter and add CLIENT_CERT to it. This enables the **Smart Card Login** button. The resulting parameter looks like the following:

   ```
                  <parameter name="AllowedPrincipalAuthenticationTypes"
     Type="xsd:string">CLIENT_CERT</parameter>
   ```

3. Save the fedsvr-core-config.xml file.

4. Open:

   <TOMCAT_HOME>\webapps\idp\WEB-INF\conf\Configuration.xml file

   in an XML or text editor.

5 Uncomment the required X.509 authenticators by removing the `<!--X509-NAME and X509-NAME-->` markup from them. For example, remove the following markup to uncomment the `X509-BASE`, `X509-LDAP`, or `X509-CRL` authenticator, respectively.

```
<!--X509-BASE ... X509-BASE-->
<!--X509-LDAP ... X509-LDAP-->
<!--X509-CRL ...  X509-CRL-->
```

6 Configure the Certificate Authorities (CA) in the X509-BASE and X509-LDAP authenticators as described in "Configuring Trusted Certificate Authorities for SSO Installations" on page 475.

7 Note that for the X509-LDAP authenticator, the following parameters must be substituted:

```
$X509_LDAP_HOST
$X509_LDAP_USER
$X509_LDAP_PASSWORD
```

Note that by default, the installer configures the X509-LDAP authenticator when the CAC option is selected.

8 The `X509-CRL` authenticator can be used in addition to `X509-BASE` or `X509-LDAP`. In this case, the `$X509_CRL_PATH` parameter must be substituted and the specified folder must contain `*.CRL` files.

9 Save the `Configuration.xml` file.

10 Restart the SBM Tomcat Service.

The `Configuration.xml` file contains the following commented out example of an authenticator. If you want to use it you must remove the comments and substitute the variables appropriate to your set up:

```
!-- ============================================================  -->
<!-- CRL validator against file based Certificate Revocation List    -->
<!-- ============================================================  -->
<!--X509-CRL
          <!Setting Name="serena-crl-validator" Type="htf:map">
          <!Setting Name="Provider" Type="xsd:string">X509-CRL<!/Setting>
           <!Setting Name="CompatibleRequestMatchers" Type="htf:namedlist">
              <!Setting Name="CredentialsTypeMatcher" Type="xsd:string">X509<!/Setting>
              <!Setting Name="AuthenticationTypeMatcher" Type="xsd:string">*<!/Setting>
             <!Setting Name="CertificateIssuerDNMatcher" Type="xsd:string">*<!/Setting>
            <!/Setting>
            <!Setting Name="CRLDir" Type="xsd:string">$X509_CRL_PATH
           <!/Setting>
            <!Setting Name="CacheFileName" Type="xsd:string">crl_cache.xml<!/Setting>
            <!Setting Name="RefreshPeriod" Type="xsd:string">1200
                   <!/Setting>
          <!/Setting>
X509-CRL-->
```

Appendix B

Secure Sockets Layer Support

About SSL	492
Configuring CM to use SDP over SSL	494
Configuring Web Tools for HTTPS	498
Initializing the Random Number Generator	498
Troubleshooting	498

About SSL

Secure Sockets Layer (SSL) is a networking protocol that provides encrypted communications on the Internet. SSL is layered beneath application protocols such as HTTP, SMTP, Telnet, FTP, Gopher, and NNTP and is layered above the connection protocol TCP/IP. It is used by the HTTPS access method.

SSL is used to ensure secure transactions between web servers and browsers. The protocol uses a third party, a Certificate Authority (CA), to identify one end or both ends of a transaction:

- A web browser requests a secure page via `https://`.
- The web server sends its public key with its certificate.
- The web browser checks that the certificate:
  - Was issued by a trusted party (usually a trusted root CA)
  - Is still valid
  - Is related to the site contacted.
- The web browser uses the public key to encrypt a random symmetric encryption key and send it to the web server together with the required encrypted URL (as well as other encrypted `http` data).
- The web server decrypts the encrypted symmetric encryption key using its private key, and uses the decrypted symmetric key to decrypt the URL and http data.
- The web sever sends back the requested HTML document and http data encrypted with the symmetric key.
- The web browser decrypts the http data and HTML document using the symmetric key and displays the information.

SSL Certificates

An SSL certificate contains:

- Information about the owner of the certificate, for example:
  - Owner’s email address.
  - Owner's name.
  - Certificate usage.
  - Duration of validity.
  - Resource location.
  - Distinguished Name (DN), which includes:
    - The Common Name (CN) (web or email address depending on the usage).
    - The certificate ID of the person who certifies (signs) this information.
- The public key.
- A hash to ensure that the certificate has not been tampered with.

As you made the choice to trust the person who signs this certificate, therefore, you also trust this certificate. This is a certificate trust tree or certificate path.

Usually your browser or application has already loaded the root certificates of well known Certification Authorities (CA) or root CA Certificates (like `VeriSign` or `Thawte`). The CA maintains a list of all signed certificates as well as a list of revoked certificates. A certificate is insecure until it is signed, as only a signed certificate cannot be modified.

A certificate that is used in an SSL connection is considered as trusted when:

- It was issued by a trusted Certificate Authority.
- It is used by a site (domain name) for which it was issued.
- Its certificate is not expired.
- Its certificate (or its issuers tree) is not revoked.

Validating SSL Certificates

NOTE CM uses the Standard Dimensions Protocol (SPD) or the Secure Standard Dimensions Protocol (SSDP). For details, see the *Dimensions CM online help*.

Dimensions CM performs SSL certificate validation. Certificates used in the *Standard Dimensions Protocol (SDP) over SSL* and `https` connections must be:

- Trusted
- Not expired
- Have a valid host name.
- Have a proper usage type (if specified), for example:

  ```
  1.3.6.1.5.5.7.3.1 Server Authentication
  ```

You can use various methods to configure what is trusted by the SSL client. With *SDP over SSL* the desktop and web clients are SSL clients.

When a CM server connects to a remote node using *SDP over SSL* or tries to obtain an SSO token, generate an ALF event, or start a build job via https, it also represents an SSL client that should successfully validate the certificate used by the remote node or web application.

- For non Java applications on all platforms you can configure a `.pem` based certificates store, for example, add the following variable to the CM configuration file, `dm.cfg`:

  ```
  DM_SSL_TRUST_CERTS_FILE %DM_DFS%cacerts.pem
  ```

  The .pem file is a text file that contains multiple BEGIN and END CERTIFICATE sections. A clean installation of CM contains an example file with a CA of the SSL certificate used by OpenText Common Tomcat.

- On UNIX systems you can use a certificate hash folder, for example:

  ```
  DM_SSL_TRUST_HASH_DIR /etc/ssl/certs
  ```

  NOTE: The path to the certificates hash folder is not unified and each UNIX system can use a different path.

- On Windows the system certificates store is used.

  TIP: To display the store for the current user run: `certmgr.msc`

- For a Windows domain based environment and a domain based CA, additional configuration should not be required if certificates from the domain CA are used for the SSL (all workstations should trust it automatically).
- By default on all platforms for non-Java applications, OpenSSL is used to validate certificates. On Windows you can switch to a Windows based validation by specifying this parameter:

  ```
  DM_SSL_VALIDATION_FACILITY Windows
  ```

  The following parameter does not work in Windows mode:

  ```
  DM_SSL_TRUST_CERTS_FILE
  ```

- For Java-based applications, including web applications on OpenText Common Tomcat, the default Java trust store is located in:

  ```
  [JRE]/lib/security/cacerts
  ```

  OpenText Common Tomcat uses the following location:

  ```
  /dimensions/<version>/common/jre/<jre-version>/lib/security/cacerts
  ```

- Self-signed certificates for *SDP over SSL* (see below) can be used but must be explicitly trusted by the clients and contain a valid CN that is equal to the host name of the server that is being accessed.

Configuring CM to use SDP over SSL

To use SSL encrypted network connections, Dimensions CM requires a `X.509` server certificate.

You can request a certificate from your IT department or a Certificate Authority. You can also create a certificate using OpenSSL, which is distributed with the CM server software.

To generate a self-signed certificate for using SDP over SSL and configure CM to use it:

1. Navigate to the following folder:
   - **Windows:** `%DM_ROOT%\prog`
   - **UNIX:** `$DM_ROOT/prog`
2. Enter each of the following commands on a single line:
   - `set OPENSSL_CONF=%DM_ROOT%\prog\openssl.cnf`
   - `openssl req -config openssl.cnf -newkey rsa:2048 -sha256 -keyout serverkey.pem -out serverreq.pem`

     This command prompts you for the inputs required to generate the self-signed certificate and associated keys.

     You should be familiar with the input terminology and have the appropriate information. The CN name on this certificate should be the host name of the server or the fully qualified server name.

Example inputs:

PEM pass phrase	`serena`
Country Name (2-letter code)	`US`
State or Province Name	`Oregon`
Locality Name	`Hillsboro`
Organization Name	`OpenText`
Organizational Unit Name	`Support`
Common Name	`dimssoserver`

Enter the following additional attributes to send with your certificate request:

- A challenge password.
- An optional company name.

- `openssl x509 -req -in serverreq.pem -sha256 -extensions v3_ca -signkey serverkey.pem -out servercert.pem -days 1825`

These commands generate a certificate with a 2048 bit key size signed by the SHA256 algorithm that does not expire for 1825 day (5 years).

UNIX:

```
cat servercert.pem serverkey.pem > server.pem
```

Windows:

```
copy /b servercert.pem + serverkey.pem server.pem
```

After the commands have run, the following files are placed in `$DM_ROOT/cm/prog`:

File	Description
`server.pem`	Contains public and private certificates in a PEM encoding format.
`servercert.pem`	Contains the server certificate (public).
`serverkey.pem`	Contains an encrypted PKCS#8 PEM format of the private key.
`serverreq.pem`	Contains an encrypted PKCS#10 certificate signing request (CSR).

3 To create the additional PEM file (with Diffie Hellman parameters that can be used in a cryptographic key exchange), enter the following command:

```
openssl dhparam -out dh2048.pem -check -text -5 2048
```

After the command has run, the following file is placed in `$DM_ROOT/cm/prog`:

`dh2048.pem`

The file contains a `dh` format encrypted key in 2048-bit format.

4 Create this folder:

- **Windows:** `%DM_ROOT%\CA`

- **UNIX:** $DM_ROOT/CA

5 Move the following files from $DM_ROOT/cm/prog to the /CA folder you created earlier:

```
server.pem
dh2048.pem
```

NOTES:

- These files are required for *SDP over SSL* and *SSDP over SSL*.
- You can remove these intermediate files: serverreq.pem and serverkey.pem
- You can move servercert.pem file to the same location in $DM_ROOT/CA and use it to configure local and remote clients.

6 Verify that $DM_ROOT/CA/server.pem has the correct values and date:

```
openssl x509 -in server.pem -text -noout
```

7 Add the following entries to listener.dat located in the dfs folder:

```
-ssl
-ssl_password $$DMSECURE$$
```

Dimensions CM uses this information to access the server certificate generated by OpenSSL. Run the following command to register the certificate in registry.dat:

```
dmpasswd -ssl_password -add -pwd <password>
```

where <password> is the password of the encrypted private key stored in:

```
$DM_ROOT/CA/server.pem
```

8 If your Dimensions CM client is not using Java, use OpenSSL to validate the certificates:

a Open this file (copied in a previous step): $DM_ROOT/CA/servercert.pem

b Copy the public certificate section (including the BEGIN and END CERTIFICATE sections).

c Open the file that is specified by the DM_SSL_TRUST_CERTS_FILE symbol in the dm.cfg server configuration file. By default, this file is:

```
$DM_ROOT/dfs/sample_cacerts.pem
```

d Add the information that you copied in step **a**. By default, sample_cacerts.pem contains a demo trusted certificate used by Tomcat for HTTPS. You can:

- Append your self-signed certificate.
- Replace the contents of the file with your CA certificates if the SSL certificates, for example, were generated using a domain based CA.

If sample_cacerts.pem does not exist, create it, and add a parameter to the CM configuration file (see "Validating SSL Certificates" on page 493).

9 Java web applications use the default Java trust store:

```
$JRE/lib/security/cacerts
```

where cacerts is a standard Java Key Store (JKS) file.

The $JRE used by Tomcat is located by default in:

```
/opt/opentext/dimensions/<version>/common/jre/<jre-version>/
```

Use the `keytool` command to add the certificate file, `servercert.pem`, to the cacerts file as a trusted certificate. The `keytool` command is located in the `bin` folder.

Run a command similar to this:

```
keytool -import -keystore $DM_ROOT/common/jre/<jre-version>/lib/
    security/cacerts -storepass changeit -file $DM_ROOT/CA/
    servercert.pem -alias <certificate host name or CN>
```

10 To configure Common Tomcat to use SSL, replace the old certificate and private key with the newly generated ones:

a Delete the old certificate:

```
keytool -delete -keystore /opt/opentext/dimensions/<version>/
    common/tomcat/<tomcat-version>/conf/sample-ssl.jks -alias
    tomcat
```

To delete the certificate, the command prompts you for the PEM pass phrase.

b Convert the certificate and private key to the `pkcs12` keystore format:

```
openssl pkcs12 -export -in $DM_ROOT/CA/servercert.pem -inkey
    $DM_ROOT/prog/serverkey.pem -out server.p12 -name tomcat
```

To convert the certificate, the command prompts you for the PEM pass phrase.

c Import the `pkcs12` keystore file into the Tomcat keystore:

```
keytool -importkeystore -deststorepass serena -destkeystore
    $DM_ROOT/common/tomcat/<tomcat-version>/conf/sample-ssl.jks
    -srckeystore server.p12 -srcstoretype PKCS12 -alias tomcat
```

To import the certificate, the command prompts you for the PEM pass phrase.

11 In the `dm.cfg` server configuration file, replace the value of the `DM_WEB_URL` symbol with the following URL:

https://<host_name>:8443

12 Restart both the Dimensions CM server and Common Tomcat.

NOTE

- You can use self-signed certificates for SDP over SSL but they must be explicitly trusted by the clients and contain a valid CN that is equal to the host name of the server that is being accessed.
- For a Windows domain based environment and a domain based CA, additional configuration should not be required if certificates from the domain CA are used for SSL (all workstations should trust it automatically). The Windows system certificates store can be accessed by running `certmgr.msc`.
- The SSL Client parameters in `dm.cfg` are:

```
DM_SSL_TRUST_CERTS_FILE %DM_DFS%cacerts.pem
DM_SSL_TRUST_HASH_DIR /etc/ssl/certs
DM_SSL_VALIDATION_FACILITY Windows
```

- Add the new certificates to the Eclipse plugin JRE cacerts file.

- If the listener is started with additional `listener.dat` entries, as described above in step 7, all clients connecting to this listener use SSL mode. All `dmlibsrv` processes spawned by the listener also use SSL mode. But any connections from `dmappsrv` or `dmlibsrv` processes on this node to `dmappsrv` or `dmlibsrv` processes on other nodes are unencrypted unless the other node's listener is also in SSL mode.

Configuring Web Tools for HTTPS

To configure web tools for HTTPS for the Tomcat and JRE installation supplied with Dimensions CM, obtain a certificate and add a secure connector entry to the `server.xml` file. For details, see the *Apache Tomcat documentation*.

NOTE There are other ways to configure Dimensions CM web tools for HTTPS, such as using an external web server, or an alternative web application container.

Dimensions CM includes an example certificate for HTTPS under Tomcat. This certificate was issued by '`Serena Software SampleCA-1`' CA with a specified host of '`localhost`'.

For full security you should generate your own certificate through your Certificate Authority (CA) for a specified host name (site). If not, your browser displays a security alert message.

CAUTION! Only use the example certificate in demonstration mode to see how the mechanism works.

Initializing the Random Number Generator

To improve SSL initialization performance, in Dimensions CM 14.x the default random number generator has changed from `/dev/random` to `/dev/urandom` on platforms that support this functionality. You can specify an alternative random number generator via the HRNGDEVICE environment variable. For example, on UNIX issue this command:

```
export HRNGDEVICE=/dev/random
```

Troubleshooting

Enable Logging

To enable SDP logging, add the following variable in the server `dm.cfg` file:

```
DM_SDP_TRACE C:\logs\
```

When SDP logging is enabled you can view information such as:

- Is SSO is enabled.
- Verification of SAML tokens

The Tomcat logs can also help you identify problems:

```
$Tomcat\logs\

$Tomcat\logs\sso-idp.log

$Tomcat\logs\sso-gatekeeper.log
```

URL Issues

Verify that the STS Endpoint is correct in these files:

- The server `dm.cfg`
- `$Tomcat\alfssogatekeeper\conf\gatekeeper-core-config.xml`

Incorrect Host Name

See Knowledgebase Solution S140718.

Certificate Issues

Certificate issues are the most common reason for problems. Typically, the problems are with certificates in browsers and the verification of keystores.

You can verify keystores and pem files with the keytool commands and/or the Keystore Explorer. The keytool command is located in one of the following directories:

```
$Tomcat_JAVA\jre\<jre-version>\bin
$JBOSS_JAVA\<jdk-version>\jre\bin
```

NOTE If a pem file is too large, the Keystore Explorer cannot read it.

The keystores and pem files are located on the Dimensions CM server:

```
$Tomcat\conf\sample-ssl.jks Password serena
$Tomcat_JAVA\jre\<jre-version>\lib\security\cacerts changeit
$Tomcat\alfssogatekeeper\conf\truststore.jks changeit
$Tomcat\alfssogatekeeper\conf\keystore.jks changeit
$Tomcat\webapps\idp\WEB-INF\conf\truststore.jks changeit
$Tomcat\webapps\idp\WEB-INF\conf\keystore.jks changeit
$DM_ROOT\cm\dfs\sts.pem changeit
$DM_ROOT\cm\dfs\cm.pem* changeit
$DM_ROOT\cm\dfs\cacerts.pem** changeit
```

*used when authenticating with SmartCards, CAC or PIV

**used when SSL is enabled

To verify certificates in Internet Explorer do the following:

1. From the Tools menu select Internet Options and then the Content tab.
2. Select Certificates.
3. Verify that:
   - Root is in the Trusted Root Certification Authorities tab.

- Intermediate Certification Authorities are in the Intermediate Certification Authorities tab.

Appendix C

Using a Non-Default Service Name and Port Number

Default Network Service Name and Port Number	502
Setting Non-Default Service Name and Port Number	502
Command-Line and Desktop Client Considerations	504
Remote Tertiary Node Considerations	505

Default Network Service Name and Port Number

By default, a Dimensions® CM server installation configures Dimensions CM to listen for client connections and communicate with remote nodes using:

- Network service name `pcms_sdp`
- TCP/IP port number 671.

This appendix explains how to configure Dimensions CM to use the following example values for:

- Network service name: `dm101sdp`
- TCP/IP and port number: 5000
- Host name: dmserver

CAUTION! Both the Dimensions CM `pdiff` standalone utility and the Dimensions Replicator command `replicator` utilize "local connection", that is, they expect the network service name to be the default `pcms_sdp`. These particular utilities do not function if you set an alternative network service name. Additionally, all sites involved in replication must use the same port number for Dimensions CM.

Setting Non-Default Service Name and Port Number

UNIX

1. Log in as `root` and shut down the Dimensions server by running the `$DM_ROOT/prog/dmshutdown` or `dm_control cm_stop` shell scripts.
2. Edit the `$DM_ROOT/dfs/listener.dat` file and ensure that the `-service` parameter is specified as follows:

   ```
   -service dm101sdp
   ```

3. Ensure that only one of the following conditions is met:

   The `$DM_ROOT/dm.cfg` file contains the following line:

   ```
   DM_SERVICE_DM101SDP_TCP 5000
   ```

   Or the `$DM_ROOT/dm.cfg` file contains the following line:

   ```
   DM_SERVICES_FILE $DM_ROOT/services.dm
   ```

   and that the `$DM_ROOT/services.dm` file exists, is readable by the Dimensions system administrator account, and contains the following line:

   ```
   dm101sdp 5000/tcp
   ```

   Or the network services database `/etc/services` contains the following line:

   ```
   dm101sdp 5000/tcp
   ```

NOTE If the Dimensions server host is configured as a NIC client, then it may be configured to use the NIS network services database, not the local network services database `/etc/services`. In that case, you must ensure that the NIS network services database contains the required service definition. Contact your network administrator for assistance.

4 Check that you have made the correct edits above, as follows:

a Log in as root and restart the Dimensions server by running the `$DM_ROOT/prog/dmstartup` or `dm_control cm_start` shell scripts.

b Run `dmcli` or a Dimensions desktop client on a remote Windows node, if available. In the login dialog box, specify `hostname:portnumber` for the **Server**. If you fail to connect, repeat Step 1 to Step 3 and carefully check your edits.

5 Launch the Administration Console:

a On the Administration Console login screen, enter the **User ID** and **Password** for the Dimensions system administrator (normally, `dmsys`), but in the **Connect to** field select **Manual...** from the list.

b Specify the **DB name** and **DB connection**.

c In the **Server** field, enter `dmserver:5000`.

d Log in to the Administration Console.

e Select:

`Distributed Development > Network administration > Network nodes`

f In the navigation pane, select the physical network node `dmserver`.

g In the content pane, in the **Network Object** section, click **New object** (**+**).

h In the **Assign Node Object** dialog box, enter the network object name: `dm101sdp`. From the **Protocol** list, select `SDP`. Optionally fill in the other fields. Click **OK**.

i Log out from the Administration Console.

6 If you use the Tomcat web application container provided by OpenText to run the Dimensions web client and Administration Console web applications, stop Tomcat as described in "Stopping Tomcat" on page 29.

7 Configure the Dimensions web applications to use the new TCP/IP port number `5000` as the default, see "Reassigning Default Web Client Connection Parameters" on page 31. When this is done, you can log in to these applications using the default **Sample database** entry in the **Connect to** field. n future, you do not need to select **Manual...** and specify `dmserver:5000` in the **Server** field.

8 If you use the Tomcat web application container provided by OpenText to run the web client and Administration Console, restart Tomcat as described in "Starting Tomcat" on page 27. Check that you can then connect as described in Step 7.

NOTE If you use your own web application container to run Dimensions web client and the Administration Console web applications, then you must restart them in order for the above changes to take effect.

Windows

1. Log in as a Windows local administrator and stop the Dimensions CM Listener service.
2. Edit the `%DM_ROOT%\dfs\listener.dat` file and ensure that the `-service` parameter is specified as follows:

   ```
   -service dm101sdp
   ```

3. Ensure that only one of the following conditions is met:
   a. The `%DM_ROOT%\dm.cfg` file contains the following line

      ```
      DM_SERVICE_DM101SDP_TCP 5000
      ```

   b. or, the `%DM_ROOT%\dm.cfg` file contains the following line

      ```
      DM_SERVICES_FILE $DM_ROOT\services.dm
      ```

      and that the `%DM_ROOT%\services.dm` file exists, is readable by the Dimensions CM system administrator account, and contains the following line:

      ```
      dm101sdp 5000/tcp
      ```

   c. or, the network services database `%SystemRoot%\system32\drivers\etc\services` contains the following line

      ```
      dm101sdp 5000/tcp
      ```

4. Check that you have made the correct edits above, as follows:
   a. Log in as a Windows local administrator and restart the `Dimensions CM Listener` Windows service.
   b. Run dmcli or a Dimensions desktop client on the local node or a remote node (if available). In the login dialog, specify `dmserver:5000` for the **Server:** field. If you fail to connect, repeat Step 1 to Step 3 and carefully check your edits.
5. Perform Step 5 to Step 8 on page 503.

Command-Line and Desktop Client Considerations

When you use `dmcli`, `dmpmcli`, and/or the Dimensions CM desktop client to connect to a Dimensions CM server that was configured to use a non-default TCP/IP port number, you must explicitly specify the port number as part of the value of either:

- The **Server:** text field in the Dimensions CM Login dialog box (for example, `dmserver:5000`).
- The `-hostname` command-line parameter (for example, `-hostname dmserver:5000`).

Remote Tertiary Node Considerations

If you want to access remote tertiary nodes from a Dimensions CM server that was configured to use a non-default TCP/IP port number, then you must ensure that either:

- the Dimensions CM listener on the remote tertiary node is configured to use the same TCP/IP port number as the Dimensions CM server, or
- a network connection has been defined between the Dimensions CM server node (client) and the remote tertiary network node (server) using the Dimensions CM SDP protocol and the network object corresponding to the TCP/IP port number used by the remote tertiary node, see the description on Step 5 on page 503 and modify as appropriate.

Appendix D

Replicator Utilities

Introduction	508
Dimensions Label Branch Utility	508

Introduction

In addition to the standard Dimensions Replicator applications, the following utility is also provided to facilitate establishing a comprehensive Replicator environment.

PRIVILEGES The user must have ADMIN_REPL privilege for the base database in order to execute these utilities.

Dimensions Label Branch Utility

Description

Use the Dimensions Label Branch utility to move items that are on the nameless branch to be placed in a named branch. For example, ACME30:FINAL_RELEASE.UNIX-C;4 is an item revision on the nameless branch whereas ACME30:FINAL_RELEASE.UNIX-C;emu#4 is an item revision on branch emu.

This facility has been provided to allow users who have not currently used the named-branch facility to enable replication using named branches.

Syntax

```
pm_label_branch <branch_name> -p <product_id> [-t <item_types>]
    -user <user-id> -pass <user-pswd> -host <server-name>
    -dbname <db-name>
```

Options

<branch_name>	Specifies the named branch to which items on the nameless branch are moved. If this branch does not exist, it is created.
-p	Product identity.
-t	Optional comma-separated list of item types that are affected by this utility. The list must be enclosed in double quotation marks (""), for example, "src,doc,dat".
-user	Specifies the operating system username of your account on the server.
-pass	Specifies the password of your operating system username account on the server.
-host	Specifies the host name on the server.
-dbname	Specifies your database identifier.

Restrictions

This utility can be used only to move items to a named branch that is not currently in use. For example, if the utility is launched to move items to the `emu` branch where items already exist on this branch, the utility exits with an appropriate error message.

Appendix E

Replicator Error and Warning Messages

This appendix alphabetically lists and describes the error and warning messages generated by Dimensions Replicator. Where appropriate it also gives the action that needs to be taken to resolve each error and warning. The messages are listed in lexicographical order.

Each message is documented in the following format.

Message Description	
Cause	The reason why the message has been generated.
Action	The action that needs to be taken to resolve the error.

`** Replication Error: Baseline failed to import.`

`Cause` Replicator encountered a problem while creating a baseline on the target system.

`Action` This message is accompanied with an error message giving the reason for the failure.

`** Replication Error: Baseline failed to replicate.`

`Cause` Replicator encountered an error while attempting to replicate a baseline.

`Action` Please see the accompanying errors for more information.

`** Replication Error: Cannot find a revision of <item_id> in the project.`

`Cause` Replicator encountered an error while creating the pedigree for a replicated item.

`Action` *This is a fatal error.* Please contact Support.

`** Replication Error: Item failed to import.`

`Cause` Replicator encountered an error while creating an item on the target system.

`Action` This message is accompanied with an error message giving the reason for the failure.

**** Replication Error: Item failed to replicate.**

Cause Replicator encountered an error while creating an item on the target system.

Action This message is accompanied with an error message giving the reason for the failure.

**** Replication Error: Item is currently extracted.**

Cause Replicator could not replicate an item to or from a replica as it is currently extracted.

Action Check in the item in question and retry the operation.

**** Replication Error: Phase 1.**

Cause Replicator encountered a problem during phase 1 of the import process.

Action This message is accompanied with an error message giving the reason for the failure.

**** Replication Error: Phase 2 <request_id> loading error on relations - deleting request.**

Cause Replicator encountered a problem during the loading of requests.

Action This message is accompanied with an error message giving the reason for the failure.

**** Replication Error: Unable to calculate LC details.**

Cause Replicator encountered an internal error calculating details on the lifecycle while creating an item on the target system.

Action This is caused by data inconsistencies in the database. Please contact Support.

**** Replication Error: Unable to find details in cache.**

Cause Replicator encountered an internal caching error while replicating an item to the target system.

Action Retry the operation. If the error persists, please contact Support.

**** Replication Error: Unable to find replica details.**

Cause Replicator encountered an internal querying problem while identifying replicas to replicate to.

Action Retry the operation. If the problem persists, please contact Support.

**** Replication Error: Unable to replicate item relationships.**

Cause Replicator encountered an internal problem querying the related items to an item or request and wrote this information to disk.

Action Retry the operation. If the problem persists, please contact Support.

**** Replication Error: Unable to replicate library files.**

Cause Replicator encountered an error while replicating item library files to the target system. *This error is produced when Dimensions on the target system cannot place an item in the target library.*

Action Check the status of your Dimensions service or Daemon and verify that you can create items on the target system.

**** Replication Error: Unable to replicate object history.**

Cause Replicator encountered an internal problem querying history of an item / design part / baseline / request and wrote this information to disk.

Action Retry the operation. If the problem persists, please contact Support.

**** Replication Error: Unable to replicate part relationships.**

Cause Replicator encountered an internal problem querying the related parts to an item / baseline / request and wrote this information to disk.

Action Retry the operation. If the problem persists, please contact Support.

-- Warning: This baseline has already been transferred.

Cause You are attempting to replicate a baseline that already exists on the replica.

Action None.

Cannot initialise Replicator.

Cause An error was encountered when attempting to start the replication service.

Action Ensure that you are logged in as the Tool Manager and re-attempt the operation.

Configuration parameter DM_REPLICATOR_SERVER specifies invalid file <filename> on host <hostname>.

Cause The Replicator server program as identified by the symbol DM_REPLICATOR_SERVER in the file dm.cfg does not exist.

Action Ensure that the file referenced is correct and has executable permission. If this file is not found, Replicator has not been installed properly and therefore must be re-installed.

Dimensions Network is not running - Unable to open the routing file.

Cause Dimensions Network has not been started on the target node.

Action Ensure that Dimensions Network has been started.

Dimensions Network: no route file on <node>.

Cause Dimensions Network has not been started on the target node.

Action Ensure that Dimensions Network has been started.

Error: A baseline with this specification already exists.

Cause You attempted to replicate a baseline to a replica using a specification which already existed on that replica.

Action Either delete the baseline on the replica, or rename the baseline you want to replicate on the primary.

Error: An error occurred attempting to delete the contents of project <project_id>

Cause An internal error was encountered while trying to delete the contents of the temporary project used for baseline replication.

Action Retry the operation. If the error still persists, please contact Support.

Error: An error occurred attempting to query the contents of project <project_id>

Cause An internal error was encountered while trying to query the temporary project used for baseline replication.

Action Retry the operation. If the error still persists, please contact Support.

Error: An error occurred attempting to create the baseline on the target site.

Cause An error occurred while creating the baseline on the replica site.

Action Please see the accompanying error messages for more details.

Error: An error occurred calculating the checksums for the temporary projects.

Cause PVCS® Replicator encountered an error while attempting to calculate the checksums on the temporary projects used for baseline replication.

Action Retry the operation. If the error persists, please contact Support.

Error: An error occurred resetting the replication log(s).

Cause A database error was encountered while updating the replication logs associated with the baseline replication.

Action Retry the operation. If the error still persists, please contact Support.

Error: An error occurred trying to clear <local | remote> temporary project(s).

Cause While attempting to replicate a baseline, an error occurred trying to purge the contents of a temporary project.

Action Rerun the replication process. If the error persists, please contact Support.

Error: An error occurred trying to delete <local | remote> temporary project(s).

Cause While attempting to replicate a baseline, an error occurred trying to delete the contents of a temporary project.

Action Rerun the replication process. If the error persists, please contact Support.

Error: A revision of this item has already failed to be replicated.

Cause Replicator attempted to create a revision of an item on the target system for which a previous revision had already failed to be created.

Action Check the transfer log created and ensure that previously reported errors for this item specification are resolved before retrying the transfer.

Error: Baseline <baseline_id> already exists.

Cause You are attempting to replicate a baseline that already exists on the replica.

Action None.

Error: Baseline <baseline_id> already exists on the replica.

Cause The baseline you are attempting to replicate already exists on the replica.

Action Replicate a different baseline.

Error: Baseline <baseline_id> does not exist on this primary.

Cause You attempted to replicate a baseline which does not exist on the primary.

Action Check the baseline you are specifying exists, and is a release baseline.

Error: Baseline <baseline_id> is not a release baseline.

Cause You attempted to replicate a non-release baseline.

Action Specify a release baseline to replicate.

Error: Cannot change to home dir for <username> on remote node.

Cause An error was encountered when trying to connect to the replication service on the target system.

Action Retry the operation. If the error persists, please contact Support.

Error: Cannot connect to <DMDB>.

Cause Replicator could not connect to the database specified by the -p command switch or, if the switch is not used, by the DMDB symbol.

Action Ensure that the database specified is correct and rerun the operation.

Error: Cannot find Item Type <PRODUCT_ID>:<TYPE_NAME>.

Cause Replicator attempted to create an item on the target system that did not have the required item type defined.

Action Use Administration Console to create the required item type against the relevant product on the target system, and then rerun Replicator

Error: Cannot find product part for <product>.

Cause Replicator attempted to create an item for a product that does not exist on the target system

Action Use Administration Console to create the product on the target system and rerun Replicator.

Error: Cannot find product part for product <product>

Cause Replicator attempted to create an item for a product that does not exist on the target system.

Action Use Administration Console to create the product on the target system and rerun Replicator.

Error: Cannot replicate a directory item.

Cause The replication process is trying to transfer a directory item. This operation is not supported.

Action None.

Error: Cannot write to file <log_file>.

Cause An error was encountered trying to create a log file.

Action Ensure that the file specification is correct and that the user has 'write' permission.

Error: Command submission error

Cause An error was encountered when trying to run a remote command on the target system.

Action Retry the operation of the error persists, please contact Support.

Error: Completion of connection to <node> has failed. Check status of Dimensions Network daemon on <node> and attempt reconnection.

Cause An error was encountered when trying to connect to the Listener on the target system.

Action Ensure that the Listener service is running on the target system using the correct socket number and rerun operation.

Error: Connect failed to <node> (<network_error_number>). Check status of Dimensions Network daemon on <node> and attempt reconnection.

Cause An error was encountered when trying to connect to the Listener on the target system.

Action Ensure that the Listener service is running on the target system using the correct socket number and rerun operation.

Error: Could not create a temporary project.

Cause Replicator encountered an error while attempting to create a temporary project for use with baseline replication.

Action Retry the operation. If the error persists, please contact Support.

Error: Could not create a temporary project on the replica.

Cause Replicator encountered an error while attempting to create a temporary project on the replica.

Action Please see the accompanying error messages for more details. If the problem persists, please contact Support.

Error: Could not determine the uid of the primary configuration on the replica node.

Cause Replicator encountered an internal error querying the database.

Action Retry the operation. If the problem persists, please contact Support.

Error: Could not find the details for lifecycle <lifecycle>

Cause Replicator tried to create an item on the target system where the item type had no lifecycle associated with it.

Action On the target system, use Administration Console to associate a lifecycle with the item type.

Error: Failed to change working directory to <directory>.

Cause Replicator tried to access a directory on the remote system that did not exist or was non-readable.

Action Ensure that on the remote target system that the directory specified by DM_REMOTE_FILE_AREA in the file dm.cfg exists and is readable.

Error: Failed to connect to site <node_name>: <database>.

Cause Replicator could not connect to the remote database site <node_name>: <database>.

Action Ensure that the remote site has a license for installed, has the Listener service running, and the remote node is accessible through TCP/IP.

Error: Failed to create temporary replication directory <directory>.

Cause Replicator tried to access a local directory that does not exist or is non-writable.

Action Ensure that on the local system, where is initiated, the directory specified by DM_REMOTE_FILE_AREA in the file dm.cfg exists and is writable.

Error: Failed to execute RPC.

Cause This is a generic error indicating that failed to perform a function on the remote target system.

Action Retry the operation. If the error persists, please contact Support.

Error: Failed to export baseline successfully.

Cause An error occurred while Replicator was attempting to transfer the baseline definition to the replica.

Action See accompanying errors for more detailed information. If the problem persists, please contact Support.

Error: Failed to import baseline successfully.

Cause An error occurred while Replicator was attempting to transfer the baseline definition to the replica.

Action See accompanying errors for more detailed information. If the problem persists, please contact Support.

Error: Failed to import baseline log successfully.

Cause An error occurred while Replicator was attempting to transfer the baseline definition to the replica.

Action See accompanying errors for more detailed information. If the problem persists, please contact Support.

Error: Failed to load replication log into database.

Cause Replicator failed to save the log of the replication process in the database.

Action This error is usually accompanied with another message detailing the reason why the load failed.

Error: failed to read initial information.

Cause Network error encountered.

Action Retry the operation. If the error persists, please contact Support.

Error: failed to read secondary information.

Cause Network error encountered.

Action Retry the operation. If the error persists, please contact Support.

Error: Failed to register all the parts for this baseline.

Cause While performing post validation checks on the baseline parts that were registered against a replicated baseline, Replicator detected that some parts seemed to be missing.

Action This is caused by data inconsistencies in the database. Please contact Support.

Error: Failed to run utility to import baseline.

Cause Replicator encountered an error while attempting to input the baseline details on the replica.

Action Rerun the replication process. If the error persists, please contact Support.

Error: Failed to run utility to import replication log.

Cause Replicator encountered an error while attempting to input the replication log.

Action Rerun the replication process. If the error persists, please contact Support.

Error: failed to write initial_data to new socket.

Cause Network error encountered.

Action Retry the operation. If the error persists, please contact Support.

Error: Filename <filename> contains one or more of the following invalid characters: <invalid_characters>.

Cause The filename being replicated contains characters that are invalid on the target operating system.

Action Change the project filename on the sender database to one that does not contain illegal characters and re-try replication.

Error: For baseline replication – you must specify a baseline to replicate.

Cause You attempted to perform a baseline replication without specifying a baseline_id.

Action Reissue the command using the `-baseline_id <baseline_id>` parameter.

Error: Item <Item_specification> already exists.

Cause The item that is being replicated already exists on the target system and was not created as a result of a previous replication from the sending system.

Action None.

Error: Library and user filenames validation failed.

Cause Replicator encountered an error while creating an item on the target system.

Action This message is accompanied with an error message giving the reason for the failure.

Error: Primary database <database_name> is not enabled for replication on the replica site.

Cause The primary database is registered on the replica site but has not been enabled for replication.

Action On the replica site use the Dimensions networking utilities to enable the replication switch for the primary base database.

Error: Primary database <database_name> is not registered on the replica database. The tool-manager of the replica database may register it using the Dimensions Network Administration Tool.

Cause The primary database has not been registered on the replica site.

Action On the replica site use the Dimensions networking utilities to register the primary base database site.

Error: No active replication was found.

Cause The replication configuration specified by the user exists but has not been activated for use.

Action Use the Administration Console or `dmpmcli` scripting interface to activate the replication configuration.

Error: Node <node> is not registered on this primary database. Please use the Dimensions networking utilities to register it.

Cause The physical node has not been defined on the primary database.

Action Use the Dimensions networking utilities to define the node on the primary database.

Error: Option -f cannot be used with Baseline replication.

Cause You attempted to perform a baseline replication using an option that only applies to item replication.

Action Reissue the command, removing the `-f` option.

Error: DM_REPLICATOR_SERVER not set on host <node>.

Cause The symbol `DM_REPLICATOR_SERVER` is not set in the `dm.cfg` file.

Action Ensure that the symbol is defined and its value points to the file `%DM_PROG%\replicator_server.exe` on Windows or `$DM_PROG/replicator_server` on UNIX.

Error: Remote function execution failed.

Cause An error was encountered trying to run the replication process on a remote target system.

Action Retry the operation. If the error persists, please contact Support.

Error: Replication configuration <replication_id> already exists on the replica database and the owner of this configuration is not the primary database.

Cause The replica database is being targeted by multiple primary base databases that are trying to use the same replication configuration id.

Action The primary base database attempting to replicate items to this replica must rename the replication configuration id so that the name is unique on the target replica.

Error: Replication configuration <replication_id> does not exist.

Cause The configuration id specified is invalid.

Action Ensure that Replicator is invoked with the correct replication configuration identifier.

Error: Replication configuration <replication_id> is not active.

Cause The replication configuration has not been activated through the *dmpmcli* scripting interface.

Action Use the Administration Console or `dmpmcli` scripting interface to activate the replication configuration identifier before re-attempting the operation.

Error: Replication could not load the details for the current database.

Cause The details on the primary base database have not been defined through the Dimensions Network Administration.

Action Ensure that the details on the primary base database have been registered through the Dimensions Network Administration.

Error: Replication Id must be specified.

Cause Replicator has been run without specifying a replication configuration id.

Action Use the –c switch to specify a replication configuration when running.

Error: Replication lower limit cannot be less than 1.

Cause Replicator has been invoked with the –m switch set to a number less than 1.

Action Invoke Replicator with the –m parameter being greater than 0.

Error: <service_name>/<network_protocol>, unknown service in TCP services file and DM_SERVICES_FILE symbol is not defined.

Cause The TCP/IP service specified in the Dimensions Network Administration has not been recognized.

Action Ensure that the service name specified in the Dimensions Network Administration has either been registered as a valid TCP/IP service, or that an appropriate entry has been made in the dm.cfg file. The format of these entries is as follows:

```
DM_SERVICE_<SERVICE_NAME>_TCP <socket number to use>
```

Error: <service_name>/<network_protocol>, unknown service in TCP services file or Dimensions Client services file <filename>. Check contents and location.

Cause The TCP/IP service specified in the Dimensions Network Administration has not been recognized.

Action Ensure that the service name specified in the Dimensions Network Administration has either been registered as a valid TCP/IP service, or that an appropriate entry has been made in the dm.cfg file. The format of these entries is as follows:

```
DM_SERVICE_<SERVICE_NAME>_TCP <socket number to use>
```

Error: <socket_name>/<network_protocol> cannot read Dimensions Client services file <filename>. Check permissions.

Cause The optional services file specified by the symbol DM_SERVICES_FILE in the dm.cfg file does not exist or is not readable.

Action Ensure that the file specified exists and has read permission.

Error: Replica database <database_name> is not enabled for replication on the replica site.

Cause On the target replica site the replica base database details have not been registered as 'replication enabled'.

Action Use the Dimensions Network Administration on the replica site to define the replica base database details as enabled for replication.

Error: Temporary file <filename> already exists.

Cause Replicator tried creating a temporary file that already existed.

Action Retry the operation. If the error persists, please contact Support.

Error: The -baseline_id parameter is only valid for baseline Replication Configurations.

Cause You attempted to replicate a baseline using an item replication configuration definition.

Action Reissue the command using a baseline replication configuration definition.

Error: The base database could not be updated for Replicator.

Cause The replication process is being initiated on a database that has not been upgraded for replication.

Action Install Replicator on the affected site and ensure that the database is upgraded to allow replication.

Error: The base database you are connected to is not the primary for the replication configuration specified.

Cause The user is trying to initiate a replication process on a replica database using a replication configuration that is owned by another primary database.

Error: The baseline failed to be created on the replica. Please check the replication logs for any errors.

Cause An error occurred while Replicator was attempting to transfer the baseline definition to the replica.

Action See accompanying errors for more detailed information. If the problem persists, please contact Support.

Error: The baseline specification <baseline_id> is invalid.

Cause The baseline_id you specified to Replicator has an invalid specification format.

Action Verify that the baseline_id you are attempting to replicate is valid.

Error: The library filename validation failed.

Cause Replicator encountered an error while creating an item on the target system.

Action See the accompanying error message for the reason for the failure.

Error: The primary database <database_name> is not enabled for replication.

Cause The primary database site has not been enabled for replication.

Action Use the Replicator Administration to ensure that the database site definition for the primary has been flagged as "Project Replication enabled".

Error: The number of items to be transferred at a time cannot be less than 1 (default is 100).

Cause Replicator has been invoked with –n less than 1.

Action Invoke Replicator with the –n parameter being greater than 0.

Error: The path name is too long.

Cause The project filename and directory specified are too long for the target operating system.

Action Ensure that the length of the project directory and filename conform to the standards of the receiving operating system.

Error: The site <site_name> is invalid.

Cause The site details have not been registered on the target system.

Action Ensure the site details as specified in the error message have been registered through the Dimensions Network Administration on the target system.

Error: The replica database <database_name> is not enabled for replication.

Cause The replica database site has not been enabled for replication.

Action Use the Replicator Administration to ensure that the database site definition for the replica has been flagged as "Project Replication enabled".

Error: The replica is still expecting items from baseline <baseline_id>

Cause You attempted to replicate a baseline using a replication configuration definition which is still being used to replicate another baseline.

Action Use a different baseline replication configuration definition, or complete the replication for the first baseline, and then use this configuration identifier.

Error: The replica Site does not have a registry entry for database <database_name>. The tool-manager of the replica database may register it using the Dimensions Network Administration Tool.

Cause The database name has not been registered on the replica system.

Action Use the Dimensions Network Administrationon the replica site to define the replica base database details as enabled for replication.

Error: The target project has had item(s) replicated to it, but the contents may have changed.

Cause As part of the baseline replication operation, a temporary project has been populated with the items necessary to create the baseline, but the contents of that baseline may have been modified by a Dimensions user prior to the baseline being created.

Action Manually delete the temporary project on the replica and rerun the replication process using the `-redo flag`.

Error: The user filename validation failed.

Cause The library filename details on the replicated item failed to be validated on the target system.

Action This message is accompanied with an error message giving the reason for the failure.

Error: The project filename and directory validation failed.

Cause The project filename details on the replicated item failed to be validated on the target system.

Action This message is accompanied with an error message giving the reason for the failure.

Error: There are no replica defined for replication configuration <replication_id>.

Cause The user attempted to replicate using a replication configuration that has no replicas defined for it.

Action Ensure that the replication configuration has at least one replica definition created.

Error: These options cannot be used with baseline replication.

Cause Replicator has detected the use of some parameters which are incompatible with baseline replication

Action Do not use the following options with baseline replication:

-sc (replicate requests from the secondary catalogue)

-dont_validate_user (do not perform any validation of OS user ids when loading requests)

-dont_replicate_items (do not replicate related items when performing request replication)

-dont_replicate_blns (do not replicate related baselines when performing request replication)

-dont_replicate_chdocs (do not replicate related requests when performing request replication)

-ignore_related_errs (ignore any errors that may result from missing related objects)

Error: These options cannot be used with request replication.

Cause Replicator has detected the use of some parameters which are incompatible with issue replication

Action Do not use the following options with issue replication:

-e do not propagate item removals from the source project to the target project.

-d disable integrity constraint checks on branch ownership at the primary site.

-dont_check_branches

Error: These options cannot be used with item replication.

Cause Replicator has detected the use of some parameters which are incompatible with item replication

Action Do not use the following options with baseline replication:

-sc (replicate requests from the secondary catalogue)

-dont_validate_user (do not perform any validation of OS user ids when loading requests)

-dont_replicate_items (do not replicate related items when performing request replication)

-dont_replicate_blns (do not replicate related baselines when performing request replication)

-dont_replicate_chdocs (do not replicate related requests when performing request replication)

-ignore_related_errs (ignore any errors that may result from missing related objects)

Error: This item's metadata can only be updated by the site which own this item.

Cause An attempt was made to update an item's metadata by a site which did not originally send the item.

Action None.

Error: Unable to add to correct project.

Cause An error was encountered trying to add the replicated item to the target replica or primary project.

Action Retry the operation. If the error persists, please contact Support.

Error: Unable to find replication details in cache.

Cause Replicator encountered an internal error while attempting to create a log of the replication process.

Action Retry the operation. If the error persists, please contact Support.

Error: Unable to import library file details.

Cause Replicator encountered an error while attempting to create an item on the target system.

Action This message is accompanied with an error message giving the reason for the failure.

Error: Unable to insert details into the database.

Cause A database error was encountered trying to create the replicated item on the target system.

Action This message is accompanied with an error message giving the reason for the failure.

Error: Unable to insert details into the database. Please check you have the appropriate role on the project to create the item, or check the Replicator error message guide for more details.

Cause While attempting to create the item in the database, Replicator encountered an error in either the permissions necessary to create a project directory, or while inserting a database entry.

Action Check that the user replicating the items has the role to create project directories on the replica, or see any accompanying Oracle errors previous to this message.

Error: Unable to insert history/relationship details.

Cause A database error was encountered trying to create the replicated item on the target system.

Action This message is accompanied with an error message giving the reason for the failure.

Error: Unable to insert number of items to replicate into database.

Cause encountered an internal error querying the database.

Action Retry the operation. If the problem persists, please contact Support.

Error: Unable to populate the baseline.

Cause An error occurred while trying to create the baseline on the replica.

Action Please see accompanying error messages for further information.

Error: Unable to query the request details for <number>.

Cause Replicator encountered an internal error querying the database.

Action Retry the operation. If the problem persists, please contact Support.

Error: Unable to query the request mode details for <number>.

Cause An internal error was encountered while trying to query the details of a request.

Action Retry the operation. If the error persists, please contact Support.

Error: Unable to read RPC response message.

Cause A network error encountered.

Action Retry the operation. If the error persists, please contact Support.

Error: Unable to set $ACTION records.

Cause Replicator encountered an internal error while attempting to create an item on the target system.

Action This message is accompanied with an error message giving the reason for the failure.

Error: Unable to set $UPDATE records.

Cause Replicator encountered an internal error while attempting to create an item on the target system.

Action This message is accompanied with an error message giving the reason for the failure.

Error: Unable to set related objects records.

Cause Replicator encountered an internal error while attempting to create an item on the target system.

Action This message is accompanied with an error message giving the reason for the failure.

Error: Unable to set the object data.

Cause Replicator encountered an internal error while attempting to create an item on the target system.

Action See the accompanying error message for the reason behind the failure

Error: Unable to update replicator update mode details.

Cause While attempting to update the details associated with an item, Replicator encountered a database error.

Action Retry the operation. If the error persists, please contact Support.

Error: Unable to verify the existence of the temporary project on the primary.

Cause Replicator encountered an error while attempting to create a temporary project on the primary.

Action Please see the accompanying error messages for more details. If the problem persists, please contact Support.

Error: Unable to verify the existence of the temporary project on the replica.

Cause Replicator encountered an unknown error while attempting to verify the existence of a temporary project on the replica site.

Action Retry the operation using the `-u` flag. If the error persists, please contact Support.

Error: Unknown host <node>.

Cause One of the replica hosts has not been registered in the hosts file.

Action Ensure the host name is correct and is accessible through ping from the node on which replication is being initiated.

Error: Validating the $ACTION records failed.

Cause Replicator encountered an error while attempting to create an item on the target system.

Action This message is accompanied with an error message giving the reason for the failure.

Error: Validating the $UPDATE records failed.

Cause Replicator encountered an error while attempting to create an item on the target system.

Action This message is accompanied with an error message giving the reason for the failure.

Error: Validating the object failed.

Cause Replicator encountered an error while attempting to create an item on the target system.

Action This message is accompanied with an error message giving the reason for the failure.

Error: Validating the related object records failed.

Cause Replicator encountered an error while attempting to create an item on the target system.

Action This message is accompanied with an error message giving the reason for the failure.

Error: Variable DMDB is not defined.

Cause The symbol `DMDB` is not defined in the current environment or in the `dm.cfg` file.

Action Ensure that the symbol is defined.

Error: Variable DM_REMOTE_FILE_AREA is not set in the configuration file.

Cause The `dm.cfg` file does not have the symbol `DM_REMOTE_FILE_AREA` set.

Action Ensure that the symbol is defined.

Error: Project <project> does not exist.

Cause The primary project name specified in the replication configuration does not exist.

Action Ensure that the project name specified in the replication configuration is correct.

Error: Project <project> does not exist on the replica.

Cause The replica project defined as part of the replication configuration currently being processed does not exist.

Action Ensure that the replica project names in your replication configuration are correct. If necessary, ask the manager on the replica system to create the target project.

Error: You are not a tool-manager for the primary base database.

Cause You do not have ADMIN_REPL princeling.

Action None.

Failed reading message.

Cause Network error encountered.

Action Retry the operation. If the error persists, please contact Support.

Failed to commit log data to the database.

Cause Encountered a database error while trying to log the replication process in the database.

Action Retry the operation. If the error persists, please contact Support.

Failed to determine if primary or replica configurations changed since previous transfer.

Cause Replicator encountered an error while trying to determine if the replication configuration details on target systems need to be updated.

Action Retry the operation. If the error persists, please contact Support.

Inherit handle failed.

Cause Network error encountered.

Action Retry the operation. If the error persists, please contact Support.

Invalid socket id <socket_id>.

Cause Network error encountered.

Action Retry the operation. If the error persists, please contact Support.

Invalid project id <project_id>.

Cause The syntax of the project identifier defined in the replication configuration for the primary is invalid.

Action Ensure that the project syntax of the primary project is correct.

Message loop failure.

Cause Network error encountered.

Action Retry the operation. If the error persists, please contact Support.

Received connection request for protocol version.

Cause There is a mismatch in the versions of and supporting installations on the primary and replica nodes.

Action Ensure that the affected nodes have the correct software installed.

Replicator **is already running.**

Cause The TCP/IP socket specified in the file *dm.cfg* is already in use.

Action Ensure that the replication service is using a unique TCP/IP socket id or contact your Support for more information.

<service_name>: Failed starting server '<server_name>' on host <node>.

Cause The server failed to start on the target node.

Action Retry the operation. If the error persists, please contact Support.

Warning: Found <no> attributes which are not defined on the recipient. The attributes are not replicated.

Cause Items which are being replicated have user-defined attributes associated with them which are not defined on the recipient system.

Action Use the Administration Console to create the appropriate item attribute definitions on the recipient system to ensure successful replication of the attribute definitions associated with an item,

Warning: Ignored on unknown replica definition.

Cause The base database site which was setup as the basis for this replica has been deleted through the Dimensions Network Administration.

Action Use the Replicator Administration utility to redefine this base database.

Warning: Primary project <wset spec> is locked. There will be no attempt to transfer items from replicas to this project.

Cause Replicator cannot import items into a locked project.

Action Unlock the project on the primary system and retry the operation.

Warning: query failed for object specification <object spec.>.

Cause Replicator encountered an internal error querying the database.

Action Retry the operation. If the problem persists, please contact Support.

Warning: Replica database <database_name> is not active.

Cause The replica definition specified has not been activated.

Action Activate the replica definition using the **dmpmcli** scripting interface.

Warning: Unable to create attribute definitions.

Cause The user-defined attributes of the replicated item could not be created on the target system.

Action Ensure the attribute definitions on the target system are the same as the sending system.

Warning: will not back replicate branch <branch name> from replica <replica name> because the replica does not own this branch in the primary database.

Cause The replica is trying to replicate back a branch that is registered on the primary as being owned by a site other than the replica. A branch can only be replicated to a site if that sender owns the branch on the recipient database.

Action Replication of this branch is not allowed, use a different branch.

Warning: Will not back replicate branch <branch id> from replica <site id> because the replica does not own this branch in the primary database.

Cause Replicator has identified a clash in branch ids.

Action Either branch must be renamed in order to replicate this configuration across sites. Please contact Support for instructions to rename version branches.

Warning: Will not replicate branch <branch id> to replica <site id> because the primary does not own this branch in the replica database.

Cause Replicator has identified a clash in branch ids.

Action Either branch must be renamed in order to replicate this configuration across sites. Please contact Support for instructions to rename version branches.

Warning: will not back replicate branch <branch name> to replica <replica name> because the primary does not own this branch in the replica database.

Cause The primary is trying to replicate a branch that is registered on the replica as being owned by a site other than the primary. A branch can only be replicated to a site if that sender owns the branch on the recipient database.

Action Replication of this branch is not allowed, use a different branch.

Appendix F

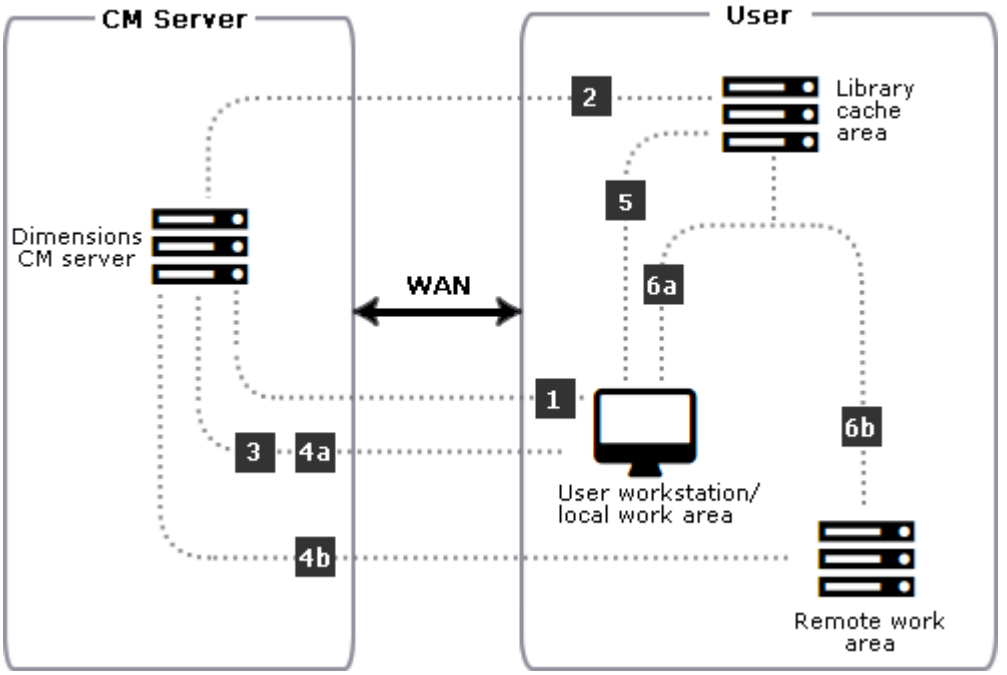
Using and Configuring Library Cache Areas

Introduction	538
Example Scenario	540
When to use Library Cache Areas	542
Configuring and Using Library Cache Areas	544

Introduction

Software development teams are often geographically dispersed with remote users connecting to a main site across a Virtual Private Network (VPN) utilizing a Wide Area Network (WAN). However, VPN connections over a WAN typically have low bandwidth and suffer from a high latency. To address this problem, a library cache mechanism is available that improves performance by caching file contents on a network node that is geographically close to the users' system. Consequently, remote users do not need to repeatedly transfer files over a slow connection from an item library node to their local machines.

The following diagram illustrates how you can configure library cache areas to optimize performance.



Step	Description
1	A remote user updates their work area.
2	The Dimensions CM server checks the status of the items to determine if the latest item revisions to be retrieved already exist in the library cache area. ■ If a copy of the requested file is not present, the server copies it to the library cache area. The initial copy to the library cache area may take more time than a direct copy from the item library as it requires the files to be extracted from the item library. Some files are not copied to the library cache area and continue to be retrieved from the item library. These include delta files, non-static files, and files of a specific size. ■ If a copy of the requested file is present, the server proceeds to the next step and there is no further communication with the library cache area. After an item library file is copied to a library cache area, it remains there until it is manually deleted. Library cache areas are not purged automatically.
3	The server sends the user's client information about which files to fetch from the library cache area.
4a	If the user's work area is local, the server copies files that are not in the library cache area from the item library to the user's client.
4b	If the user's work area is remote, the server copies files that are not in the library cache area from the item library to the remote work area.
5	The user's client requests the required cached files from the library cache area.
6a	If the user's work area is local, the server copies the files from the library cache area to the local work area. The performance may significantly benefit from use of the library cache area. To further improve performance the files are compressed prior to retrieving them. If the user performed the update from the web client to a local work area, Dimensions CM bypasses the library cache area.
6b	If the user's work area is remote, the server copies the files from the library cache area to the remote work area.

Example Scenario

Overview

The Qlarius development team is distributed across multiple sites in the United States, the United Kingdom, and India. The team uses a high-speed network connection to share files across these sites, and use Dimensions CM to manage processes, change, and configuration.

The development teams primarily use the Visual Studio integration with CM to update and deliver on a daily basis. The teams use an "optimistic" locking method; they are not required to lock files that they are working on. The files are a combination of compiled binaries and source code files. They do not use item header substitution when getting files.

Workflow

The Qlarius development team plans to use library cache areas to improve the overall performance and has the following workflow:

1 The US team checks their changes into the CM repository at the end of their work day. The repository is located in the US and performance is very good, particularly for the users co-located with the repository. The US team does not need to use library cache areas.

2 The first person to start work in the morning in the United Kingdom office gets the latest files using the Visual Studio integration. A library cache area is implemented for this site. Any files that are not already in the library cache area are copied there. This ensures that any changes that the US team checked in during their previous work day are populated to the UK library cache.

3 As each of the other team members start their work days, they also update the files using the Visual Studio integration. Because the latest versions already exist in the library cache area, Dimensions CM does not have to transfer the files from the item library across the low performance WAN before copying them locally. Consequently, performance is improved.

4 At the end of the day, team members deliver their changes. If any of their local changes conflict with other users' changes, they merge the different versions of the files to resolve the conflicts before checking in.

Configuring the Environment

The Qlarius development team's configuration includes a UNIX server and Windows clients. The server and clients meet the system requirements for Dimensions CM and their RDBMS.

Network Configuration and Latency

The Qlarius network configuration includes 4 E1 lines with a 150 millisecond latency and a deviation of + or – 5 millisecond.

NOTE E1 is the European format for digital transmission and is similar to the North American T1. E1 carries signals at two Mbps (32 channels at 64Kbps, with two channels reserved for signaling and controlling). T1 carries signals at 1.544 Mbps (24 channels at 64Kbps). E1 and T1 lines may be interconnected for international use.

Projects and Files

The Qlarius development team are working on a number of projects. These typically include a mixture of large and small source files. The following table summarizes the data set that the team tested performance against:

# Files	# Folders	Total size	Mean file size	Median file size
1,634	218	16 MB	9.75 KB	4 KB

There are no delta files or files with header substitution.

The following table summarizes the specific file size.

Size	No of Files
< 1KB	347
> 2KB	245
< 4KB	322
< 20KB	590
< 1MB	128
>= 1MB	2

Performance Improvements

Because the UK-based team has medium to high latency on their WAN, and mean file sizes as high as 10 KB, they implemented library cache areas in most (but not all) of their projects. Specifically, they decided to:

- Implement library cache areas for their Visual Studio projects, which have a mixture of very large resource files and small files.
- Set the minimum file size to copy into the library cache area to 4 KB.
- Disable delayed TCP acknowledgments (ACKs) and the Nagle algorithm to significantly improve performance.

After implementing library cache areas, the UK team measured the performance difference both with and without library cache areas, and found that projects that met the above criteria yielded significant performance improvements.

The team also determined that:

- Using library cache areas provides little or no benefit for projects that include very small source code files only, for example 2K or smaller. The cost of populating the library cache areas and copying them to the local work area is higher than copying them directly.
- The US-based team does not benefit from using library cache areas for projects that are already hosted in Dimensions CM repositories located on their LAN.

When to use Library Cache Areas

Consider the following environmental factors when deciding whether to use library cache areas.

Library Cache Area User Types

- Users with the appropriate network and library cache privileges (for example, the Dimensions CM administrator). These users are responsible for:
  - Initially setting up library cache areas for use by users.
  - Administering those library cache areas.

  NOTE The privileges required for these tasks are specified in the *Command-Line Reference* for the commands `CNN` and `UNC` (network privileges) and `CLCA`, `LLCA`, `PLCA`, `RLCA`, and `ULCA` (library cache privileges).
- Normal Dimensions CM users. Each Dimensions CM user must associate their current project or stream with a particular library cache area before they can make use of the library cache mechanism.

Network Latency

Latency is one of the key factors in deciding to use library cache areas or not. Typically, the higher the latency, the more likely that library cache areas improve performance. To find out your network latency, on a client machine of average performance or slower, use the `ping` command. The following table gives latency category guidelines.

Category	Value
High	200ms
Medium	100ms
Low	<1ms

Server Proximity

If performance is poor with a server that is located nearby, library cache areas do not improve performance.

Project Characteristics

The performance improvement using library cache areas also depends on the types of files contained in the project or stream. Not all files are cached, for example, files with item header substitution, delta files, and very small files. The greater the number of large files (such as tar, zip, or Visual Studio resource files) the bigger the benefit of using library cache areas. Smaller files are excluded from the library cache area as associated overheads do not make it worthwhile. You can configure the excluded file size. The default is 2K though setting it to 4k or higher has better results.

Your project distribution and usage is important. For example, if a project had 98% of its files under 25K and the other 2% averaged over one megabyte, library cache usage may make a significant difference when downloading the entire project. However, if the large files do not change very often, library cache usage does not help much when downloading changed files because only the very small files are downloaded.

Although library cache areas may be useful for some projects and not others, there is no need to take measurements for all projects. You can group similar projects together and test a sample project from each project group.

File Prerequisites

Even if library cache areas are enabled, certain file types are not copied and are retrieved directly from the item library, including:

- Directory item library files.
- Library item files with header substitution turned on. Because files are dynamic, their content differs between the item library and the library cache.
- Delta library item files.
- Compressed item library files.
- Small library item files below a default size of 2K.

If most of your library item files are in these categories, you are unlikely to obtain any benefit from enabling library cache areas.

All other types of item library files can utilize the library cache area mechanism and gain an improvement in performance.

Work Area Requirements

Depending on the CM client that you use, a library cache may improve performance if:

- You access the server over a Wide Area Network.
- You check files out to the type of work area (remote or local) indicated in the following table:

	Work Area	
Client	**Local**	**Remote**
Desktop	X	X
Web client	-	X
dmcli	X	X
Eclipse integration	X	X
Visual Studio integration	X	X
Windows Explorer integration	X	X

For example, if your users primarily use the Windows Explorer integration to refresh local work areas with the latest versions of files, library cache areas improve performance when getting and checking out files. If your users mostly use the web client with local work areas, then the library cache feature does not yield performance benefits.

Configuring and Using Library Cache Areas

NOTE For information about improving and optimizing library cache performance, see the *online help*.

IMPORTANT!

- WAN performance improvements gained by enabling library cache areas vary between installations. The setup must be configured to maximize throughput; in some cases, the inappropriate use of library cache areas may decrease performance.
- Dimensions CM 14.x is not backwards compatible with library cache areas that have earlier versions of CM installed. You should upgrade all library cache nodes to Dimensions CM 14.1 or later.

Dimensions CM Privileges

Dimensions CM privileges apply when using library cache areas. Users' access to projects, streams, and items are controlled by privileges the same as they are when library cache areas are not in use. For details, see the *online help*.

Dimensions CM Listener

Library cache areas can run on a network node that has the Dimensions CM listener installed on it. The listener is installed by performing a Dimensions CM Agent install. The Listener Service is required on each system hosting a build area and does the following:

- Authenticates users when they connect to Dimensions CM.
- Acts as a connection pool manager for Dimensions CM application servers for accessing local or remote Dimensions CM databases.
- Starts Dimensions CM user file access server processes that perform file and build operations.
- Starts item library server processes that access item libraries.

Adding a Physical Network Node

Before creating a library cache area, add the system to host the library cache to Dimensions CM as a physical network node.

NOTE In the following circumstances you also need to define a network node connection (see "Network Node Connections" on page 298) between the Dimensions CM server and the library cache area:

- The Dimensions CM listener on the system where you define the library cache area uses a non-standard port number.
- The library cache area is on a mainframe system. You need to specify a codepage to apply during file transfers.

PRIVILEGES To add a node you need the following privilege:
Administration Privileges | Other Admin | Manage Network Definitions

1. As a user with the appropriate privilege, log in to the Dimensions CM Administration Console and select Distributed Development | Network Administration.
2. Select the Network Nodes tab.
3. Click New.
4. Select Physical Network Node.
5. Specify a Physical node name, Operating system, Contact, and Description fields.

Installing the Dimensions CM Listener

For instructions about installing the Dimensions CM agent on your Windows or UNIX machine, see the CM installation guides.

Starting the Dimensions CM Listener

The Dimensions CM listener must be started on every machine that:

- Provides access to a Dimensions CM database and to a item libraries hosted locally on that node.
- Performs file operations in user work areas.

The Dimensions CM listener is started as part of the general Dimensions CM startup.

- UNIX: startup is performed by the following scripts:

  ```
  dmstartup
  dmshutdown
  ```

- Windows: startup is performed automatically through a Windows service.

The Dimensions CM listener is activated by the program `dmlsnr` and is configured to listen to a specific port to service client network requests.

For details about the command syntax of `dmlsnr`, see Chapter 10, "General Operating Guide".

To restart the Dimensions CM listener:

- On UNIX:
  - As the user "root", run the script `dmshutdown`.
  - Run the script `dmstartup`.
- On Windows: as a user with the appropriate privilege, restart the Dimensions CM listener service.

Creating Library Cache Areas

TIP If you are the Dimensions CM system administrator and other users have the privilege to create projects but not library cache areas, consider adding a trigger to notify you when projects and streams are created so that you can enable library cache areas.

PRIVILEGES Administration Privileges | Area Management | Create Library Cache Areas

1. Log in to the Administration Console as a user with the appropriate privilege.
2. Select Distributed Development | Areas and Deployment | Area Definitions.
3. Click New and select Library Cache Area.
4. Specify values for the new library cache area.
   - Make sure there is sufficient disc space in the library cache area to download the project or stream.
   - The library cache area ID cannot be the same as the remote node name.
   - (Optional) Prefix the library cache area ID with LCA_ so that it is easily recognized.
   - The node user ID must have write permission to the directory.

Associating Library Cache Areas with Projects or Streams

To use the library cache mechanism, associate your current project or stream with a library cache area.

1. Do one of the following:
   - Desktop client: from the Project or Streams menu select Preferences.
   - Web client: in the status bar click Settings.
2. Select a library cache area.

To dissociate a project or stream from a library cache area, do one of the following:

- Enter the following command with the appropriate project name parameter and the value 'null':

  `SCWS <project_name> /LIBRARY_CACHE_AREA="."`

- In the desktop client or web client, enter "." in the Library Cache Area field.

Testing Library Cache Areas

After you have associated your project or stream with a library cache area, test that the files are being copied to it.

PRIVILEGES Administration Privileges | Area Management | Update Library Cache Area Properties

1. Log into the desktop client as a user with the appropriate privilege.
2. Open a folder and select all its items.
3. Right-click and select Get. The files are copied to your working area and to the library cache area.
4. Navigate to the directory in the library cache area.
5. Check that the files have been copied to the area. Not all files are copied into the library cache area, for example, you can exclude files smaller than a certain size.

Populating Library Cache Areas

After you have tested that files are being copied to the area, you can populate it.

- If you have enabled library cache and try to compare or merge revisions, the operation is slow unless all the revisions are in the cache. Consider copying earlier revisions, in addition to the tip revisions, for some projects.
- When you populate an empty library cache area, it may take three or four times longer than a normal DOWNLOAD operation.
- When you populate library cache areas for the first time, consider creating scripts or a cron job that run daily to refresh the library cache areas.

Updating Library Cache Area Properties

PRIVILEGES Administration Privileges | Area Management | Update Library Cache Area Properties

1. Log in to the Administration Console as a user with the appropriate privilege.
2. Select Distributed Development | Area and Deployment.
3. Select the Area Definitions tab.
4. On the File Area tab select a library cache area and click Edit.
5. Edit the fields. For details, see the *online help*.

Deleting Library Cache Area Definitions

Deleting a library cache area does not delete the files contained in the area.

PRIVILEGES Administration Privileges | Area Management | Delete Library Cache Area Properties

1. Log in to the Administration Console as a user with the appropriate privilege.
2. Select Distributed Development | Areas and Deployment.
3. Select the Area Definitions tab.
4. Select a library cache area definition and click Delete.

Purging Library Cache Areas

You can purge files from a library cache area by removing all files or all files except for the latest revision.

NOTE When you upgrade Dimensions CM, we recommend that you remove all items from library cache areas.

PRIVILEGES Administration Privileges | Area Management | Update Library Cache Area Properties

To remove all files *except* for the latest revision:

1. As a Dimensions CM user with administrator privileges, start a `dmcli` session.
2. Enter the `plca` command:

   ```
   plca <library_cache_area_id>
   ```

   For example:

   ```
   plca lca_vs_payroll_1
   ```

To remove all files:

1. As a Dimensions CM user with administrator privileges start a `dmcli` session.
2. Enter the `plca` command:

   ```
   plca <library cache area> /purge_all
   ```

Example:

```
plca lca_vs_payroll_1 /purge_all
```

Optimizing Library Cache Area Performance

For details about optimizing library cache area performance, see the *online help*.

Using the Command-Line to Manage Library Cache Areas

You can use the following `dmcli` commands to manage library cache areas. For details see the *Command-Line Reference*.

- Create a library cache area: `CLCA`
- Associate a library cache area with a project or stream: `SCWS`
- Download items to a library cache area: `DLCA`
- List all library cache areas or the details of a specific library cache: `LLCA`
- Remove a library cache area: `RLCA`
- Update a library cache area's properties: `ULCA`

Appendix G

Executing Operations on Remote Nodes

Overview	552
Remote Job Flow	552
Executing Jobs on Remote Nodes	554
Preserving Job Logs	554

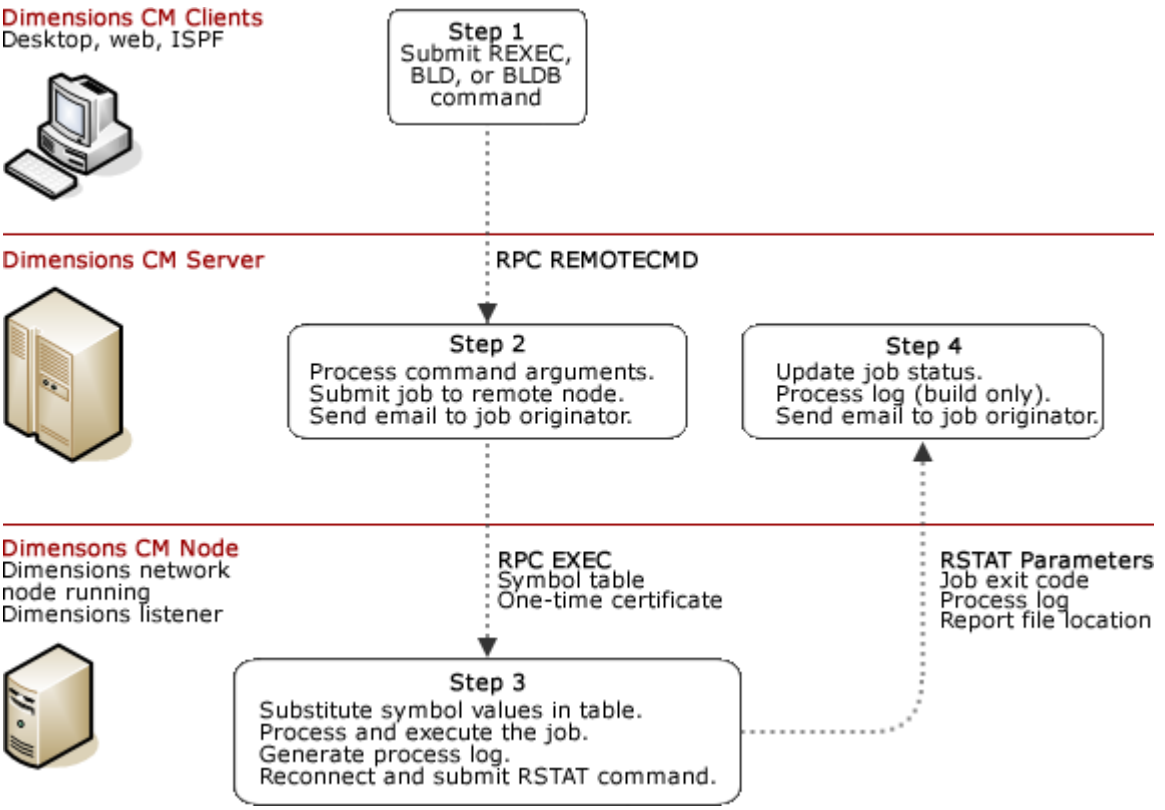
Overview

Remote job execution enables you to process operations on remote nodes on which a Dimensions® CM listener is running.

Dimensions CM supports a templating language on Windows, UNIX, and z/OS (USS and MVS) that you can use to customize the remote construction of processing. For details, see "The Templating Language and Processor" and "Remote Job Execution Templates" chapters in the *Developer's Reference*.

Remote Job Flow

The diagram below illustrates the remote job execution flow. The Dimensions CM client, Dimensions CM server, and remote node can be on the same, or separate, machines.



Step 1 Submit REXEC, BLD, or BLDB Command

A client initiates a remote operation, such as a build command or a remote job, by issuing the REXEC commands. Build executions go from a Dimensions CM server to a build server and then to a remote node.

You can use the desktop client's custom scripting tools to submit Dimensions CM commands to a Dimensions CM server. For details, see the *Dimensions CM online help*.

You can also use the Dimensions for z/OS ISPF SDK to customize user panels to perform actions against a server. For details see the SDK online help located in the Dimensions for z/OS installation media.

Step 2 Dimensions CM Server Processing

The Dimensions CM server processes the command, records the command information in the internal job queue, sends an email to the user who submitted the command, and submits a request to the remote node. The command's arguments and values, specified in the `/PARAMETERS` qualifier of the REXEC command, are embedded in the request. To view the contents of the job queue run the `RLIST` command.

If you specify the `/CAPTURE` qualifier, the server generates a one-time certificate and includes it in the request. You can use this certificate for custom post-build processing, for example, to update the status of a job using the `RSTAT` command.

Step 3 Dimensions Network Node Processing

The remote node receives the request and substitutes the parameters embedded in the request in the job template that you specified. The template can use the templating language described in *The Templating Language and Processor* chapter in the *Developer's Reference*. The job script that results from substitution of parameters in the template is then executed.

On MVS, the job is passed to JES for execution; on Windows, the batch file is passed to the Windows command-line processor *cmd.exe*; and on UNIX or USS, the shell script is passed to the shell processor.

Optionally, the template can invoke the Dimensions CM command-line tool and run the `RSTAT` command to update the job status on the Dimensions CM server.

Step 4 Dimensions Server Processing (Optional)

The `RSTAT` command is processed on the server and updates the job status.

You can also configure the Dimensions CM server to send email notifications when a job status is updated. Use the `DM_MAILS` variable in the Dimensions CM configuration file *dm.cfg* on the current local node, as described in "DM_MAILS" on page 78.

Executing Jobs on Remote Nodes

1. Specify the location of the template library on the current node in the Dimensions CM configuration file variable DM_TEMPLATE_CATALOG<n>. The defaults are:

   Windows: %DM_ROOT%\templates

   UNIX and MVS: $DM_ROOT/templates

   For more information about DM_TEMPLATE_CATALOG<n> see the *The Templating Language and Processor* chapter in the *Developer's Reference*.

2. Create the template and place it in the template library. For details see the *Remote Job Execution Templates* chapter in the *Developer's Reference*.

3. Submit the job using the REXEC command. For details see the *Command-Line Reference.*

4. After REXEC has completed use the following commands as required:

   - RSTAT: updates a job's attributes.
   - RLIST*:* lists jobs in the job queue.
   - RDEL: deletes jobs from the job queue.

IMPORTANT! If a remote node is running Windows Server 2003 or 2008 the user ID executing the script or job must be an administrator.

Preserving Job Logs

To aid the traceability of deployment areas you can view the results of jobs by examining the log files that are preserved in CM. There are different mechanisms to control when logs are preserved for REXEC jobs and deployment scripts.

Controlling Log Preservation for REXEC Jobs

To control when log files are preserved for an REXEC command, specify these optional qualifiers:

- /PRESERVE

  Preserves a log as an item in Dimensions CM. Specify one of these parameters:

  - YES: preserve the log.
  - ONERROR: only preserve the log if there is a job error.
  - ONSUCCESS: only preserve the log if the job succeeds.
  - NO: do not preserve the log.

  Administrators can specify the location in CM where logs are preserved, for details see page 67.

- /USER_FILENAME=[node::]filename

Saves a log in the path and folder that you specify.

- /[NO]SHOW

  Displays a log in the user's current session.

- /[NO]LOCK

  Locks a preserved job so that it cannot be deleted from CM unless it is unlocked.

Controlling Log Preservation for Deployment Scripts

To control when log files are preserved for deployment scripts, use the CA (Create Area) or UA (Update Area) commands.

Specifying Transfer Scripts

Transfer scripts are templates, processed on a CM server, that you customize to control the preservation of logs. Add the qualifier /TRANSFER_SCRIPTS to the Create Area or Update Area commands and specify a comma-separated list of pre, post, and fail transfer scripts in this order:

```
/TRANSFER_SCRIPTS=(
   <pre-script>.template
   <post-script>.template,
   <fail-script>.template)
```

The scripts are executed as follows:

- *Pre-script*: before items are deployed to an area.
- *Post-script*: after items are successfully deployed to an area.
- *Fail script*: if any script fails or any item cannot be deployed to an area.

Specifying Parameters for Transfer Scripts

You can specify log preservation control variables, and other values, as script parameters that are processed locally in a deployment area. Add the qualifier /SCRIPT_PARAMETERS to the Create Area or Update Area commands and specify a comma separated list of:

- Values or arrays of values:

  ```
  /SCRIPT_PARAMETERS=
     (<NAME>="value",
     <ARRAY NAME>="[value1,value2]")
  ```

- Variables to control when job logs are preserved in CM. The variables types are:
  - PRESERVE: preserve logs as items in CM.
  - SHOW: display logs in the user's current session.
  - LOCK: lock preserved jobs so that they cannot be deleted from CM.

  Each variable can have one of these parameters:

  - YES: preserve the log.
  - ONERROR: only preserve the log if there is a job error.
  - ONSUCCESS: only preserve the log if the job succeeds.

- NO: do not preserve the log.

The table below describes the variables:

Variable Name	Type	Description
DMPRE_SHOW	SHOW	Pre-script that displays the log in the user's current session.
DMPRE_PRESERVE	PRESERVE	Pre-script that preserves the log in CM.
DMPRE_LOCK	LOCK	Pre-script that locks the preserved log so that it cannot be deleted.
DMPOST_SHOW	SHOW	Post-script that displays the log in the user's current session.
DMPOST_PRESERVE	PRESERVE	Post-script that preserves the log in CM.
DMPOST_LOCK	LOCK	Post-script that locks the preserved log so that it cannot be deleted from CM.
DMFAIL_SHOW	SHOW	Fail script that displays the log in the user's current session.
DMFAIL_PRESERVE	PRESERVE	Fail script that preserves the log in CM.
DMFAIL_LOCK	LOCK	Fail script that locks the preserved log so that it cannot be deleted.
DMSHOW	SHOW	Displays all logs in the user's current session. This is the default value if you do not specify any pre, post, or fail SHOW values.
DMPRESERVE	PRESERVE	Preserves all logs. This is the default value if you do not specify any pre, post, or fail PRESERVE values.
DMLOCK	LOCK	Locks all logs so that they cannot be deleted from CM. This is the default value if you do not specify any pre, post, or fail LOCK values.

Examples

- To set JAVA_HOME and ANT_HOME, preserve all logs in CM. Lock the preserved fail logs so that they cannot be deleted from stages S1 and S2 (on the area SYSTEST):

```
UA SYSTEST
   /SCRIPT_PARAMETERS=
   (JAVA_HOME="C:\prog\Java\bin",
   ANT_HOME="C:\prog\Ant\bin",
   STAGES=["S1","S2"],
   DMPRESERVE=YES,
   DMFAIL_LOCK=YES)
```

- To preserve all pre, post, and fail logs when an error occurs on the SYSTEST area:

```
UA SYSTEST
   /TRANSFER_SCRIPTS=(pre.template,post.template,fail.template>)
   /SCRIPT_PARAMETERS=(
   DMPRESERVE=ONERROR)
```

- To preserve and lock all logs for the PROD area:

```
CA PROD
   /TRANSFER_SCRIPTS=(pre.template,post.template,fail.template>)
   /SCRIPT_PARAMETERS=(
   DMPRESERVE=YES,
   DMLOCK=YES)
```

Appendix H

Schedule Job Utilities and Configuration Parameters

About Schedule Jobs	558
dmschedule Utility	558
Schedule Job Parameters in dm.cfg	559
Schedule Job Configuration Parameters	560

About Schedule Jobs

Schedule jobs enable you to schedule the execution of any Dimensions® CM command, and to repeat the execution at preset intervals. For example, you can use it to schedule the automatic deployment of a baseline.

dmschedule Utility

`dmschedule` is a utility that is automatically started by `dmpool` on the Dimensions CM server host. `dmschedule` queries Dimensions CM databases for upcoming schedule jobs and runs them through the Dimensions CM server at the scheduled time.

Modes of Operation

`dmschedule` is a command-line utility that you can run as follows:

- Manually from a command prompt.
- Automatically via an existing customer scheduling system, such as `cron` (UNIX) or Scheduled Tasks (Windows).
- Automatically via the Dimensions CM listener. This is the default.

Security

You must run `dmschedule` from a user account that is part of the Dimensions CM ADMIN group. If you configure dmschedule using the Dimensions CM listener (see the dm.cfg parameters below) it runs as the same user as the dmpool process, which is specified in the following files:

- `%DM_ROOT%\dfs\listener.dat` (Windows)
- `$DM_ROOT/dfs/listener.dat` (UNIX)

Command-Line Syntax

```
dmschedule
[-dmpool]
[<config-file> | -]
[keyword=value]
```

where:

- `-dmpool`

  Causes dmschedule to continuously check for schedule jobs.

- `[<config-file> | -]`

  Specifies the location of the file containing configuration parameters. If you omit this option the default locations are:

- `%DM_ROOT%\dfs\schedule_config.dat` (Windows)
- `$DM_ROOT/dfs/schedule_config.dat` (UNIX)

If you want to use the default configuration file and parameters, use the minus symbol (-) instead of 'config-file'.

For more details about configuration parameters see "Schedule Job Configuration Parameters" on page 560.

- [keyword=value]

  Specifies configuration parameter key-value pairs. Parameters that you enter at the command line take precedence over parameters in the configuration file.

  If you need spaces in the parameters use double quotes.

  Parameter names are not case sensitive and values of `Y/y` and `N/n` are equivalent.

Schedule Job Parameters in dm.cfg

You can use the following entries in the Dimensions CM `dm.cfg` file.

```
#########################################################
# Scheduler service configuration
#########################################################
#
# Path to Scheduler Service Executable.
#
DM_DMSCHEDULE       %DM_PROG%dmschedule.exe
#
# Enable startup of Scheduler Service together with Dimensions Server.
#
DM_SCHEDULE_AUTOSTART       Y
#
# Working files and a log of Service activity is recorded in
# this directory.
#
DM_SCHEDULE_AUTOSTART_DIR       %DM_ROOT%temp\
#
# Interval in seconds to check the list of Schedule Jobs ready
# for execution.
#
DM_SCHEDULE_INTERVAL        30
#
# Enable a Schedule Service startup timeout, letting Dimensions Server
# to start all necessary components.
#
DM_SCHEDULE_SLEEP       Y
#
#Enable the continuous running of Scheduler Service started out of
# Dimensions Server (any external tool).
#
#DM_SCHEDULE_NO_STOPFILE        Y
```

where:

- DM_DMSCHEDULE

  Specifies the full path and filename of dmschedule.

- DM_SCHEDULE_AUTOSTART

  Specifies whether the Dimensions CM server shell automatically starts `dmschedule`. To disable scheduling, leave this variable empty or comment it out. You can still schedule jobs, but they will not automatically run at the scheduled time.

- DM_SCHEDULE_AUTOSTART_DIR

  Specifies the directory for storing working files. If you enable scheduler logs, they are stored under this path by default. For details, see Schedule Job Configuration Parameters.

- DM_SCHEDULE_INTERVAL

  Specifies the interval in seconds between queries that dmschedule makes to the Dimensions CM database(s) for the list of upcoming jobs. The shorter the interval, the more accurate the execution time, but more database resources are used.

- DM_SCHEDULE_SLEEP

  Starts the processing of schedule jobs by dmschedule after the startup time-out, allowing the Dimensions CM server to start all necessary components. Disabling this variable can cause the failure of the first attempt(s) to execute schedule jobs (depending on your configuration and environment).

- DM_SCHEDULE_NO_STOPFILE

  Starts `dmschedule` externally, not with Dimensions CM server or dmpool. If you disable this variable and start dmschedule externally, it exits immediately after first querying the database(s) and initiating job executions.

Schedule Job Configuration Parameters

You can specify the following configuration parameters in the file `schedule_config.dat` or at the command line. The parameters are case-insensitive and are specified as keyword=value pairs. `'#'` indicates a comment and blank lines are allowed.

- LOGGING

  If set to `'Y'` enables the output of activity logs.

- LOG

  Redirects the log output to a specific file. By default log files are created in the directory `%DM_SCHEDULE_AUTOSTART_DIR%`.

- DEBUG

  If set to `'Y'` adds detailed information to the log output.

- DATABASE

  You can process multiple databases in sequence by specifying `DATABASE_n` where `'n'` is an integer order number (beginning with one, followed by two, then three, and so on) for each database to be processed. In each case, the `DATABASE` key specifies the

usual Dimensions CM connection string for a database. If the database requires a password to allow the user ID to connect, set it using the dmpasswd utility. For example:

```
DATABASE_1=intermediate@DIM1013
DATABASE_2=qlarius_cm@DIM1013q
```

- HOST

  The HOST variable defines the Dimensions CM server for the job execution. Each 'HOST_n' variable should correspond to the DATABASE_n value. For example:

  ```
  HOST_1=localhost
  HOST_2=dim1013box
  ```

- DMUSER

  Specifies the Dimensions CM user who initiates the job execution. Each 'DMUSER_n' is used to connect the Dimensions CM server running on the corresponding 'HOST_n' host, for example:

  ```
  DMUSER_1=dmsys
  DMUSER_2=dmsys
  ```

  The specified user must have the privilege Manage Schedule Job to run other users' schedule jobs.

Contents of the Configuration File

The default contents of the %DM_ROOT%\dfs\schedule_config.dat (Windows) or $DM_ROOT%/dfs/schedule_config.dat (UNIX) file are as follows:

```
# ################################
#
# Scheduler service configuration
# -------------------------------
#
# Do not modify these parameters, unless you are aware of their impact
# on the application.
#
# ################################

#
# Specify the variable below if you want to enable logging.
# By default, it is switched off.
#
# LOGGING=Y

#
# Specify this variable if you want the program's logging output
# to be redirected to specific file.
# By default, %DM_SCHEDULE_AUTOSTART_DIR% folder is used.
#
# LOG=/schedule.log

#
# Specify the variable below if you want detailed debug enabled.
# By default, it is switched off.
```

```
#
# DEBUG=Y

#
# Databases to process
# ====================
#
# You can specify any number of databases below to process by using the
    format
# DATABASE_[n]=<databaseName>, where [n] is the order in which you wish
# the databases to be processed.
#
DATABASE=qlarius_cm@DIM1013q

#
# Application server host
# =======================
#
# You should specify the host(s) running Dimensions CM Server, which is
    working
# with corresponding database(s) specified by the parameter DATABASE_[n]
    above.
#
HOST=localhost
#
# Schedule Job Manager
# ====================
#
# You should specify Dimensions User(s) with the privilege 'Manage
    Schedule Jobs'
# to initiate execution of Schedule Jobs on the corresponding Dimensions
    CM
# Server(s) specified by parameters HOST_[n] and DATABASE_[n] above.
#
DMUSER=dmsys
```

NOTES

- The variables DATABASE, HOST, and DMUSER are preset by the installer to the current Dimensions CMserver configuration.
- If there is only database connection defined (that is, one of each DATABASE, HOST, and USER variable), you do not need to append the variable index ('_1') to each variable. For example, in the \dfs\schedule_config.dat configuration file, the default entries are as above, namely:

  ```
  DATABASE=qlarius_cm@DIM1013q

  HOST=localhost

  DMUSER=dmsys
  ```

NOTES (continued)

- If there are several database connections defined, you need to append the appropriate variable index (`'_n'`) to each variable (that is, `DATABASE_n`, `HOST_n`, and `DMUSER_n` appropriately). The first occurrence of each variable should have the index '_1', the second occurrence of each variable '_2', and so on. For example:

```
DATABASE_1=intermediate@dim10
DATABASE_2=qlarius_cm@dim10
DATABASE_3=qlarius_cm@DIM1013q

HOST_1=localhost
HOST_2=qlariushost
HOST_3=host1013

DMUSER_1=dmsys
DMUSER_2=dmsys
DMUSER_3=dmsys
```

Appendix I

Migrating MBCS Data to UTF-8 Data

Chapter Overview	566
Disclaimer	566
DMDBA Migrate Utility Features and Prerequisites	566
Overview of the Migration Process	567
DMDBA Migrate Syntax	567

Chapter Overview

IMPORTANT! The utility described in this appendix should only be run by a database administrator or a user with detailed technical knowledge.

This appendix describes a Dimensions® CM `dmdba migrate` utility that assists in the migration of data from an MBCS format Serena-Supplied Runtime RDBMS system or an Oracle Enterprise system to an UTF-8 format system of the similar kind. If you require more details concerning the invoking of `dmdba`, see "Invoking the dmdba Tool" on page 136.

NOTE There is an alternative method using Oracle data pump and a few scripts which check such things as request descriptions. This method is quicker and more efficient than the using the MIGRATE utility. For details, see the Knowledgebase article:

How to migrate WE8ISO8859P1 databases to UTF8 via the Oracle data pump method.

Disclaimer

OpenText makes no warranty of any kind in regard to the contents of this appendix, including but not limited to implied warranties of merchantable quality or fitness for any particular purpose. OpenText shall not be liable for errors contained in it or for incidental or consequential damages in connection with the furnishing, performance, or use of this appendix or associated software. The information in this appendix is subject to change without notice.

DMDBA Migrate Utility Features and Prerequisites

The following are features and prerequisites of the `dmdba migrate` utility:

- The utility is only available for use on a Serena-Supplied Runtime RDBMS system or an Oracle Enterprise system.
- The utility performs a migration of data *not* an upgrade, that is, the data in the original system is left unchanged.
- The migration process is a "single-shot" process. It does not support an incremental migration approach. But it does have the ability to pick up after a severe error.
- You need to two separate Oracle instances involved in the migration process, an MBCS read-only source and an UTF-8 writable target. The MBCS database source is not altered in any way, treated as a read-only system.
- The migration is performed on a per-table basis. The ability to recover from severe errors supports restarting the migration process from the table that failed.
- The user must provide the details of the source MBCS character set. It is also assumed that all the source data is in this encoding.
- Verbose status information is passed back to the user, providing information about the progress where possible.

- The UTF-8 instance must already have the standard Dimensions tablespaces setup using a standard Dimensions CM installation.

Overview of the Migration Process

The following details a high-level overview of the migration process (a full description of the syntax is given in "DMDBA Migrate Syntax" on page 567):

1. Ensure that the source MBCS system has been successfully upgraded to the appropriate version of Dimensions CM.
2. After the source instance (and databases) is successfully upgraded, create a separate UTF-8 instance. The size of the UTF-8 instance needs to be of sufficient size to reflect the greater database size requirements of UTF-8 compared to MBCS.
3. Ensure that Dimensions CM and the Serena-Supplied Runtime RDBMS or Oracle Enterprise are configured on the MBCS machine to allow access to both the MBCS instance and the UTF-8 instance. This configuration is required primarily to allow `dmdba` access to both the MBCS and UTF-8 instances (`dmdba` can automatically detect and handle MBCS/UTF-8 character set differences on a database connection).
4. Run `dmdba` migrate as explained in "DMDBA Migrate Syntax" on page 567.

   The migration runs as an unattended process reporting progress and exits before completion only if a severe error occurs.
5. When the migration is completed successfully, Dimensions CM is automatically configured to point to the new UTF-8 system.
6. If the MBCS database has real multibytes characters (for example, Korean hieroglyphs), add:

   ```
   DBIO_SLOW_ROW_COPY     TRUE
   ```

   to the Dimensions CM server configuration file `dm.cfg`.

DMDBA Migrate Syntax

The `dmdba migrate` syntax is as follows:

```
dmdba <MBCS_systemAccountDetails> migrate <UTF-
    8_systemAccountDetails> <sourceCharacterSet>
    [/logfile=<logfile>] [/schemas=<schema1,schema2>] [/force]
    [/debug]
```

where:

`<MBCS_systemAccountDetails>`	If you are using the Serena-Supplied Runtime RDBMS, this is the standard `system/<system_password>@DNS` account details for the source MBCS instance. For example, `system/manager@db1`.

	If you are using your own Oracle Enterprise, this could be either: ■ the details of the standard `system/<system_password>@DNS` account for the source MBCS instance; or ■ if you are operating in a "locked down Oracle environment", the details of the specific Oracle account to whom DMDBA privileges have been granted for the source MBCS instance (see "Oracle Privileges Required for DMDBA Commands" on page 133).
`<UTF-8_systemAccountDetails>`	If you are using the Serena-Supplied Runtime RDBMS, this is the standard `system/<system_password>@DNS` account details for the target UTF-8 instance. For example, `system/manager@db2`. If you are using your own Oracle Enterprise, this could be either: ■ the details of the standard `system/<system_password>@DNS` account for the target UTF-8 instance; or ■ if you are operating in a "locked down Oracle environment", the details of the specific Oracle account to whom DMDBA privileges have been granted for the target UTF-8 instance (see "Oracle Privileges Required for DMDBA Commands" on page 133).
<sourceCharacterSet>	Is the source character set of the MBCS system. The following character sets are supported: https://www.daylight.com/meetings/emug04/Delany/charsets.html
`<logfile>`	Is an optional parameter for specifying a logfile for output messages.
<listOfSchemasToMigrate>	Is an optional parameter to list schemas to be migrated. If such a list is not specified, all schemas are migrated by default.
`<force>`	Is an optional parameter to force the migration process to start from the beginning.
`<debug>`	Is an optional parameter to run the migration process in debug mode.

Appendix J

Logging and Tracing Facilities

Chapter Overview	570
Displaying Connection Pooling Statistics	570
Enabling Dimensions CM Listener Tracing	570
Logging ALF Events	570
Logging Schedule Jobs	571
Logging Web Services	571
Managing Replication Logs	572
Tracing API Events and Merge Events	572
Tracing Template Processing	573

Chapter Overview

This appendix describes Dimensions® CM logging and tracing facilities. It duplicates, expands upon, or cross refers to such facilities already described in this and other guides in the Dimensions CM documentation set. Details described include such details as:

- How to turn logging/tracing on.
- The default log/trace locations and where these defaults can be changed.

Displaying Connection Pooling Statistics

- See "Displaying Connection Pooling Statistics" on page 170

Enabling Dimensions CM Listener Tracing

There may be occasions when it is necessary to diagnose possible issues with the Dimensions CM Listener. Dimensions CM provides two special initialization parameters that you can use to start the listener in a mode that traces internal progress and status information to a log file for debugging purposes.

To enable tracing, add the following lines to the `listener.dat` file contained in the `$DM_ROOT/dfs` directory (UNIX server or agent) or `%DM_ROOT%\dfs` directory (Windows server or agent):

```
-tracedir <directory_name>
-trace
```

where

`<directory_name>` is the name of a directory where the trace files are created.

After you have made this change, stop and restart Dimensions CM. To disable this tracing, remove the two variables and restart Dimensions CM.

Logging ALF Events

Trace logging of the ALF events fired by Dimensions CM is provided as part of the SDP tracing. This may be enabled by adding the following to the Dimensions CM dm.cfg configuration file located in the Dimensions CM "root" directory:

- To enable SDP tracing:

```
DM_SDP_TRACE    <full directory path>
```

- To enable ALF event specific tracing:

```
DM_DEBUG_ALF_EVENTS    1
```

Logging Schedule Jobs

See "Schedule Job Configuration Parameters" on page 560.

Logging Web Services

All web services contain a logging mechanism that logs:

- The web service input parameters.
- The result of performing the Dimensions CM command and exceptions if any occur.

By default, the logging level of Dimensions CM web services is set to DEBUG, it can be changed in the following file in an installed Dimensions CM Windows installation:

```
<Tomcat-Root>\webapps\dmwebservices2\
    WEB-INF\classes\log4j.properties
```

or from the following file in an installed Dimensions CM UNIX installation:

```
<Tomcat-Root/webapps/dmwebservices2/
    WEB-INF/classes/log4j.properties
```

The Tomcat log file for a Dimensions CM Windows installation is:

```
<Tomcat-Root>\logs\stdout_<YYYYMMDD>.log
```

and for a Dimensions CM UNIX installation it is:

```
<Tomcat-Root>/logs/stdout_<YYYYMMDD>.log
```

The following is an example of the logging of a web service that executed successfully:

```
[DEBUG] entering createRequest
[DEBUG] Request details:
   TypeName=CR
   ProductId=QLARIUS
   Title=WS2 Test 12/9/2008 16:20
   DetailedDescription=description string
[DEBUG] Setup Attribute:
   [0]:Name=Severity
   [0]:Type=Single_Value
   [0]:DataType=Char
   [0]:Value=1
[DEBUG] Setup Attribute:
   [1]:Name=PLAN_FINISH
   [1]:Type=Single_Value
   [1]:DataType=Date
   [1]:Value=2008-11-07T15:31:25+02:00
 [DEBUG] Result of DM operation: "SUCCESS: The request QLARIUS_CR_66 has been forwarded
    to PETA

Operation completed"
[DEBUG] leaving createRequest
```

The following is an example of the logging of a web service that failed to execute successfully:

```
[DEBUG] entering deployRequest
[DEBUG] Request details:
   requestId=QLARIUS_CR_48
   stage=DEVELOPMENT
   comment=comment string
   traverseChildren=false
[DEBUG] leaving deployRequest with exception: Error: Request is already at stage
    DEVELOPMENT
```

In some cases the log may not contain records concerning web service invocation. If a SOAP request was actually issued, this usually means that it is highly probable that the request did not pass the Axis2 validation.

Managing Replication Logs

See "Managing Replication Logs" on page 368 and online help for associated detailed dialog box information.

Tracing API Events and Merge Events

IMPORTANT! We recommend that API and merge event tracing is only enabled upon request by Support to help them assist you with resolving an events problem.

Tracing of the API events fired by Dimensions CM is enabled by modifying the `DM_EVENT_TRACE` Dimensions CM `dm.cfg` configuration symbol to a non-null value. For example:

```
DM_EVENT_TRACE     Y
```

API event trace logging produces debug information whenever an event is fired, for example;

- Pre-event calls.
- Validation.
- First (and possibly second) calls for each event.

For Dimensions CM for UNIX, the trace log is created as a `ptxxxxx` temporary file in the directory `/tmp` on the UNIX server.

Trace logging of merge events is enabled by modifying the `DM_EVENT_TRACE` and `DM_TRACE` Dimensions CM `dm.cfg` configuration symbols to a non-null values. For example:

```
DM_EVENT_TRACE     Y
DM_TRACE           Y
```

Tracing Template Processing

Tracing of template processing is enabled by adding the following Dimensions CM server `dm.cfg` configuration symbols with non-null values:

`DM_TPLB_LOG`	Switches on tracing of template processing. Used to specify the name of the file to contain detailed messages issued during template processing, and the substitutions performed. You can customize this string before using it: ■ `%d`—inserts the date in the format `yyyymmdd`. ■ `%t`—inserts the time in the format `hhmmss`. ■ `%p`—inserts the *process id*. ■ `%%`—inserts one '%' character. Default: none Example: `tmp/template_log_%d_%t.log`
`DM_TPLB_TTLOG`	Specifies that any trace sent to the file that you specify in `DM_TPLB_LOG` is also sent to the message routine (the trace comes back to the client to be displayed). ■ Default: null ■ Example: yes

Appendix K

VRS Data Caching

Overview 576

Pre-Caching VRS Data 576

Overview

The Dimensions CM server pre-caches Versioned Repository Schema (VRS) data into a directory on disk. Pre-caching is automatic when a project, stream, or baseline is accessed for the first time. The main benefits of data caching are improved performance when:

- Opening, or switching between, projects, streams, and baselines.
- Performing operations such as update, deliver, create baseline, create stream, and merge.

CM administrators can perform initial pre-caching of the name store, project, or baseline structure data for the current base database.

Pre-Caching VRS Data

NOTE Shut down your Dimensions CM server processes before running this operation.

Run the following DMBDA command to perform initial pre-caching of VRS data:

```
PRECACHE
<mode>
[<project-spec> | <baseline-spec> | /BULK_FILE=<file> ]
```

where:

- `<mode>` specifies the type of data to be pre-cached. This parameter is mandatory and must have one of the following values:
  - `NAMESTORE`

    Pre-caches namestore data.
  - `PROJECT`

    Pre-caches project structure data.
  - `BASELINE`

    Pre-caches baseline structure data.
- `<project-spec>`

  Specifies a single project or stream.
- `<baseline-spec>`

  Specifies a single baseline.
- `/BULK_FILE=<file>`

  Specifies a file containing multiple project or baseline specifications to pre-cache.

TIP The project and/or baseline specification has the same syntax as a LIKE expression in SQL.

Usage examples:

```
DMDBA cm_typical/cm_typical@dim14
```

```
CM_TYPICAL> PRECACHE NAMESTORE

CM_TYPICAL> PRECACHE PROJECT %

CM_TYPICAL> PRECACHE BASELINE QLARIUS:%

CM_TYPICAL> PRECACHE PROJECT /BULKFILE="C:\Temp\A,txt"
```

where `A.txt` contains multiple project specifications, for example:

- `QLARIUS:BRANCHA\`
- `QLARIUS:BRANCHB\`