



Project and Portfolio Management Center

Software Version: All versions

Deployment Management Extension for Oracle

Go to **HELP CENTER ONLINE**

<http://admhelp.microfocus.com/ppm/>

Legal Notices

Disclaimer

Certain versions of software and/or documents (“Material”) accessible here may contain branding from Hewlett-Packard Company (now HP Inc.) and Hewlett Packard Enterprise Company. As of September 1, 2017, the Material is now offered by Micro Focus, a separately owned and operated company. Any reference to the HP and Hewlett Packard Enterprise/HPE marks is historical in nature, and the HP and Hewlett Packard Enterprise/HPE marks are the property of their respective owners.

Warranty

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

Restricted Rights Legend

Contains Confidential Information. Except as specifically indicated otherwise, a valid license is required for possession, use or copying. Consistent with FAR 12.211 and 12.212, Commercial Computer Software, Computer Software Documentation, and Technical Data for Commercial Items are licensed to the U.S. Government under vendor’s standard commercial license.

Copyright Notice

© Copyright 1997-2022 Micro Focus or one of its affiliates.

Contents

Project and Portfolio Management Center	1
Getting Started	15
Introduction to the Extension for Oracle E-Business Suite	15
What's New and What's Changed in Version 9.50	16
About This Document	16
Related Information	18
Prerequisite Knowledge and Experience	18
Integration of the Extension with Oracle E-Business Suite	19
Local Integration (Single Database)	20
Linked Database Integration	21
Business Concepts Used in the Oracle Environment	22
Business Roles	22
Project Manager	22
Project Sponsor	23
Track Owner	23
Database Administrator (DBA)	24
Business Lead	24
Team Lead	24
Developer	25
Analyst	25
Additional Business Concepts	25
Business Area	26
Source	26
Track	26
Installing or Upgrading the Extension	27
Overview of Installation and Upgrade	27
System Requirements	27
PPM	27
Oracle E-Business Suite Instances	28
General Upgrade Impacts and Guidelines	29
Specific Upgrade Impacts for Version 9.50	30
Preparing for Installation or Upgrade	30
General Preparations for Installation or Upgrade	30
Performing Backup and Restarting the Server in Restricted Mode	32
Installation Procedure	32
Run the Installation Script and Check the Logs	32

Logs Generated During Installation	34
Verify the Installation	35
Restart the PPM Server in Normal Mode	36
Installing the Extension’s Preconfigured Dashboard Pages	36
Configuring the Extension	40
Overview of Configuration	40
Update the server.conf File	40
Configuring Database Links	41
Configuring Environments	41
Configuring the Host Tab	42
Configuring the Extension Data Tab, Oracle e Subtab	44
App Code Loader	48
Remote Execution of Object Migrator Concurrent Requests	49
Configuring Oracle E-Business Suite Users	52
Configuring Security Groups	53
Configuring Request Types	54
Configuring Workflows	55
Configuring Object Types	58
AOL and GL Object Types	59
AOL:Single <Entity> Object Types	59
OraApps Oracle Workflow Object Type	59
OraApps AKLOAD Migrate AK Setups Object Type	60
FNDLOAD Automation Object Types	61
Managing Integration with Oracle E-Business Suite	61
Changing the Integration Method to Local	62
Changing the Integration Method to Linked	63
Managing Linked Integration During Oracle E-Business Suite Downtime	64
Configuring a server.conf Parameter for Fewer WatchDog Errors	65
Configuring a server.conf Parameter for Better PPM Center Performance	65
Extension Interface	67
Menus	67
Oracle Applications Menu	67
Issue Management Submenu	68
Project Management Submenu	68
Deployment Management Submenu	69
Instance Management Submenu	70
Patch Management Submenu	71
Oracle Applications Roles Menu	71
Project Sponsor Submenu	72

Project Manager Submenu	73
Track Owner Submenu	74
Apps DBA Submenu	76
Preconfigured Dashboard Page and Portlets	77
Oracle Apps Project Manager Dashboard Page and Its Portlets	78
Oracle Apps 11i: Overall Status Portlet	79
OraApps: Activity by Priority Portlet	79
OraApps: Critical Activities Portlet	80
Oracle Apps: Activities Summary (Past 2 Weeks) Portlet	80
Oracle Apps Project Sponsor Dashboard Page and Its Portlets	81
Oracle Apps 11i: Overall Status Portlet	81
Oracle Apps 11i: All Phases Portlet	82
OraApps: Critical Activities Portlet	82
Oracle Apps Reports Track Owner Dashboard Page and Its Portlets	83
Oracle Apps 11i: Reports Track Status Portlet	84
Reports: Activity by Priority Portlet	84
Reports: Critical Activities Priority	84
Oracle Apps DBA Dashboard Page and Its Portlets	85
Oracle Apps: Patch Deployment Status Portlet	86
Bugs reported to Oracle Portlet	86
OraApps: Config. Deployments Portlet	86
Oracle Apps: Instance cloning queue Portlet	87
Managing Issues	88
Overview of Managing Issues	88
Capturing and Tracking Issues	88
Processing User Requests	89
Reports About Issues	90
OraApps Critical Requests Summary Report	91
OraApps Apps Issues Detail Report	91
OraApps Apps Issues Summary Report	91
OraApps IT Demand Summary Report	91
Managing Projects	92
Overview of Managing Projects	92
Planning an Upgrade Strategy	93
Planning an Upgrade Strategy	93
Integrating Project Management and Demand Management	94
Performing a Gap Analysis	94
Chapter 6: Using the OraApps Release11i Upgrade Work Plan Template	95
Overview of Work Plan Template	95

Work Plan Template Structure	96
Phase I: Project Startup	98
Phase II: Business Operations Analysis	98
Phase III: Prototype, GAP Analysis, High Level Design	98
Phase IV: Development and System Test	99
Phase V: Integration and Performance Testing	100
Phase VI: Simulations	100
Phase VII: Cut Over	101
Phase VIII: Post Implementation Support	101
Customizing the Work Plan Template	101
Reporting Status	101
OraApps Status Update Request Request Type	102
OraApps Status Update Request Workflow	102
Reports About Project-Related Status	103
OraApps Critical Requests Summary Report	103
OraApps IT Demand Summary Report	103
Configuring Extension Entities for Managing Projects	103
Configuring Request Types	104
OraApps GAP Analysis Request Request Type	104
OraApps Status Update Request Request Type	104
Configuring Workflows	105
OraApps GAP Analysis Process Workflow	105
OraApps Status Update Request Workflow	106
Configuring Report Types	106
Managing Changes	107
Overview of Managing Changes	107
Requesting Changes	108
Requesting New Reports	108
Requesting New Interfaces	109
Requesting Conversions	109
Requesting Enhancements	110
OraApps Enhancement Request Request Type	110
OraApps Design & Development Request Type	111
Requesting Setup Changes	112
Deploying Oracle Changes	113
Overview of AOL and GL Migration	114
AOL or GL Migration Execution	116
AOL Concurrent Request Log or GL Concurrent Request Log	116
Overview of FNDLOAD Automation Object Types	117

Managing Instances	119
Overview of Managing Instances	119
Cloning Requirements	120
Cloning Instances	121
Cleaning Up After Cloning	123
Configuring Instance Management	124
Configuring Environments	124
OraApps Cloning Request Request Type	126
OraApps Cloning-Related Workflows	127
OraApps Cloning Process Workflow	127
OraApps 11i Cloning Workflow	128
OraApps 11i Cloning Object Type	128
ksc_oa_copy_clone_scripts Special Command	129
	129
Managing Patches	130
Overview of Managing Patches	130
Analyzing Patch Information	131
OraApps Patch Detail Report	131
OraApps Patch Analysis Report	132
OraApps Patch Data Capture Subworkflow	132
Analyzing Patches	133
Overview of Applying Patches	133
Automating ADPATCH and ADADMIN	134
Limitations	135
Execution Logs	135
Merging Patches	136
Reports About Patches	137
OraApps Patch Detail Report	137
OraApps Patch Analysis Report	138
Patch Application Comparison Report	138
Patches Applied to an Environment Report	138
Pending Patches Report	138
OraApps Patch Matrix Report	139
OraApps Patches Applied for Specific Bug Report	139
Oracle E-Business Suite Implementation and Patching Scenarios	139
Configuring Patch Management	140
System Requirements	140
Configuring the OA -Oracle Patch Admin Security Group	141
Configuring Environments	141
Patch Stage Environment	142

Patch Destination Environment	145
SERVER_ENV_NAME Environment	149
Configuring Object Types	149
Oracle Patch Object Type	150
OraApps Merge Patch Object Type	151
Object Types that Automate Use of the ADADMIN Utility	152
Configuring Workflows	153
OraApps Patch Deployment Workflow and Its Subworkflows	153
OA -Capture Patch Workflow Step Source	155
Creating a Workflow to Merge Patches	156
Configuring Report Types	156
OraApps Patch Analysis Report	156
OraApps Patch Detail Report	157
Special Commands	157
ksc_oa_capture_patch	157
ksc_oa_purge_temp_patch	157
Maintaining PPM Center	158
Maintaining the PPM Server	158
Maintaining the PPM Center Database	158
Appendix A: Object Types	159
Reference Object Types	159
List of Object Types	159
Application Data Migration Object Types	163
AOL and GL Object Types	163
Description	163
Migration Order	164
Field Descriptions	164
AOL:Single Conc Mgr Entry Object Type	166
Description	166
Configuration Consideration	166
Field Descriptions	166
AOL:Single GUI Menu Entry Object Type	168
Description	168
Configuration Consideration	168
Field Descriptions	168
AOL:Single Report Group Unit Object Type	169
Description	170
Configuration Consideration	170
Field Descriptions	170
OraApps AKLOAD Migrate AK Setups Object Type	172

Description	172
Configuration Consideration	172
Field Descriptions	173
OraApps Oracle Workflow Object Type	175
Description	176
Configuration Consideration	176
Field Descriptions	176
Patch and Administration Object Types	177
Oracle Patch Object Type	178
Description	178
Field Descriptions	179
OraApps Merge Patch Object Type	181
Description	181
Configuration Consideration	182
Field Descriptions	182
OraApps ADADMIN Compile APPS Schema Object Type	184
Description	184
Configuration Consideration	184
Field Descriptions	185
OraApps ADADMIN Compile Flexfields Object Type	186
Description	186
Field Descriptions	186
OraApps ADADMIN Generate Jar Files Object Type	188
Description	188
Field Descriptions	188
OraApps ADADMIN Generate Messages Object Type	190
Description	190
Field Descriptions	190
Instance Management Object Type	192
OraApps 11i Cloning Object Type	192
Description	192
Configuration Consideration	194
Field Descriptions	194
FNDLOAD Automation Object Types	196
AR:Phone Country Codes Object Type	196
Description	196
Configuration Consideration	197
Field Descriptions	197
INV:Units of Measure Object Type	198
Description	199
Configuration Consideration	199
Field Descriptions	199

Run/Patch File System File Migration object types	200
OA- Run System File Migration object type	201
Description	201
Configuration consideration	201
Field descriptions	201
OA- Patch System File Migration object type	203
Description	203
Configuration consideration	203
Field descriptions	203
Special commands to get Run/Patch file system	205
ksc_oa_get_run_fs	205
ksc_oa_get_patch_fs	205
Validations to get Run /Patch file system location	206
OA - Run File System location	206
OA - Patch File System location	206
Appendix B: Request Types	207
Overview of the Request Process	207
Reference Request Types	208
List of Request Types	208
OraApps Application Issue Request Type	209
Description	209
Configuration Considerations	210
Field Descriptions	211
OraApps Cloning Request Request Type	213
Description	214
Configuration Considerations	214
Field Descriptions	214
OraApps Conversion Request Request Type	216
Description	216
Configuration Considerations	217
Field Descriptions	217
OraApps Design & Development Request Type	218
Description	219
Configuration Considerations	219
Field Descriptions	219
OraApps Enhancement Request Request Type	221
Description	221
Configuration Considerations	222
Field Descriptions	222
OraApps GAP Analysis Request Request Type	223

Description	224
Configuration Considerations	224
Field Descriptions	224
OraApps Interface Request Request Type	226
Description	226
Configuration Considerations	226
Field Descriptions	227
OraApps Report Request Request Type	228
Description	228
Configuration Considerations	229
Field Descriptions	229
OraApps Setup Change Request Request Type	230
Description	230
Configuration Considerations	231
Field Descriptions	231
OraApps Status Update Request	233
Description	233
Configuration Considerations	233
Field Descriptions	234
Appendix C: Workflows	237
Reference Request Types	237
List of Request Types	237
General Configuration Considerations	239
OraApps Application Issue Workflow	239
Workflow Diagram and Step Descriptions	240
Workflow Steps with Predefined Security	242
OraApps Cloning Process Workflow	243
Workflow Diagram and Step Descriptions	243
OraApps Conversion Process Workflow	244
Workflow Diagram and Step Descriptions	244
Steps with Predefined Security	246
OraApps Customization/Configuration Deployment Workflow	246
Configuration Considerations	247
Workflow Diagram and Step Descriptions	248
Steps with Predefined Security	250
OraApps Design & Development Workflow	251
Workflow Diagram and Step Descriptions	252
Steps with Predefined Security	253
OraApps Enhancement Process Workflow	254
Workflow Diagram and Step Descriptions	254

Steps with Predefined Security	256
OraApps GAP Analysis Process Workflow	256
Workflow Diagram and Step Descriptions	257
Steps with Predefined Security	258
OraApps Interfaces Process Workflow	259
Workflow Diagram and Step Descriptions	259
Steps with Predefined Security	261
OraApps Patch Deployment Workflow and Its Subworkflows	261
Configuration Considerations	262
Workflow Diagram and Step Descriptions	263
Steps with Predefined Security	266
OraApps Reports Process Workflow	267
Workflow Diagram and Step Descriptions	268
Steps with Predefined Security	269
OraApps Setup Change Proces Workflow	270
Workflow Diagram and Step Descriptions	270
OraApps Status Update Request Workflow	271
Workflow Diagram and Step Descriptions	272
Steps with Predefined Security	274
OraApps 11i Cloning Workflow	276
Workflow Diagram and Step Descriptions	277
Appendix D: Report Types	280
Reference Report Types	281
List of Request Types	281
General Configuration Considerations	282
OraApps Apps Issues Detail Report	282
Description	282
Configuration Considerations	282
Field Descriptions	283
OraApps Apps Issues Summary Report	287
Description	287
Configuration Considerations	287
Field Descriptions	288
OraApps Critical Requests Summary Report	291
Description	291
Configuration Considerations	292
Field Descriptions	292
OraApps IT Demand Summary Report	293
Description	293
Configuration Considerations	294

Field Descriptions	294
Patch-Related Reports	296
OraApps Patch Analysis Report	297
Description	297
Field Descriptions	297
OraApps Patch Detail Report	300
Description	300
Configuration Considerations	300
Field Descriptions	300
OraApps Patch Matrix Report	303
Description	303
Configuration Considerations	304
Field Descriptions	304
OraApps Patches Applied for Specific Bug Report	306
Description	306
Configuration Considerations	307
Field Descriptions	307
Patch Application Comparison Report	308
Description	308
Field Descriptions	309
Patches Applied to an Environment Report	310
Description	310
Field Descriptions	311
Pending Patches Report	312
Description	312
Field Descriptions	312
Appendix E: Tokens	314
List of Tokens	314
Send Us Feedback	317

Getting Started

Introduction to the Extension for Oracle E-Business Suite

Deployment Management Extension for Oracle E-Business Suite Guide (usually referred to hereafter as “the Extension”) helps organizations to automate deployment management and manage a wide range of business processes in the Oracle E-Business Suite (Oracle Apps) environment.

The Extension enhances the functionality of the Deployment Management application in Project and Portfolio Management Center (PPM) by providing predefined entities that are unique to Oracle E-Business Suite environments. These entities include specialized workflows, request types, work plan templates, object types, special commands, tokens, and report types (beyond the reports of Oracle E-Business Suite itself).

The Extension lowers costs and shortens implementation time for Oracle E-Business Suite by providing automated processes built upon known best practices for the following:

- Deployment management, including project and ongoing operational requests
- Project management, including implementation and upgrade of Oracle E-Business Suite and gap analysis
- Deployment management, including, for example, managing the following:
 - Requests for enhancements
 - Requests for new report types
 - Requests for conversion and importation of data from third-party applications into Oracle®
 - Deployment of changes
- Instance management, including instance cloning
- Patch management, including impact analysis and patch deployment

In addition, the Object Migrator and GL Migrator products work in conjunction with the Extension. See the *Object Migrator Guide* and the *GL Migrator Guide*.

Note: For information about which Oracle releases, which Object Migrator versions, and which GL Migrator versions the Extension supports, see the *System Requirements and Compatibility Matrix*.

The installation process for the Extension copies various entities into the PPM Center database, including the following:

- Menus in the standard interface and preconfigured Dashboard pages that are specifically designed for Oracle E-Business Suite
- Oracle Applications or Ora Apps tabs (for Oracle E-Business Suite) in the Environment, Object Type, and User windows
- A work plan template
- Object types
- Report types
- Request types
- Workflows
- Security groups
- Validations

After PPM (the “base” software) has been installed at or upgraded to version 9.40, you can install the Extension or upgrade it to version 9.40 on the same system.

["Installing or Upgrading the Extension" on page 27](#) describes system requirements for installing the Extension or upgrading it to version 9.40, and the impacts of upgrading the Extension to version 9.40.

What’s New and What’s Changed in Version 9.50

Extension version 9.50 makes no functional changes compared to version 9.40. Upgrading to version 9.50 does not affect existing Extension functionality.

After installing PPM 9.50, you can install Extension version 9.50 for the first time, or you can upgrade the Extension from version 9.40.

Note:

PPM supports having multiple languages on the same instance, including translations of specific PPM entities and interfaces. Micro Focus does not provide translations of any Extension entities or interfaces with the Extension or any language pack.

However, like any customer-defined entity, Extension entities such as request types can be translated by using the `kExportAttributes.sh` and `kImportAttributes.sh` scripts as part of a translation process. For more information, see the Multilingual User Interface Guide.

["Installing or Upgrading the Extension" on page 27](#) describes general impacts of upgrading the Extension from version 9.40 to version 9.50.

About This Document

This document is intended for the following audiences:

- People responsible for installing, upgrading, configuring, and maintaining the PPM system and database environments where the Extension resides, for setting up and maintaining the PPM schema, and for integrating PPM with one or more Oracle E-Business Suite instances
- System or instance administrators who maintain access and security for, or support use of the Extension
- People responsible for managing Oracle-related projects
- People responsible for work associated with patching Oracle instances, including analysis, application and evaluation, and reporting
- Developers or others responsible for creating customizations of Oracle-related objects and promoting them to different instances
- People responsible for managing or participating in the development process or issue management process

For more information about business roles in connection with Oracle E-Business Suite, see ["Business Roles" on page 22](#).

This guide provides information about installing or upgrading to version 9.50 of Deployment Management Extension for SAP Solutions, and this guide provides conceptual, procedural, and reference information about the product.

This guide is organized as follows:

- ["Getting Started" on page 15](#) provides an introduction to the Extension, the features and changes introduced in version 9.50, information about the intended audiences for this document, related information, prerequisite knowledge including integration of the Extension with Oracle E-Business Suite, and business concepts used in the Oracle environment.
- ["Installing or Upgrading the Extension" on page 27](#) provides overview and detailed information about installing or upgrading the Extension, including information about the impacts of upgrading the Extension.
- ["Configuring the Extension" on page 40](#) provides information about configuring the Extension after installing or upgrading it, and about managing integration with Oracle E-Business Suite.
- ["Extension Interface" on page 67](#) provides information about preconfigured PPM Dashboard pages, environment definitions, and other interface elements of the Extension.
- ["Managing Issues" on page 88](#) provides information about the Extension's issue management functionality.
- ["Managing Projects" on page 92](#) provides information about the Extension's project management functionality.
- ["Managing Changes" on page 107](#) provides information about the Extension's functionality for managing changes to Oracle E-Business Suite.

- ["Managing Instances" on page 119](#) provides information about the Extension's instance management functionality.
- ["Managing Patches" on page 130](#) provides information about the Extension's patch management functionality.
- ["Object Types" on page 159](#) provides reference information about the Extension's object types.
- ["Request Types" on page 207](#) provides reference information about the Extension's request types.
- ["Workflows" on page 237](#) provides reference information about the Extension's workflows.
- ["Report Types" on page 280](#) provides reference information about the Extension's report types.
- ["Tokens" on page 314](#) provides reference information about the Extension's tokens.

Related Information

The following documents also include information useful in managing Deployment Management Extension for Oracle E-Business Suite:

- *Installation and Administration Guide*
- *System Requirements and Compatibility Matrix*
- *Security Model Guide and Reference*
- *What's New and What's Changed*
- *Deployment Management Configuration Guide*
- *Deployment Management User's Guide*

The following additional Extension and Migrator documentation for Oracle environments might be of particular interest:

- *Object Migrator Guide*
- *GL Migrator Guide*
- *Deployment Management Extension for Oracle Technology Guide*

Prerequisite Knowledge and Experience

To install, upgrade, configure, maintain, or use the Extension, you need to understand the following:

- Deployment management
- Environments
- Software deployment

- Packages
- PPM Dashboard pages and portlets
- PPM Workbench
- Object types
- Request types
- Workflows and workflow steps
- Report types
- Special commands
- Validations
- Tokens
- PPM entities installed by the Extension

In addition, you must have practical experience in the following:

- Installing, upgrading, configuring, and using PPM, if you are responsible for configuring the Extension
- Using SAP Solutions

Integration of the Extension with Oracle E-Business Suite

Deployment Management Extension for Oracle E-Business Suite, which is installed on the PPM Server, integrates tightly with Oracle E-Business Suite to submit and monitor Object Migrator and GL Migrator concurrent requests, as well as certain CONCSUB requests.

To be fully functional, the Extension must communicate and integrate with the instance of Oracle E-Business Suite on which Object Migrator and, optionally, GL Migrator are installed and run by default. This instance is known as the primary Object Migrator host. Parameters in the PPM Center configuration file (`server.conf`), for example `com.kintana.core.server.CONC_REQUEST_USER`, refer to this Oracle E-Business Suite instance.

A remote Object Migrator host is another Oracle E-Business Suite instance on which Object Migrator and, optionally, GL Migrator are installed. For more information, see ["Remote Execution of Object Migrator Concurrent Requests" on page 49](#).

The following methods can be used to integrate the Extension with Oracle E-Business Suite:

- Local integration (single database)
- Linked database integration

- No integration

Before beginning installation of the Extension, decide upon the integration method to be used. The integration method can be changed after installation, but changing it requires stopping and restarting the PPM Server. For information about changing the current integration method, see ["Changing the Integration Method to Local" on page 62](#) or ["Changing the Integration Method to Linked" on page 63](#).

The following sections describe integration methods. For information about how to install Object Migrator and how to set up database links, see the *Object Migrator Guide*.

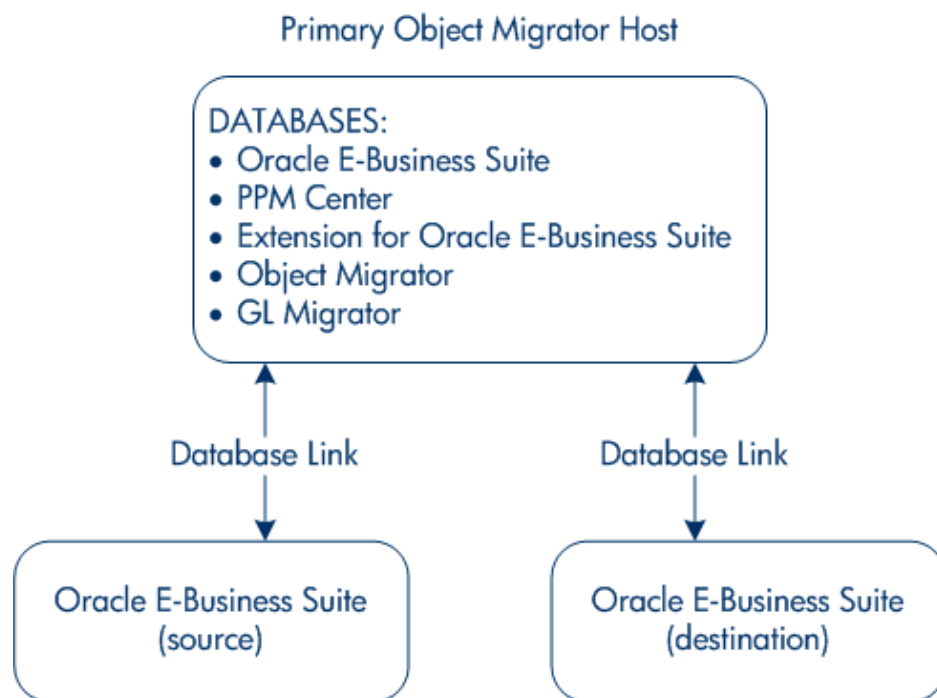
Note:

These integration options do not affect the configuration or use of remote Object Migrator functionality. Use a remote Object Migrator host instead of the primary Object Migrator host as necessary to accomplish specific migrations. See ["Remote Execution of Object Migrator Concurrent Requests" on page 49](#).

Local Integration (Single Database)

In local integration, PPM and the primary Object Migrator host use the same database, as shown in ["Figure 1-1. Local integration of Oracle E-Business Suite and the Extension"](#).

Figure 1-1. Local integration of Oracle E-Business Suite and the Extension

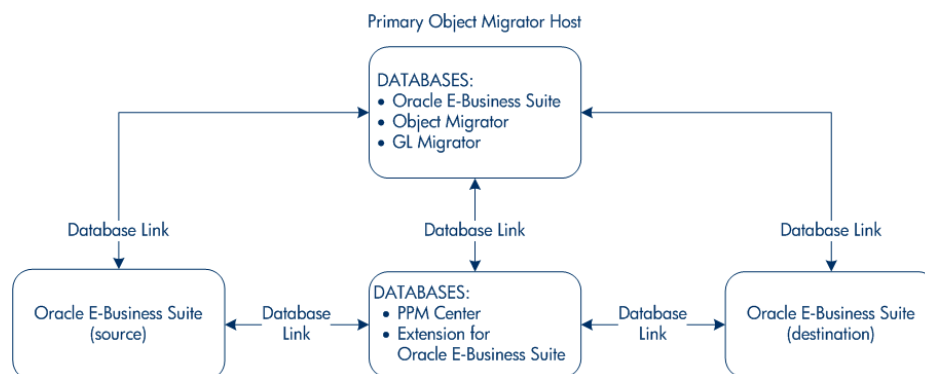


This configuration provides the most efficient integration of PPM Center with Oracle E-Business Suite, and should provide the best performance in high-volume situations. However, having two separate applications using the same database might conflict with company policies, or you might find that there are conflicts between the needs of PPM Center (for example, database patch levels or parameters) and how you want to maintain your Oracle E-Business Suite instance. In this case, we recommend a different integration option.

Linked Database Integration

In linked database integration, PPM and the primary Object Migrator host use separate databases, and they communicate using database links, as shown in ["Figure 1-2. Linked database integration of Oracle E-Business Suite and the Extension"](#).

Figure 1-2. Linked database integration of Oracle E-Business Suite and the Extension



While linked database configuration provides more flexibility than local integration, it might be less efficient because of reliance on SQL*Net for database link connectivity. If the primary Object Migrator host is unavailable (due to downtime, network configuration issues, or other reasons), some Extension functionality (including integration with Object Migrator or GL Migrator) is unavailable. Linked integration might also require more administration to manage downtime windows (see ["Managing Linked Integration During Oracle E-Business Suite Downtime" on page 64](#)).

Business Concepts Used in the Oracle Environment

The following sections describe business roles and other business concepts that apply to the Oracle E-Business Suite environment.

Business Roles

To manage their work, organizations operating in the Oracle E-Business Suite environment define business roles that are differentiated by their skill sets and specific responsibilities. The roles usually vary by project or implementation size. The following sections describe the typical business roles used in managing Oracle E-Business Suite with Deployment Management Extension for Oracle E-Business Suite.

Project Manager

A project manager has overall responsibility for a project, the accuracy of the work plan, and delivery of the project on time and within budget. A person in this role defines the project structure and is responsible for ensuring that adequate resources (that is, people and equipment) are available to the project at the time they are required.

The project manager actively manages project issues. If necessary, the project manager escalates issues to the project sponsor and works with appropriate IT and business resources (inside or outside of the project) to achieve resolution.

The project manager is usually a project management or IT role, and there is often more than one person involved in this work. This person interacts with the project sponsor and with external organizations such as operations, networking, and vendors. When using Deployment Management Extension for Oracle E-Business Suite, the project manager manages tasks using Project Management, and might own all risk- or project-related issues.

Project Sponsor

A project sponsor is the overall sponsor for the project. The person in this role is usually a high-level business representative, often a director or vice president in a business area directly impacted by Oracle E-Business Suite. The project sponsor is responsible for ensuring overall company buy-in to the new or changed system, and for ensuring that any critical process or policy issues are addressed in a timely manner.

Major process changes in an organization (which usually entail changes in policy, or require some type of external reporting) usually require project sponsor approval. The project sponsor is often responsible for the overall project budget, approval of implementation cutover, and “go” or “no-go” decisions.

Project sponsors can have varying degrees of interaction with the project team, depending on project size and organizational structure, and are likely to be the primary approvers of initial project scope and changes to project scope.

Track Owner

A track owner oversees the work of a group of related tasks on the project, such as conversions, financials, interfaces, or reporting. Track definitions are project specific. A track can involve more than one departmental business process. The person in this role is responsible for detailed work plans for the governed area, and identification and resolution of issues relating to the track.

A track owner is usually in a project management or IT role. Responsibilities include the following:

- Proactively providing input to the project manager
- Defining standards or templates to be used for the track
- Coordinating cross-functional dependencies within the track
- Identifying and coordinating cross-track dependencies
- Understanding the relationship of the track’s efforts to the larger project

For mid-sized projects, track owners might be responsible for design review, ownership of the code, and resource sharing across different tracks. Track owners might also prioritize and assign incoming requests to developers.

Database Administrator (DBA)

A database administrator (DBA) manages Oracle databases. This role can also have responsibility for the following work:

- Administering the entire Oracle E-Business Suite installation, including maintenance and patching
- Determining the Oracle configuration to be used and implementing the configuration
- Ensuring the overall safety and security of system data, including mechanisms for backup and recovery
- Developing and enforcing performance- or data model-related coding standards for SQL or PL/SQL

Business Lead

A business lead is the representative of one or more related business processes or sectors within the company. Larger organizations might have more than one business lead.

The primary responsibility of business leads is to determine the accuracy of business requirements given to the project team, and to prioritize those business requirements, both within their own area of expertise and across the project. Business leads are directly involved in project scoping, including review and approval of scope changes and late enhancement requests. They help to ensure that a new system meets business requirements.

Business leads are usually responsible for the acceptance of functional designs, approval of test outcomes, and providing input to “go” or “no-go” decisions. They identify project risks and issues, and actively work to resolve them. They are business counterparts of track owners and work closely with them. They often contribute to or write functional test cases.

Team Lead

A team lead directs the work of a (generally small) group of developers or analysts during various phases of the project. The team lead focuses on a relatively narrow functional area of a project such as accounts payable, electronic funds transfer (EFT) interfaces, or order management. Team leads are often technical experts in their areas of focus, providing guidance and help as team members analyze and develop solutions.

A team lead is usually responsible for technical and functional review of designs and code, and might have input to templates and standards in use by the team. The team lead's responsibilities might include the following:

- Coordinating low-level team efforts and proactively identifying issues or risks
- Understanding the relationship of the team's efforts to the larger project
- Generating design documents (usually for small projects)
- Conducting gap analysis and solution proposal
- Generating accurate time estimates for proposed solutions or development work

The distinction between track owner and team lead is based on the scale of a project. Larger projects might include separate roles for functional lead (track owner) and technical lead (team lead).

Developer

A developer is responsible for creating a particular segment of functionality (for example, a report or form). This role might include analysis tasks if a separate analyst role does not exist in the organization. Developers must be technically skilled in the development tools used on the project and have technical understanding of the data structures that are integrated with the functionality. Developers must also possess a basic functional understanding of the business process related to their area of development.

Developer responsibilities often include the following:

- Creating design documents
- Code development and unit testing
- Technical gap analysis and solution proposals
- Detailed effort estimates

Analyst

An analyst is responsible for translating business requirements into integrated functional definitions. This role is usually responsible for mapping requirements to Oracle functionality, and then identifying and analyzing gaps. These individuals might also provide level-of-effort estimates for development. In some organizations this role is combined with the developer role.

Additional Business Concepts

The following sections describe business areas, sources, and tracks.

Business Area

A business area is a major business process or group in operation at a company, such as order management, finance, or shipping. The business area usually identifies a grouping of functional areas for which there is a particular business or IT owner responsible for prioritizing and managing requests. These functional areas are usually groupings of multiple, related applications. A business lead is usually responsible for one or more business areas.

Source

A source refers to the type of work that was underway when a problem was identified. Examples of sources are integration test and production use.

Track

A track is a group of related work within a project, such as conversions, financials, interfaces, and reporting.

Installing or Upgrading the Extension

This section includes the following topics:

- ["Overview of Installation and Upgrade" below](#)
- ["Preparing for Installation or Upgrade" on page 30](#)
- ["Installation Procedure" on page 32](#)
- ["Installing the Extension's Preconfigured Dashboard Pages" on page 36](#)

Overview of Installation and Upgrade

The following sections describe system requirements that must be met before installing or upgrading the Extension, and upgrade impacts.

System Requirements

The following sections specify requirements that must be met by the instance on which PPM will be installed or upgraded to PPM 9.50, and by the Oracle E-Business Suite instances.

PPM

Deployment Management Extension for Oracle E-Business Suite requires the following in regard to PPM:

- Before you install the Extension or upgrade it to version 9.50, PPM Center version 9.50 must be installed. PPM and the Extension are installed on the same system and have the same system requirements.

For information about installing PPM version 9.50, see the *Installation and Administration Guide*. For information about upgrading PPM to version 9.50, see the *Upgrade Guide*.

For information about version levels and other general system requirements, see the *System Requirements and Compatibility Matrix*, available at the PPM Help Center: <http://admhelp.microfocus.com/ppm/>

- Connectivity must be provided to each Oracle E-Business Suite database using a database link from PPM.
- The `open_links` database parameter (which is set in the `init.ora` file) must support the number of database links defined in the PPM schema.
- All Object Migrator and GL Migrator concurrent requests initiated from PPM must be processed to completion before starting an upgrade.

Oracle E-Business Suite Instances

The following sections describe system requirements for the following Oracle E-Business Suite instances:

- Primary Object Migrator host (the instance of Oracle E-Business Suite that hosts Object Migrator and GL Migrator)
- Source or destination instances of PPM activity

Primary Object Migrator Host

The primary Object Migrator host must meet the following criteria:

- The Oracle E-Business Suite release must be as specified in the *System Requirements and Compatibility Matrix*.
- Application Object Library (AOL) must be installed and fully functional.
- If you are using GL Migrator, Oracle General Ledger must already be installed.
- For continuity of the Object Migrator and GL Migrator installations, the database must not be refreshed.
- The primary Object Migrator host can use the same database as PPM or a different database.
 - For local integration, the databases for the primary Object Migrator host and PPM reside on the same server, and both databases must meet the general database specifications of PPM Center. See the Installation and Administration Guide.
 - For linked integration, the databases for the primary Object Migrator host and PPM Center reside on different servers, and the PPM schema must have a fully functional database link to the APPS account of the primary Object Migrator host.

You must decide upon the integration method to be used before you install the Extension.

For more information about integration methods, see Integration of the Extension with Oracle E-Business Suite on page 24.

- Any remote Object Migrator hosts must allow connectivity using a database link from the PPM schema.

For more information about remote migrations, see Remote Execution of Object Migrator Concurrent Requests on page 62.

- FTP or SCP connectivity must be allowed for retrieval of Concurrent Request Logs.

Oracle E-Business Suite Instances as Sources or Destinations of PPM Center Activities

Oracle E-Business Suite instances that are the sources or destinations of activities controlled by PPM must meet the following criteria:

- The Oracle E-Business Suite release must be as specified in the *System Requirements and Compatibility Matrix*.
Specific Extension functionality might have additional release requirements.
- You must allow connectivity using the database link from the PPM schema.
- You must allow “dumb terminal” login at the operating system level with a bash-compatible shell.
- To migrate PPM workflows related to the Oracle environment, you must have a UTL_FILE_DIR definition in the database init.ora file.
- FTP or SCP connectivity must be provided for file transfer.
- The logon used for servers must have the following:
 - Read/write access to Oracle E-Business Suite directories.
 - A method to set an Oracle E-Business Suite context from the command line in a bash-compatible shell.

For information about instance cloning, see ["Managing Instances" on page 119](#).

For potential additional requirements related to Object Migrator and GL Migrator, see the following documents:

- Installation and Administration Guide
- *Object Migrator Guide*
- *GL Migrator Guide*

General Upgrade Impacts and Guidelines

Each new version or service pack of the Extension provides the following types of entities, which the Extension upgrade process installs or does not install in the PPM instance, as described:

- **Reference entities.** The names of reference entities start with (REFERENCE).
You cannot edit reference entities. However, you can copy and rename them and then edit the copies as non-reference entities, as appropriate for your environment.

The upgrade process deletes all of the Extension’s existing (previously installed) reference entities, such as its object types, and then the process installs a new set of reference entities for the new version. The new set of reference entities can be identical to the previous set or, to support changes in functionality, the new set can have new, deleted, renamed, or modified reference entities.
- **Non-reference entities.** The non-reference entities and corresponding reference entities that the version or service pack provides are identical, except that the

names of the reference entities start with (REFERENCE).

You can edit (or copy and edit) non-reference entities, as appropriate for your environment. The currently installed version of the Extension might include non-reference entities that have been edited (customized).

To preserve all existing customizations, the upgrade process compares the names of the non-reference entities delivered with the new version to the names of existing non-reference entities in the instance, and if the process finds an existing non-reference entity with the same name, the process does not overwrite or modify that entity in the instance. If the new version introduces reference entities with new names along with non-reference copies of those entities, then the upgrade process adds both the reference and non-reference copies.

In general, if an upgrade changes an Extension's reference entities, you must evaluate how those changes should affect the associated, previously customized non-reference entities. After the upgrade is performed, you can revise those non-reference entities or create new ones as necessary.

Specific Upgrade Impacts for Version 9.50

Upgrading to version 9.50 does not affect existing Extension functionality.

Preparing for Installation or Upgrade

Prepare for installation or upgrade of the Extension as described in the following sections.

Note:

During installation or upgrade, the PPM Server must run in restricted mode.

General Preparations for Installation or Upgrade

To prepare for installation or upgrade of the Extension:

1. Obtain the Extension software.
2. Decide which Oracle E-Business Suite instance will be your primary Object Migrator host and whether it will use local integration or linked integration with PPM.

The integration method can be changed after installation, but changing it requires stopping and restarting the PPM Server.

The Extension can also be installed with no integration, but Object Migrator- and GL Migrator-related functions will not be available.

For more information about integration methods, see ["Integration of the Extension with Oracle E-Business Suite" on page 19](#).

3. Install and configure Object Migrator on the primary Object Migrator host, if it is not already installed. If the current version of Object Migrator is not installed, we recommend that you upgrade to the current version before or after upgrading the Extension.

For more information about installing Object Migrator, see the *Object Migrator Guide*.

4. Install and configure GL Migrator on the primary Object Migrator host, if it is not already installed. If the current version of GL Migrator is not installed, we recommend that you upgrade to the current version before or after upgrading the Extension.

For more information about installing GL Migrator, see the *GL Migrator Guide*.

5. Collect the following information, which you will need to supply during the installation procedure:
 - The logon username and password for the Extension's server (the same server on which PPM Center is or will be installed). The username (typically "admin") must belong to a security group that has the following access grants:
 - Sys Admin: Migrate PPM Objects
 - Sys Admin: Server Administrator
 - The database password for the server's schema.
 - For local integrations, the database name and password for the Oracle E-Business Suite APPS user on the primary Object Migrator host.
For linked integrations, the name of the database link, defined in the database schema of the Extension's server, that connects to the APPS account of the primary Object Migrator host.
For more information about integration methods, see ["Integration of the Extension with Oracle E-Business Suite" on page 19](#).

6. Log on to the PPM Server.
7. Verify that the system requirements have been met. See ["System Requirements" on page 27](#).
8. Save the Extension installation file (ppm-940-OracleApps.jar) to one of the following directories:
 - `<PPM_Home>`(the recommended location).
 - `<PPM_Home>` represents the path where your PPM instance is installed. For example:xyzserver/E/PPMServer.
 - A subdirectory of `<PPM_Home>`. If a subdirectory of `<PPM_Home>` is specified in the `ITG_DEPLOYMENT_HOME` environment variable, the installation script finds and uses the installation file in that subdirectory. If you want to save the installation file to a subdirectory of `<PPM_Home>`, make sure the

value of the `ITG_DEPLOYMENT_HOME` environment variable is `<PPM_Home>` followed by that subdirectory, for example, `<PPM_Home>/Extension`.

- Any directory (with, optionally, any subdirectories) you choose, for example, `dirA/sub1/sub2`.
9. In the database schema of the Extension's server, define a database link to the APPS account of the primary Object Migrator host, and test the link.
 10. Decide whether separate tablespaces will be used for patch data captured by the Extension. If so, create them.

To reduce the potential for performance impacts or space contention, specify separate tablespaces for patch analysis and grant unlimited access on the tablespaces to the database schema.

The required size for the tablespaces depends on your expected patching activity. For example, patch detail data for the 11.5.7 maintenance pack requires 50 megabytes if no other patch data is included.

We recommend initial allocations of 100 megabytes for tables and 125 megabytes for indexes.

Performing Backup and Restarting the Server in Restricted Mode

The steps in this section are recommended but not required.

Note:

For more information about the steps in this procedure, see the *Installation and Administration Guide*.

For a new installation or an upgrade, do the following:

1. Back up the database and file system for the PPM Server.
2. To stop the PPM Server and restart it in restricted mode:
 - a. Stop the PPM Server.
 - b. Run the following script:

```
sh ./setServerMode.sh RESTRICTED
```
 - c. Start the PPM Server.

Installation Procedure

Perform the procedures in the following sections to install the Extension.

Run the Installation Script and Check the Logs

To run the installation script to install the Extension, and to check the logs:

1. Be sure you have completed all the steps in ["Preparing for Installation or Upgrade" on page 30](#).
In particular, be sure the PPM Server is running in restricted mode. See ["Performing Backup and Restarting the Server in Restricted Mode" on the previous page](#).
2. On the PPM Server, navigate to the bin subdirectory of the <PPM_Home> directory (or other directory as described in [" Save the Extension installation file \(ppm-940-OracleApps.jar\) to one of the following directories:" on page 31](#)).
3. Start the installation or upgrade. In step 5 on page 18, if you saved the installation file to <PPM_Home> , which is the recommended directory, or to a subdirectory of <PPM_Home>, run the following script:

```
sh ./kDeploy.sh -i OracleApps
```

However, if you saved the installation file to a different directory, see the example in step 5 on page 18 and specify that directory in the script command, as in the following example:

```
sh ./kDeploy.sh -i OracleApps -D dirA/sub1/sub2
```

4. Follow the script's on-screen prompts to complete the installation. Prompts can include the database password for the schema and the logon name and password for the server.
Files are installed in various subdirectories under <PPM_Home>. Data is also placed in the database. When the installation script is complete, the following message appears:

```
Deployment OracleApps has been successfully installed.
```

5. Use a Web browser to check the installation summary report, which is located at:

```
<PPM_Home>/logs/deploy/940/oracleApps/<Log_x>/installLog.html
```

where <Log_x> is initially a random number generated by kDeploy.sh during installation. The number increments by one each time the installation script is run, so the installation summary report for the most recent run is in the directory with the highest number. The installation summary report lists all the entities that are installed as part of the Extension installation process. Each entity that was installed correctly is marked as Complete. If there is an error for a particular entity, the report contains a direct link to another log file (HTML page) with additional information.

If necessary, correct any errors and repeat the installation process.

Installation of the Extension generates the logs that are described in ["Logs Generated During Installation"](#), depending on the installation options.

Logs Generated During Installation

Depending on the installation options that were chosen, the logs listed and described in "Table 2-1. Logs generated during installation" below can be generated during installation and saved in the following directory:

<PPM_Home>/logs/deploy/940/OracleApps

The log number (<#####>) shown in "Table 2-1. Logs generated during installation" below is a random number (generated by kDeploy.sh) that makes each log file name unique.

Table 2-1. Logs generated during installation

File Name	Description
ddlDriver.<#####>.log	Contains information about data model changes made during installation
fnd_apps_grants.<#####>.log ^a	Contains information about the creation of database grants from the APPS account of a local integration to the PPM Center schema.
fnd_knta_synonyms.<#####>.log ^a	Contains information about the definition of synonyms in the PPM Center schema when the primary Object Migrator host shares the same database as PPM Center. If the connectivity to the primary Object Migrator host was linked but is now local, this log contains a message to that effect.
fnd_linked_synonyms.<#####>.log ^a	Contains information about the refresh of connectivity to a linked primary Object Migrator host residing in a database different from PPM Center, using the database link provided at the start of installation. If the database link used for connectivity changes, this log contains a message to that effect.
preXMLDriver.<#####>.log	Contains information about the application of SQL scripts required before the installation of Micro Focus-supplied data, such as the definition for Deployment Management Extension for SAP Solutions

Table 2-1. Logs generated during installation, continued

File Name	Description
jarxvf.<#####>.log	Contains information from the procedure that unpacks the .jar file
log_<n>/installLog.html	Contains log for importing the modules if the prerequisites were met, where <n> is a sequential number defining a new subdirectory for each time the oraapps_modules.sh script is run.
modules.check.<#####>.log	Indicates whether prerequisites were met for installing the preconfigured Dashboard pages (modules) using the oraapps_modules.sh script, where <#####> is a random number generated by that script.
packageDriver.<#####>.log	Contains information about the installation of database code; for example, reports.
postXMLDriver.<#####>.log	Contains information about the application of SQL scripts required after the installation of Micro Focus-supplied data.
preXMLDriver.<#####>.log	Contains information about the application of SQL scripts required before the installation of Micro Focus-supplied data, such as the definition for Deployment Management Extension for SAP Solutions.
recompile_invalid.<#####>.log	Lists any objects made invalid by prior scripts and recompiled.
a. This file might or might not exist, depending on the installation options that were chosen.	

Verify the Installation

We strongly recommend that you verify correct installation. To verify that Extension version 9.50 for Oracle E-Business Suite is listed among the installed Extensions, navigate to the <PPM_Home>/bin directory and run the following script:

```
sh ./kDeploy.sh -l
```

where the last character in the command is the lowercase letter “l.”

The name OracleApps should appear in the list of installed Extensions.

For example, if both Deployment Management Extension for Oracle E-Business Suite and Deployment Management Extension for Oracle Technology are now installed at version 9.50, the following table entries are displayed:

Deployment	Version	Deployed	Description
oracleApps	950	<date and time>	Oracle Apps Extension
OracleTech	950	<date and time>	OracleTech Extension

Restart the PPM Server in Normal Mode

Note:

For more information about the steps in this procedure, see the *Installation and Administration Guide*.

After you have completed all installation or upgrade procedures, if you previously restarted the PPM Server in restricted mode, to stop and restart it in normal mode:

1. Stop the PPM Server.
2. Run the following script:

```
sh ./setServerMode.sh NORMAL
```

3. Start the PPM Server.

Note:

After the Extension has been running successfully for a substantial period of time, you can optionally delete all of the installation files. However, we recommend that you retain (or copy) the log files.

Installing the Extension's Preconfigured Dashboard Pages

After installing the Extension, you can install its preconfigured Dashboard pages (also known as modules), listed in "[Table 2-2. Preconfigured Dashboard pages for the Extension and their security groups](#)". Each user who belongs to the indicated security group can then add these pages to his or her PPM Dashboard.

Table 2-2. Preconfigured Dashboard pages for the Extension and their security groups

Preconfigured Dashboard Page	Associated Security Group
Oracle Apps Project Manager	OA - Management
Oracle Apps Project Sponsor	OA - Sponsor
Oracle Apps Reports Track Owner	OA - Track Owner
Oracle Apps DBA	OA - Oracle Patch Admin

Note:

If you have performed an upgrade from version 9.40, and if the modules were previously installed, you do not need to reinstall them. They have not been changed since version 9.40 and they will continue to work in version 9.50. In this case, proceed to ["Configuring the Extension" on page 40](#).

To install the preconfigured Dashboard pages:

1. Enable all of the Extension's request types, as follows:
 - a. Log on to PPM.
 - b. From the menu bar, select **Open > Administration > Open Workbench**. The PPM Workbench opens.
 - c. From the shortcut bar, select **Demand Mgmt > Request Types**. The Request Type Workbench opens.
 - d. Open a request type.
 - e. In the **Enabled** field in the Request Type Workbench, click **Yes**.
2. To open the OraApps Release11i Upgrade Work Plan Template, which works for all Oracle E-Business Suite releases supported by the Extension:
 - a. From the menu bar, select **Open > Administration > Project Types & Templates > Manage Work Plan Templates**.
The Manage Work Plan Templates page opens.
 - b. Select **OraApps Release11i Upgrade Work Plan Template**.
The OraApps Release11i Upgrade Work Plan Template opens
3. To keep the original work plan template available, we recommend that you copy it, then rename and revise the copy, as follows:
 - a. Click **Copy** to copy the template. A new work plan template is created with the name Copy of OraApps Release11i Upgrade Work Plan Template.
 - b. Rename the work plan template as desired. For example, you might rename the new template (that is, the copy) OraApps Release11i Upgrade Work

Plan Template Rev1.

- c. Revise this new work plan template as desired.
4. To create a project with the new work plan template:
 - a. From the menu bar, select **Project Management > Projects & Tasks > Create a Project**.
The Create New Project window opens.
 - b. Complete the fields in the Create New Project window. For the **Project Type** field, select **Enterprise**.
 - c. Click **Create**.
The Project Overview page opens, with the **Project Summary** tab selected.
 - d. Click the **Project Details** tab of the Project Overview page.
 - e. Click **Launch** to make the project active.
 - f. Click the **Project Summary** tab of the Project Overview page.
 - g. In the **Work Plan** section of the **Project Summary** tab, click **Create Work Plan from a Template** and select the new work plan template you created in "[To keep the original work plan template available, we recommend that you copy it, then rename and revise the copy, as follows:](#)".
 - h. Click **Create**.
The work plan based on the new work plan template opens.
 - i. Open (double-click) the first row of the work plan, change its **Status** field to **Active**, and click **Done**.
5. To enable and add the security groups for the user who will install the preconfigured Dashboard pages:
 - a. Log on to PPM.
 - b. From the menu bar, select **Open > Administration > Open Workbench**. The PPM Workbench opens.
 - c. From the shortcut bar, select **Sys Admin > Security Groups**.
The Security Group Workbench opens.
 - d. Click **List** to list the security groups.
 - e. Open the following security groups and select the Enabled option for them:
 - o OA - Management
 - o OA - Oracle Patch Admin
 - o OA - Sponsor
 - o OA - Track Owner
 - f. Add these security groups to the user who will install the preconfigured Dashboard pages in "[To install \(import\) the preconfigured Dashboard pages, from the <PPM_Home>/bin directory, run the following script:](#)".

6. To install (import) the preconfigured Dashboard pages, from the `<PPM_Home>/bin` directory, run the following script:

```
sh ./oraapps_modules.sh
```

As the script runs, respond to its prompts.

When the installation procedure is complete, the console log shows that it was successful.

Proceed to ["Configuring the Extension" on page 40](#).

Configuring the Extension

This section includes the following topics:

- ["Overview of Configuration" below](#)
- ["Update the server.conf File" below](#)
- ["Configuring Database Links" on the next page](#)
- ["Configuring Environments" on the next page](#)
- ["Configuring Oracle E-Business Suite Users" on page 52](#)
- ["Configuring Security Groups" on page 53](#)
- ["Configuring Request Types" on page 54](#)
- ["Configuring Workflows" on page 55](#)
- ["Configuring Object Types" on page 58](#)
- ["Managing Integration with Oracle E-Business Suite" on page 61](#)

Overview of Configuration

Configuring the Extension involves the following:

- Updating the `server.conf` file
- Configuring a database link in the PPMr schema to the APPS account of each Oracle E-Business Suite instance that is a source or destination of Object Migrator migrations
- Configuring a variety of environments
- Configuring Oracle E-Business Suite users
- Configuring security groups
- Enabling and configuring request types
- Enabling and configuring workflows
- Enabling and configuring object types

You must also manage integration with Oracle E-Business Suite as desired.

Update the `server.conf` File

Use of Object Migrator and GL Migrator object types requires configuring particular parameters in the `server.conf` file (described in the *Installation and Administration Guide*) in PPM as follows:

1. Stop the PPM Server.

Note:

All `server.conf` parameter names begin with `com.kintana.core.server.`

2. Set `CONC_REQUEST_USER` and `CONC_REQUEST_PASSWORD` to a user with access to the log and output directories for concurrent requests for the primary Object Migrator host.
3. Set `CONC_LOG_TRANSFER_PROTOCOL` to an allowed protocol to retrieve files from the primary Object Migrator host's concurrent request directories.
The default value is `FTP`.
4. Set `ORACLE_APPS_ENABLED` to `True`.
5. Set `SERVER_ENV_NAME` to the environment name of the PPM Server. The default is `KINTANA_SERVER`.
6. Restart the PPM Server.

Configuring Database Links

A valid database link must be defined in the PPM schema to the APPS account of each Oracle E-Business Suite instance that is a source or destination of Object Migrator migrations.

The Extension must integrate with the Oracle E-Business Suite instance that runs Object Migrator (and optionally GL Migrator) requests by default, that is, the primary Object Migrator host. Otherwise, users might be unable to complete the environment setups required to submit Object Migrator requests, or they might select invalid data for the setups.

The required setup is primarily at the database level, and the PPM Server must be restarted whenever changes are made in this configuration.

For information about integration configurations, see [Integration of the Extension with Oracle E-Business Suite](#) on page 24.

Configuring Environments

Before you can use the Extension, in the PPM Workbench you must configure an environment for each of the following:

The primary Object Migrator host.

- Each Oracle E-Business Suite instance that is a source or destination for Object Migrator, a workflow, or an AK Web Setup migration.
- An environment corresponding to the `SERVER_ENV_NAME` configuration parameter, to represent the PPM Server and to move log files to the PPM Server.
- A single common repository for all Oracle E-Business Suite patch archives. The Extension looks here to find the patch to be applied. The environment name must be `Patch Stage`.

Note:

If you use environment groups in your workflows and you migrate AOL or GL information, the following must be done:

- The environment groups must be configured for serial execution in order for the Object Migrator or GL Migrator dependencies to be considered.
- All environments in the group must use the same Object Migrator or GL Migrator instance. For example, a mixture of remote Object Migrator and local Object Migrator executions or a mixture of different Object Migrator hosts is not allowed.

To configure an Oracle E-Business Suite environment:

1. Log on to PPM
2. From the menu bar, select **Open > Administration > Open Workbench**. The PPM Workbench opens.
3. From the shortcut bar, select **Environments > Environments**. The Environment Workbench opens.
4. Open a new environment or, if the Oracle E-Business Suite environment of interest already exists, open it.

The Environment window opens to the **Host** tab by default.

5. As required for the Oracle E-Business Suite environment, do the following:
 - Configure the **Host** tab. See **Configuring the Host Tab**.
 - Click the **Extension Data** tab and then, at the bottom of the Environment window, click the **Oracle Applications** subtab and configure it. See **Configuring the Extension Data Tab, Oracle Applications Subtab on page 57**.

Configuring the Host Tab

On the Host tab of the Environment window, complete the fields in the **Server** section. "[Figure 3-1. Host tab sample data](#)" shows the Host tab of a sample environment. "[Table 3-1. Environment window, Host tab fields](#)" lists the field names and their descriptions.

Figure 3-1. Host tab sample data

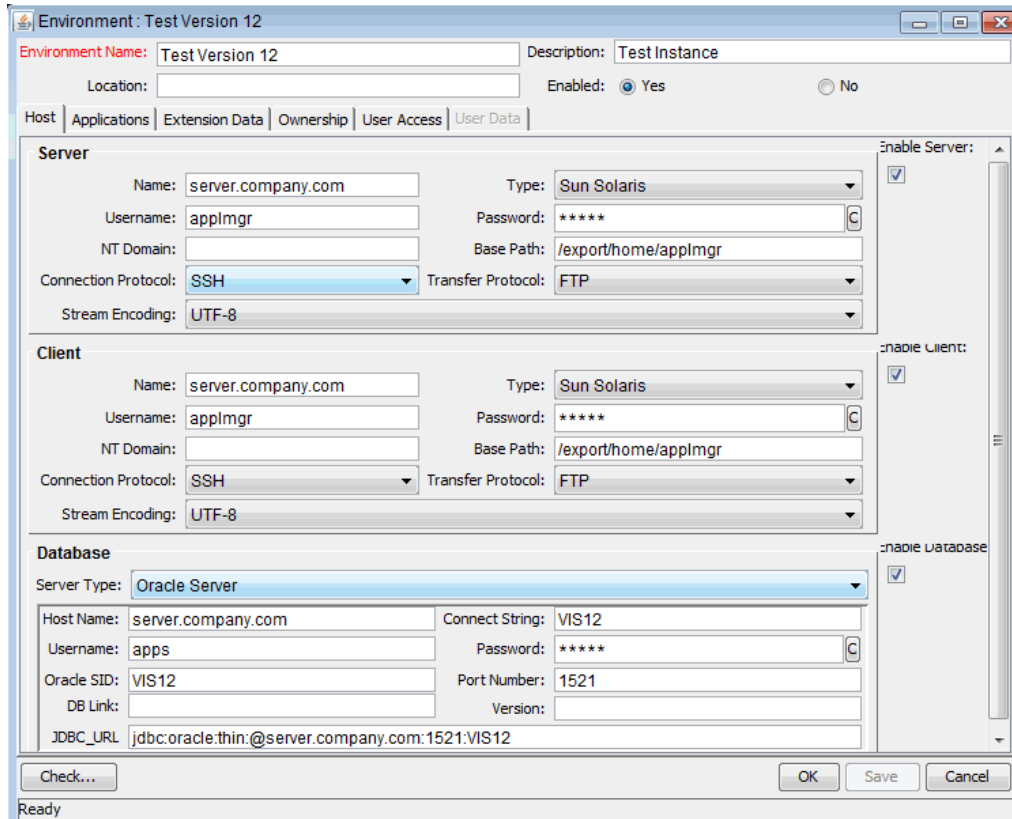


Table 3-1. Environment window, Host tab fields

Field Name	Value or Description
Server section	
Enable Server	Select this check box so that fields in the Server section can be configured.
Name	Host name of the server for the Source or Destination environment.
Type	Type of hardware and operating system for the host.
Username	Valid Username that has read/write access to the database UTL_FILE_DIR directory (for workflow migration) and has rights to execute WFLOAD and AKLOAD. Typically, this would be APPLMGR.
Password	Password for the Username.
NT Domain	The domain name for a Windows Server.
Base Path	Login directory for the user, or the APPL_TOP for the instance. The user must have read/write rights to the directory and should be able to set the Oracle context information.

Table 3-1. Environment window, Host tab fields, continued

Field Name	Value or Description
Connection Protocol	Connection protocol to use when connecting to this host.
Transfer Protocol	Transfer protocol to use when moving files.
Stream Encoding	Character encoding scheme used by the server.
Database section	
Enable Database	Select this check box if you need to configure fields in the Database section.
Server Type	Oracle Server or SQL Server . Fields in this section change according to the option selected, and are described here for the Oracle Server option.
Host Name	Host name of the server where the database runs. Used by AKLOAD to define a JDBC connection.
Connect String	Connect string for the database. Used by the OraApps Oracle Workflow object type.
Username	Database schema and user used for the APPS account. Passed to the AKLOAD and WFLOAD utilities.
Password	Password for the database user. Passed to the AKLOAD and WFLOAD utilities.
Oracle SID	SID for the database. Used by AKLOAD to define a JDBC connection. Used by the OraApps Oracle Workflow object type to set Oracle context.
Port Number	Port being monitored by the database listener. Used by AKLOAD to define a JDBC connection.
DB Link	Database link name.
Version	Oracle version number for this database. Used for reporting purposes.
JDBC_URL	JDBC URL for the database.

Configuring the Extension Data Tab, Oracle e Subtab

In the PPM Workbench, environments for Oracle E-Business Suite instances are specified in the Environment window's **Extension Data** tab, **Oracle Applications** subtab.

"Figure 3-2. Extension Data tab, Oracle Applications subtab sample data" shows the Extension Data tab, SAP subtab of a sample environment. "Table 3-2. Extension Data tab, Oracle Application subtab fields" lists the field names and their descriptions.

Figure 3-2. Extension Data tab, Oracle Applications subtab sample data

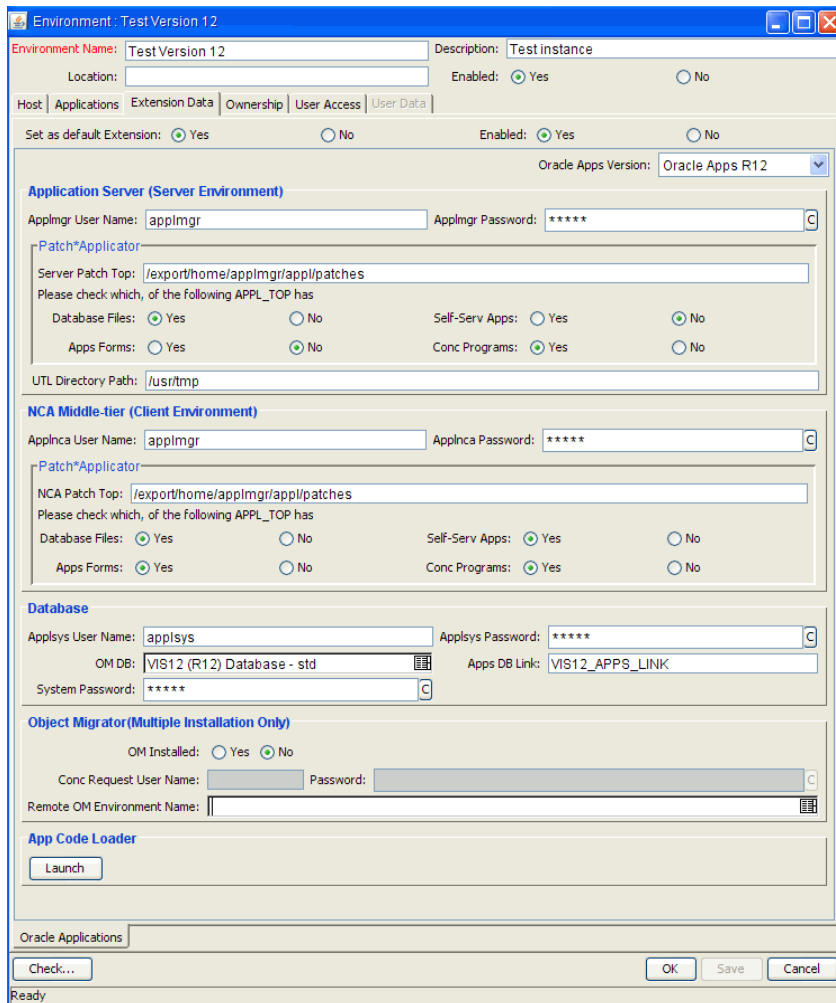


Table 3-2. Extension Data tab, Oracle Application subtab fields

Field Name	Value or Description
Set as default Extension	Option to show Oracle Applications (Oracle E-Business Suite) as the default subtab for this window's Extension Data tab.
Enabled	Select the Yes option to make the fields on this subtab available to be configured.
Oracle Apps Version	Version of Oracle E-Business Suite running in this environment—Oracle Apps R11 (for Oracle E-Business Suite instances at Release 11 or Release 11i), or Oracle Apps R12.

Table 3-2. Extension Data tab, Oracle Application subtab fields, continued

Field Name	Value or Description
Application Server (Server Environment) section	
Note: The fields in this section are used primarily by PPM Center during patch application and AD Administration tasks.	
Applmgr User Name	Oracle E-Business Suite file system user (usually APPLMGR) on the server host.
Applmgr Password	Password for the Oracle E-Business Suite file system user.
Server Patch Top	Top-level directory used to store patch files for use with Oracle Patch Applicator.
Database Files ^a	Whether or not APPL_TOP has database files installed on the server host. This value is used during execution of the ADPATCH and ADADMIN utilities on Release 11, and to determine whether patch detail information will be captured for a given Release 11, Release 11i, or Release 12 patch. Patch detail information is captured only for database tiers.
Self-Serv Apps ^a	Whether or not APPL_TOP has self-serve applications installed on the server host.
Apps Forms ^a	Whether or not APPL_TOP has forms installed on the server host.
Conc Programs ^a	Whether or not APPL_TOP has concurrent programs installed on the server host.
UTL Directory Path	Directory in which to place files for uploading or downloading Oracle workflows in an Oracle E-Business Suite instance. Used by the OraApps Oracle Workflow object type, this field must correspond to a UTL_FILE_DIR entry in the database's init.ora file.
NCA Middle-Tier (Client Environment) section	
Note: The fields in this section are used primarily by PPM Center during patch application and AD Administration tasks.	
Applnca User Name	Oracle E-Business Suite file system user (usually APPLNCA) on the middle tier host.
Applnca Password	Password for the Oracle E-Business Suite file system user on the middle tier host.

Table 3-2. Extension Data tab, Oracle Application subtab fields, continued

Field Name	Value or Description
NCA Patch Top	Top-level directory used to store NCA patch files for use with Patch Applicator.
Database Files ^a	Whether or not APPL_TOP has database files installed on the middle tier host.
Self-Serv Apps ^a	Whether or not APPL_TOP has self-serve apps installed on the middle tier host.
Apps Forms ^a	Whether or not APPL_TOP has forms installed on the middle tier host.
Conc Programs ^a	Whether or not APPL_TOP has concurrent programs installed on the middle tier host.
Database section	
Note: The fields in this section describe general database information for the Oracle E-Business Suite installation and are used by multiple functions including patching, AD Administration tasks, and Object Migrator.	
Applsyst Username	Oracle AOL database user (usually applsyst). Used for patching and AD Administration tasks.
Applsyst Password	Database password for the AOL user. Used for patching and AD Administration tasks.
OM DB	Object Migrator entry representing the database in this environment being specified. Retrieve the value from the CLM_DATABASES value set in the primary Object Migrator host. This field links a PPM Center environment definition to a source and destination defined for use with Object Migrator.
Apps DB Link	Name of the DB Link from the PPM Center schema to the APPS schema for the database in this environment being specified, if applicable. Used for Object Migrator migration and AKLOAD verifications.
System Password	Password for the SYSTEM user on this database. Used for patching and AD Administration tasks.
Object Migrator (Multiple Installation Only) section	
Note: The fields in this section are used only for remote Object Migrator functionality.	

Table 3-2. Extension Data tab, Oracle Application subtab fields, continued

Field Name	Value or Description
OM Installed	See Remote Execution of Object Migrator Concurrent Requests on page 62 if you are using this configuration.
Conc Request User Name	See Remote Execution of Object Migrator Concurrent Requests on page 62 if you are using this configuration.
Password	See Remote Execution of Object Migrator Concurrent Requests on page 62 if you are using this configuration.
Remote OM Environment Name	See Remote Execution of Object Migrator Concurrent Requests on page 62 if you are using this configuration.
App Code Loader section	
Launch button	Launches App Code Loader (for information about this utility, see App Code Loader).
a. This field is enabled only for Oracle E-Business Suite Release 11 environments. The settings are used during Release 11 patch applications and AD Administration tasks, and are only informational for Release 11i and Release 12. In Release 11i and Release 12, Oracle's patch and AD Administration programs derive this information automatically.	

App Code Loader

The App Code Loader utility can speed up the process of configuring environments. You can use this utility to generate new application codes or update the server and client base paths of existing application codes in an environment.

An Oracle E-Business Suite installation typically has many different applications, for example, OE and GL. Most Oracle E-Business Suite installations include configuration files that define the file system locations of each of these applications. These files usually have extensions of `.ora` or `.env`. The files contain entries that specify the top of the application's tree. For example, the following code segment specifies that the home directory for OE is located under `/u2/apps/oe/11.5.0`:

```
OE_TOP=/u2/apps/oe/11.5.0
GL_TOP=/u2/apps/gl/11.5.0
```

App Code Loader reads these files and imports this information into an environment. To launch App Code Loader, navigate to the Oracle Applications subtab of the Extension Data tab of an environment and click Launch. Detailed instructions describing how to use App Code Loader are available within the utility.

Note:

App Code Loader can be launched only when working in a saved environment. To enable the **Launch** button while specifying a new environment, click **Save**.

For a Windows® environment, refer to the environment file that is normally in the APPL_TOP directory, with a name such as SID_NAME.env. Based on this file, create a new file that has all the variables (Product Tops) you want to load using App Code

Loader, but without the string SET. Path specifications should include backslashes (\) to be consistent with Windows. An example variable is as follows:

```
FND_TOP=D:\OraApps\appl\TestApp1\fnd\11.5.0
```

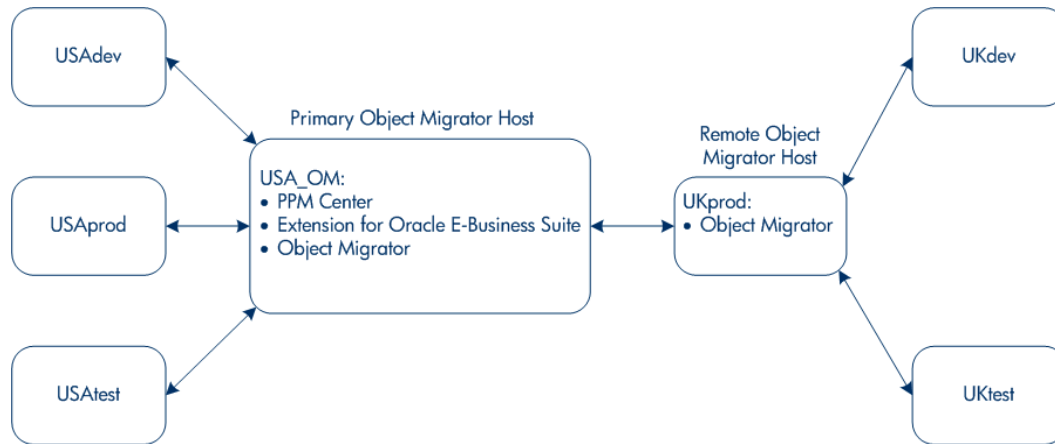
Remote Execution of Object Migrator Concurrent Requests

The Extension knows the instance of Oracle E-Business Suite on which Object Migrator requests run by default. This instance is called the primary Object Migrator host. Package line executions in Deployment Management are routed as concurrent requests to the primary Object Migrator host.

When network performance or geographical distance is an issue, or when organizations within a company are required to maintain their systems independently, it can be beneficial to install Object Migrator on certain remote Oracle E-Business Suite instances also. Such instances are called remote Object Migrator hosts. Then, instead of using the primary Object Migrator host to entirely manage the migration of objects among the remote Oracle E-Business Suite instances, you can use the Remote Execution feature of the Extension to make PPM Center submit concurrent requests to the remote Object Migrator host, which then manages the migration remotely and more efficiently.

"[Figure 3-3. Example setup for remote execution of Object Migrator](#)" shows an example setup of remote migration. (The example uses local integration. For more information, see "[Integration of the Extension with Oracle E-Business Suite](#)" on page 19.) Oracle E-Business Suite is installed on every instance shown. This example is explained in detail later.

Figure 3-3. Example setup for remote execution of Object Migrator



You must specify an environment in the Environment Workbench (see ["Figure 3-2. Extension Data tab, Oracle Applications subtab sample data" on page 45](#)) for the remote Object Migrator host and associated instances. If a particular environment (instance) is either a remote Object Migrator host or an instance for which a remote Object Migrator host is used to manage migrations, you must specify the fields in the **Object Migrator (Multiple Installation Only)** section of the Environment Workbench as follows:

- **OM Installed.** This field should be set to **Yes** only if this environment you are specifying is a remote Object Migrator host, that is, if it has Object Migrator installed and is intended to manage remote migrations.

This field should be set to **No** in the following cases:

- You are specifying an environment (instance) that does not have Object Migrator installed, even if the environment has its migrations managed by a remote Object Migrator host.
- The environment has Object Migrator installed but is not intended for use as a remote Object Migrator host.

In a typical remote configuration, Object Migrator is installed on only the remote Object Migrator host, and it manages migrations among several other remote Oracle E-Business Suite instances.

- **Conc Request User Name and Password.** These fields are used only when you are specifying the environment of a remote Object Migrator host. These fields are the operating system user name (usually APPLMGR) and password for a user who has permissions to FTP the Concurrent Request Log and output files to the PPM Server.

These fields should be left blank in the following cases:

- You are specifying an environment (instance) that does not have Object Migrator installed, even if the environment has its migrations managed by a remote Object Migrator host.
- The environment has Object Migrator installed but is not intended for use as a remote Object Migrator host.
- **Remote OM Environment Name.** This field is the name of the remote Object Migrator host that manages migrations for the environment you are specifying. Provide this

name when you are specifying the environment of the remote Object Migrator host. Also, provide this name when you are specifying the environments of other remote instances for which this remote Object Migrator host manages migrations.

Note:

In addition to specifying the environments, you must make sure that the `CLM_DATABASES` and `CLM_DB_LINKS` value sets in the primary Object Migrator host and all remote Object Migrator hosts are identical and have entries for all the databases associated with these Object Migrator hosts. We strongly recommend that you first specify master `CLM_DATABASES` and `CLM_DB_LINKS` value sets on the primary Object Migrator host and then use the Value Set Migrator to copy them to all the remote Object Migrator hosts.

The example configuration in "[Figure 3-3. Example setup for remote execution of Object Migrator](#)" on page 49 includes a primary Object Migrator host in the United States (USA_OM), a remote Object Migrator host in the UK (UKprod), other instances in the US for development, production, and test, and other instances in the UK for development and test.

Using USA_OM to migrate an entity from one machine in the UK (UKdev) to another (UKtest) would take more time than using the Object Migrator installed on the remote Object Migrator host, UKprod, because latency across international networks is generally much higher than latency within a relatively local network.

Therefore, when migrating entities among any of the databases in the UK, UKprod should be used instead of the default Object Migrator, USA_OM.

To efficiently set up the remote migration example shown in "[Figure 3-3. Example setup for remote execution of Object Migrator](#)":

1. Install the same version of Object Migrator that is installed in USA_OM on UKprod. Modify the `CLM_DATABASES` and `CLM_DB_LINKS` value sets in USA_OM to support all the instances (databases) in the USA and the UK. Then use the Value Set Migrator in Object Migrator to copy the value sets to UKprod.
2. Log on to PPM at USA_OM.
3. From the menu bar, select **Open > Administration > Open Workbench**.
The PPM Workbench opens.
4. From the shortcut bar, select **Environments > Environments**.
The Environment Workbench opens.
5. Open the UKprod environment and select the **Extension Data** tab, **Oracle Applications** subtab. (It is assumed that this environment does not yet have remote Object Migrator functionality configured, but has already been otherwise configured, including the **OM DB** and **Apps DB Link** fields.)
6. In the **Object Migrator (Multiple Installation Only)** section, set **OM Installed** to **Yes**.

7. Type values for the **Conc Request User Name** (usually APPLMGR) and **Password** fields. These fields are used to access the OUT and LOG files of the concurrent submission.
8. Click **Save** to save this environment.
9. In the **Remote OM Environment Name** field, select **UKprod**.
10. Click **OK** to save.
11. Open the environments of UKdev and UKtest, and perform the following steps for both environments. (It is assumed that these environments do not yet have remote Object Migrator functionality configured, but have already been otherwise configured, including the **OM DB** and **Apps DB Link** fields.)

You can open multiple environments simultaneously by holding down the Ctrl key while selecting items.

- a. Make sure that **OM Installed** is set to **No**.
- b. Set the **Remote OM Environment Name** field to **UKprod**.
- c. Click **OK** to save.

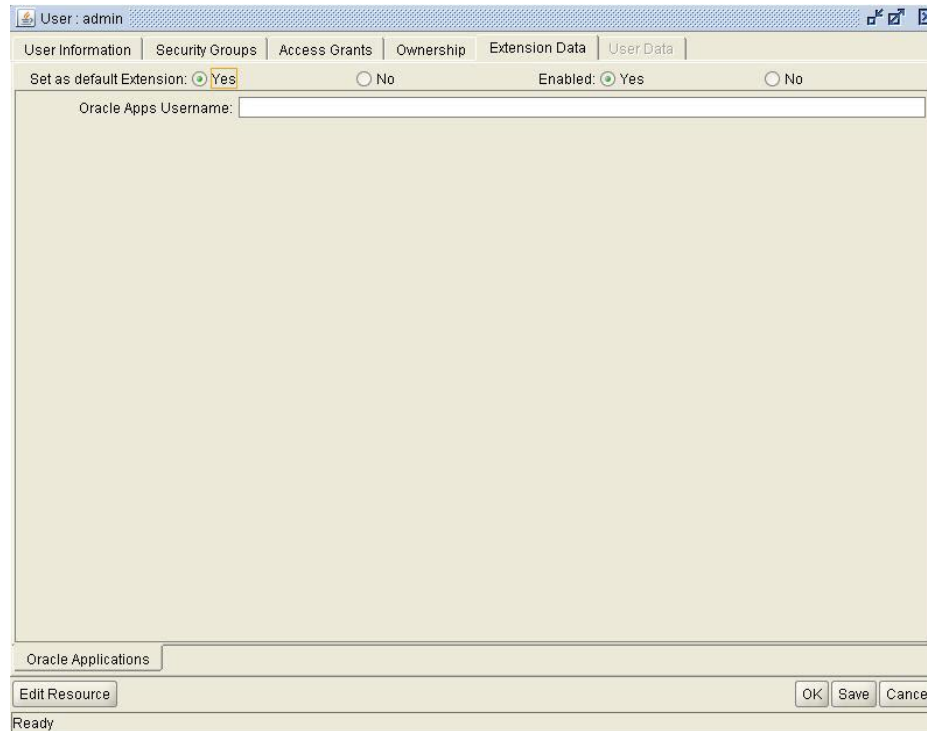
Now, whenever the destination is either UKdev or UKtest, UKprod will be used as the remote Object Migrator host.

In addition, you can use either the primary or remote Object Migrator host to manage migrations of objects from the USA to the UK or vice versa. For such migrations in either direction, the source and destination servers are so far apart that there is no performance advantage to using one Object Migrator host over the other.

Configuring Oracle E-Business Suite Users

The Extension allows an organization to include additional information specific to how a user accesses Oracle E-Business Suite. The **Oracle Apps Username** field is used when PPM submits concurrent requests, and it is reflected in Concurrent Request Logs. Figure 3-4 shows a user's **Extension Data** tab, **Oracle Applications** subtab in the PPM Workbench.

Figure 3-4. User window, Oracle Applications subtab



1. Log on to PPM.
2. From the menu bar, select **Open > Administration > Open Workbench**.
The PPM Workbench opens.
3. From the shortcut bar, select **Sys Admin > Users**.
The User Workbench opens.
4. Open a new or existing user.
5. Click the **Extension Data** tab for the user.
6. Click the **Oracle Applications** subtab at the bottom of the user window.

For more information about defining PPM Center users and privileges, see the *Installation and Administration Guide*.

Configuring Security Groups

The Extension includes sample security groups representing common roles in organizations that use Oracle E-Business Suite. The workflows included in the Extension use these predefined Oracle-related security groups by default. You must configure these security groups (and add any others your organization requires) for access, membership, and limitations, so they can act on workflow steps.

The predefined security groups for the Extension are as follows:

- OA - Business Owner
- OA - Developer
- OA - Functional Designers

- OA - Functional Gap Reviewer
- OA - Management
- OA - Oracle Patch Admin
- OA - Sponsor
- OA - Technical Gap Reviewer
- OA - Track Owner

For more information about security groups, see the *Security Model Guide and Reference*.

Configuring Request Types

The Extension includes request types containing sample values for field validations. You must configure these validations to match the specifics of your organization. You should also assign a default workflow to each request type.

The request types for the Extension, described in detail in **Appendix B, Request Types**, on page 263, are as follows:

- OraApps Application Issue
- OraApps Cloning Request
- OraApps Conversion Request
- OraApps Design & Development
- OraApps Enhancement Request
- OraApps GAP Analysis Request
- OraApps Interface Request
- OraApps Report Request
- OraApps Setup Change Request
- OraApps Status Update Request

To configure request types:

1. Log on to PPM Center.
2. From the menu bar, select **Open > Administration > Open Workbench**.
The PPM Workbench opens.
3. From the shortcut bar, select **Demand Mgmt > Request Types**.
The Request Type Workbench opens.
4. Open a new request type or, if the request type of interest already exists, open it.
5. In the **Workflows** tab of the request type in the Request Type Workbench, select the workflow that can be used with the request type.
Typically, only one workflow is attached. "[Table 3-3. Deployment Management request type/workflow pairs](#)"³ lists associated request types and workflows.
6. In the **Rules** tab of the request type, select the default workflow.

Table 3-3 provides the information required to make this selection.

7. (Optional) In the **Ownership** tab of the request type, select the security groups allowed to modify the request type.
8. (Optional) In the **User Access** tab of the request type, select the security groups allowed to create requests of this request type. For example, you might want to do this if only some of the roles are allowed to request setup changes.
9. (Optional) To restrict visibility or editability of one or more fields, define field-level security.
 For example, you might want to limit edit access on analysis fields to developers and track owners.
10. Enable the request types you want to use that have not been enabled. Select Yes in the Enabled field of the request type.

Table 3-3. Deployment Management request type/workflow pairs

Request Type	Workflow
OraApps Conversion Request	OraApps Conversion Process
OraApps Design & Development	OraApps Design & Development
OraApps Enhancement	OraApps Enhancement Process
OraApps Interface	OraApps Interfaces Process
OraApps Report	OraApps Reports Process
OraApps Setup Change Request	OraApps Setup Change Process

Note:

In addition to the workflows listed in Table 3-3, the OraApps Customization/ Configuration Deployment workflow can be used at the Create Package and Wait steps of any of the request types to deploy the changes targeted to different instances.

For more information about all of the request types, including those in "[Table 3-3. Deployment Management request type/workflow pairs](#)", see Appendix B, Request Types, on page 263.

Configuring Workflows

The Extension includes workflows that model proven real-world practices. Some workflows contain sample security information. You can configure the workflows to represent business processes for your specific environment. You must also verify or modify security and notification information.

The workflows for the Extension, described in detail in Appendix C, Workflows, on page 303, are as follows:

- OraApps 11i Cloning
- OraApps Application Issue
- OraApps Cloning Process
- OraApps Conversion Process
- OraApps Customization/Configuration Deployment
- OraApps Design & Development
- OraApps Enhancement Process
- OraApps GAP Analysis Process
- OraApps Interfaces Process
- OraApps Patch Deployment workflow, and its subworkflows as follows:
 - OraApps Patch Deployment Subworkflow
 - OraApps Patch Data Capture Subworkflow
- OraApps Reports Process
- OraApps Setup Change Process
- OraApps Status Update Request

Figure 3-5 shows a sample workflow step configuration window.

The screenshot shows the 'Workflow Step' configuration window. The 'Properties' tab is selected. The 'Step Number' is 6 and the 'Step Name' is 'Deploy from DEV to TEST'. The 'Source Type' is 'Execution' and the 'Source Name' is 'OA - Manual Execution'. The 'Enabled' checkbox is checked. The 'Display' dropdown is set to 'Always'. The 'Workflow Parameter' is 'NONE'. The 'Source Environment' is 'DPM_DEV' and the 'Dest Environment' is 'DPM_TEST'. The 'Save to OM/GLM Archive?' checkbox is unchecked. The 'Authentication Required' dropdown is set to 'None'. The window has 'OK', 'Apply', and 'Cancel' buttons at the bottom right.

To configure an Oracle-specific workflow:

1. Log on to PPM.
2. From the menu bar, select **Open > Administration > Open Workbench**.
The PPM Workbench opens.
3. From the shortcut bar, select **Configuration > Workflows**.
The Workflow Workbench opens.
4. Open a new workflow or, if the workflow of interest already exists, open it.
5. In the **Layout** tab of the Workflow window, right-click an execution step and click **Edit**.
6. Assign security groups to the workflow steps to make sure that only the appropriate set of users can act on a given step.

For example, you would probably want to limit which users are permitted to act on the Sign-Off Functional Design step to only those users authorized to approve functional designs.

The Extension includes predefined security groups on some workflow steps.

7. To prevent unexpected migrations, limit the object types that can be used with a particular workflow.

The Deployment Management object types can be used with any workflow that specifies valid source and destination environments. The OraApps

Customization/Configuration Deployment workflow is specifically designed to handle these entities integrated with other activities.

8. Modify the workflow to reflect the Oracle E-Business Suite instances in use.
Set source and destination environments on the workflow execution steps. The **OM DB** fields from source and destination environments are used as the source and destination fields to Object Migrator or GL Migrator.
9. Use the Object Archive when using Deployment Management to run Object Migrator or GL Migrator. You can do this in one of the following ways:
 - Create an environment (with a name such as Archive). In the **Extension Data** tab, **Oracle Applications** subtab, set the **OM DB** field to **Object Archive**.
 - Set the **Save to OM/GLM Archive** field to **Yes** when defining your workflow migration step. Then Object Migrator and GL Migrator will both migrate to your destination environment and save to the Object Archive.

In order to save to the Object Archive, you must provide a value for the **Version Label** field on the package line Object Migrator or GL Migrator entry. Because the same entity cannot be archived twice in the same version, no more than one step on your workflow should have the **Save to OM/GLM Archive** field set to **Yes**.

10. Verify that the target workflow has been enabled.

For more information about workflows, see **Appendix C, Workflows**, on page 303.

Configuring Object Types

The Extension includes many predefined object types that might need to be configured to reflect special logic required for your organization's migrations, such as preprocessing work, integration with source control, and setup of the execution environment.

For information about configuring object types, see **Overview of Object Types** on page 215.

Extension-specific object types are categorized as follows and are described in detail in **Appendix A, Object Types**, on page 215:

- Application data migration object types, which are discussed in the following sections under the following subcategories:
 - AOL and GL object types
 - AOL:Single <Entity> object types
 - OraApps Oracle Workflow object type
 - OraApps AKLOAD Migrate AK Setups object type
 - FNDLOAD automation object types
- Application patching and administration object types, which are discussed in **Patch and Administration Object Types** on page 237
- An instance management object type (the OraApps 11i Cloning object type), which is discussed in **Instance Management Object Type** on page 255

AOL and GL Object Types

To configure AOL or GL object types:

1. Enable the object type.
2. To modify the order of Object Migrator submissions to Oracle, adjust the Object Migrator or GL Migrator sequence value.

For more information about these object types, see [AOL and GL Object Types](#) on page 221.

AOL:Single <Entity> Object Types

AOL:Single <Entity> object types include the following:

- AOL:Single Conc Mgr Entry
- AOL:Single GUI Menu Entry
- AOL:Single Report Group Unit

AOL:Single <Entity> object types allow users to do the following:

- Add a single specialization rule to an existing concurrent manager definition
- Add a single component to an existing GUI menu or report group

You must enable AOL:Single <Entity> object types before you can use them.

For more information about these object types, see the following sections:

- [AOL:Single Conc Mgr Entry Object Type](#) on page 224
- [AOL:Single GUI Menu Entry Object Type](#) on page 226
- [AOL:Single Report Group Unit Object Type](#) on page 228

OraApps Oracle Workflow Object Type

The OraApps Oracle Workflow object type allows users to migrate embedded Oracle workflow definitions between Oracle E-Business Suite Release 11i and Release 12 instances. By using Micro Focus products to migrate these definitions, users can increase the security, control, and visibility of these changes, and also integrate with a source code control system.

To configure the OraApps Oracle Workflow object type:

1. Before downloading or uploading Web setup data, set the Oracle instance context.
The object type assumes this context is automatically set at login. If not, amend the object type to set the context, for example, by calling the appropriate `APPSORA.env` file.
2. Enable the object type.

For more information about this object type, see OraApps Oracle Workflow Object Type on page 234.

OraApps AKLOAD Migrate AK Setups Object Type

The OraApps AKLOAD Migrate AK Setups object type allows users to automate the migration of custom Web page configuration data using the Oracle AKLOAD utility, saving effort, increasing auditability, and reducing data entry errors. The following types of data can be migrated:

- Attributes
- Objects
- Regions
- Flows

Caution:

The Oracle AKLOAD utility supports only Oracle Release 11i.

Be aware of the following configuration considerations:

- Only custom data should be migrated with this object type, because the object type does not validate or prevent migration of standard or Micro Focus-supplied data.
- This object type is configured by default to connect using JDBC, but you can configure it to use other protocols supported by AKLOAD.
- Only data from Oracle E-Business Suite instances that are at the same point release and patch level should be migrated. This object type does not check the Oracle E-Business Suite version.
- You can put the extract file under a specific AppCode's base path instead of the server base path by specifying the AppCode's base path in the package line definition.
- You must apply Oracle patch #2594863 to your Oracle E-Business Suite instances, or the instance should be at Release 11.5.9 or later, for the following fields to get populated by the AKLOAD utility:
 - `display_height`, `css_class_name`, `poplist_viewobject`
 - `poplist_display_attribute`, `poplist_value_attribute`
 - `css_label_class_name`
- Any database views upon which your Web setup depends should already exist in the destination instance. The existence of these views is not validated by the object type.
- Migration of security data is not supported. Security information can be migrated using the AOL:Resp migrator.
- Migration of personalization values is not supported.

To configure the OraApps AKLOAD Migrate AK Setups object type:

1. Enable the object type.
2. The Oracle E-Business Suite instance context must be set before downloading or uploading Web setup data. The object type should be modified to do so if the existing logic is not adequate.
3. This object type uses a destination value of `$NLS_LANG` during import. If you need a different value, change the object type logic.

For more information about this object type, see OraApps AKLOAD Migrate AK Setups Object Type on page 230.

FNDLOAD Automation Object Types

The following object types automate the use of Oracle's FNDLOAD utility for migrations:

- The AR:Phone Country Codes object type is a setup entity that is used to migrate the standard country codes used for the phone number formats of various countries in Oracle's Financial Module Accounts Receivable (AR).
- The INV:Units of Measure object type is a setup entity that is used to migrate the Units of Measure (such as gallons, feet, and minutes) used for various inventory items in Oracle's Manufacturing Module Inventory (INV).

Caution:

If you use either object type as a template to build custom objects for other entities that you intend to use with FNDLOAD, you must fully understand the functionality of those entities and create custom FNDLOAD control (.lct) files for each one. You must ensure that the control file includes all the business logic and validations needed to maintain the referential integrity for Oracle entity IDs across the instances during a migration. Using a control file that does not include the necessary business logic and validations can lead to data corruption at the destination.

Managing Integration with Oracle E-Business Suite

The following sections provide information about changing the integration method to local or linked, and managing Oracle E-Business Suite downtime in linked integrations. For an introduction to integration, see ["Introduction to the Extension for Oracle E-Business Suite" on page 15](#).

Caution:

Never change the integration method while the PPM Server is running. Doing so can lead to inconsistent and unexpected results or errors.

All existing concurrent requests must be completed and processed within PPM Center before changing the integration method. Otherwise, request data might be lost.

Changing the Integration Method to Local

To change the integration method to local:

1. Be sure that PPM is sharing a database with the Oracle E-Business Suite instance that you want to be the primary Object Migrator host.
2. Be sure that all package line executions related to Object Migrator and GL Migrator in PPM Center have completed.
3. Stop the PPM Server.

Caution:

Never change the integration method while the PPM Server is running. Doing so can lead to inconsistent and unexpected results or errors.

4. While logged into the database as the database user who owns PPM, run the `fnd_drop_dummy.sql` script.

This script drops any existing local tables and database packages that need to be refreshed.

You must run this script from

```
<PPM_Home>/deploy/940/OracleApps/phases/db/fnd/
```

where `<PPM_Home>` represents the path where your PPM Center instance is installed. For example: `xyzserver/E/PPMServer`. For example, run the following:

```
spool fnd_drop_dummy.log  
@@fnd_drop_dummy.sql
```

5. Review the log for errors. Correct any problems found and run the script again.
6. From the `<PPM_Home>/deploy/940/OracleApps/phases/db/fnd/` directory, and while logged into the database as the database user who owns PPM Center, run the `fnd_knta_synonyms.sql` script.

This script does the following:

- Uses `<oraapps_schema>`, which represents the name of the Oracle E-Business Suite APPS schema in the database
- Defines synonyms in the PPM Center account for certain objects in the APPS account

For example, run the following:

```
spool fnd_knta_synonyms.log @  
@fnd_knta_synonyms.sql <oraapps_schema>
```

7. Review the log for errors. Correct any problems found and run the script again.
8. From the `<PPM_Home>/deploy/940/OracleApps/phases/db/fnd` directory, and while logged into the database as the database user who owns the Oracle E-Business Suite APPS schema (usually APPS), run the `fnd_apps_grants.sql` script.

This script does the following:

- Uses `<ppm_schema>`, `<oraapps_schema>`, and `<oraapps_password>` where
`<ppm_schema>` represents the name of the schema where PPM Center is installed.
`<oraapps_schema>` represents the Oracle E-Business Suite APPS schema in the database.
`<oraapps_password>` represents the password for the APPS schema.
- Installs the KINTANA_SUBMIT_REQUEST Package
- Grants access to certain objects to the PPM Center schema
- Should run as the APPS database user For example, run the following:

```
spool fnd.apps.grants.log  
@@fnd_apps_grants.sql <ppm_schema> <oraapps_schema> <oraapps_password>
```

9. If any values have changed, such as the user or password that is used to retrieve log files from the primary Object Migrator host, change the PPM Server configuration to reflect the new values.

10. Start the PPM Server.

Changing the Integration Method to Linked

To change the integration method to linked:

1. Be sure that all package line executions related to Object Migrator and GL Migrator in PPM Center have completed.
2. Stop the PPM Server.

Caution:

Never change the integration method while the PPM Server is running. Doing so can lead to inconsistent and unexpected results or errors.

3. While logged into the database as the database user who owns PPM, run the `fnd_drop_dummy.sql` script.

This script drops any existing local tables and database packages that need to be refreshed.

You must run this script from

```
<PPM_Home>/deploy/940/OracleApps/phases/db/fnd/
```

where `<PPM_Home>` represents the path where your PPM instance is installed. For example: `xyzserver/E/PPMServer`. For example, run the following:

```
spool fnd_drop_dummy.log  
@@fnd_drop_dummy.sql
```

4. Review the log for errors. Correct any problems found and run the script again.

5. Make sure that a database link to the APPS account of the Oracle E-Business Suite instance is defined in the PPM Center schema.
6. Test the link as shown in the following example:

```
select count(*) from fnd_application@<db_link_name>;
```

where <db_link_name> represents the name of the database link (for example, PROD_APPES_LINK) defined in the PPM Center schema that connects to the APPS account of the primary Object Migrator host.

7. From the <PPM_Home>/deploy/940/OracleApps/phases/db/fnd/ directory, and while logged into the database as the database user who owns PPM Center, run the `fnd_linked_synonyms.sql` script.

For example, run the following:

```
spool linked_synonyms.log @  
@fnd_linked_synonyms.sql <db_link_name>
```

8. Review the log for errors. Correct any problems found and run the script again.
9. If any values have changed, such as the user or password that is used to retrieve log files from the primary Object Migrator host, change the PPM Server configuration to reflect the new values.
10. Start the PPM Server.

Managing Linked Integration During Oracle E-Business Suite Downtime

When the primary Object Migrator host uses linked integration with PPM Center, if the Oracle E-Business Suite database for Object Migrator becomes unavailable, there might be some loss of functionality in PPM Center such that you cannot do any of the following:

- Submit Object Migrator or GL Migrator concurrent requests to the primary Object Migrator host
- Track the status of concurrent requests submitted to the primary Object Migrator host
- Create new package lines for AOL objects (because values will be unavailable for certain fields)
- Add or modify the OM DB value in the Environment Workbench window, **Extension Data** tab, **Oracle Applications** subtab, **Database** section for any migrations using AOL and GL object types
- Validate the **OM DB** value using the Environment Checker

The following sections describe setting parameters in the PPM Center configuration file (`server.conf`) to reduce errors reported by the Extension's Concurrent Request WatchDog function and to improve PPM Center performance when the Oracle E-Business Suite database is unavailable.

Configuring a server.conf Parameter for Fewer WatchDog Errors

The Concurrent Request WatchDog is a part of the Extension that does the following:

- Tracks the status of concurrent requests submitted to Oracle E-Business Suite
- Updates the related package lines with the outcome
- Links the log and output files of a concurrent request to the package line execution log where Object Migrator is running

These activities cannot take place when the Oracle E-Business Suite database is unavailable. If the Concurrent Request WatchDog cannot locate a submitted concurrent request after three attempts, it stops watching the request and terminates the package line execution with a command execution error.

Normally, this is not a problem. However, if linked database access goes down while some concurrent requests are still pending, the requests can produce a PPM Center error even though they will eventually run in Oracle E-Business Suite. To help manage this situation, you can add the following parameter in the PPM Center configuration file (server.conf) to specify a value for the number of times you want PPM Center to check for the request before forcing an error status:

```
com.kintana.core.server.ORACLE_APPS_MAX_CHECKS=<value>
```

For example, to set this parameter to 60 (that is, to check 60 times before forcing an error status), add the following:

```
com.kintana.core.server.ORACLE_APPS_MAX_CHECKS=60
```

If the Oracle E-Business Suite database is scheduled to be down for an extended period while PPM Center remains available, you must make sure that all concurrent requests are completed before the Oracle E-Business Suite database is taken down.

Configuring a server.conf Parameter for Better PPM Center Performance

When the PPM Server starts up, it detects the release of Oracle E-Business Suite being used in the primary Object Migrator host instance. If the database link to the Oracle E-Business Suite instance is down at the time the PPM Server starts up (for example, if the Oracle E-Business Suite database is down for backup or patching), PPM Center cannot determine the Oracle E-Business Suite release.

In this case, PPM Center determines the release as required during processing. In high-volume situations, this can lead to performance degradation in PPM Center, because redundant queries are issued. If you restart the PPM Server, it attempts to detect the Oracle E-Business Suite release again.

If this situation is likely to occur frequently, and restarting the PPM Server is not a viable option, you can provide default release (version) information for PPM Center to use during processing.

For any Oracle release supported by the Extension, add the following parameter to your server configuration file (server.conf):

```
com.kintana.core.server.ORACLE_APPS_VERSION=R11
```

Caution:

You must specify the value `R11`. The value `R11` applies to Release 11, Release 11i, and Release 12.

If you change the primary Object Migrator host, and if it was manually defined, you must update this parameter.

Extension Interface

This section includes the following topics:

- ["Menus" below](#)
- ["Preconfigured Dashboard Page and Portlets" on page 77](#)

Menus

The menu bar in the standard interface has the following menu paths that are specifically associated with the Extension:

- **Open > Oracle Applications**, which is organized by Extension functionality, allowing quick access to tasks in functional areas such as deployment management
- **Open > Oracle Applications Roles**, which is organized by role, providing direct access to tasks associated with specific roles

Both menu paths provide access to key Extension functionality. Each is structured to support the following predefined business roles:

- Project manager
- Project sponsor
- Track owner
- DBA

Items relating to a given business role are defined to be accessed by that role. For more information about these business roles, see **Business Concepts Used in the Oracle Environment on page 27**.

The Extension menus are seamlessly integrated with other menus in the menu bar. The Oracle-related menus include security definitions that restrict display of menu items to only those users authorized to view them.

Note:

The menus described here represent the default configuration installed with the product. Your customized configuration might differ.

Oracle Applications Menu

The **Open > Oracle Applications** menu path has a submenu for each major area of Extension functionality. Users view only those menu items for which they have licenses and permissions. The submenu options are as follows:

- Issue Management
- Project Management
- Deployment Management

- Instance Management
- Patch Management

Depending on the submenu, users can do the following:

- Create requests
- Run reports
- View or maintain key business processes
- Access the PPM Workbench for additional tasks

Issue Management Submenu

"[Table 4-1. Issue Management submenu in Oracle Applications menu](#)" describes menu items and submenus of the **Open > Oracle Applications > Issue Management** submenu.

Table 4-1. Issue Management submenu in Oracle Applications menu

Menu Item or Submenu	Description
Create Apps Issue Request	Creates a new OraApps Application Issue request
Reports submenu	
OraApps Apps Issues Detail	Runs the OraApps Apps Issues Detail Report
OraApps Issues Summary	Runs the OraApps Apps Issues Summary Report
Administration submenu	
Manage Apps Issue Process	Opens the OraApps Application Issue workflow in the PPM Workbench

Project Management Submenu

"[Table 4-2. Project Management submenu in Oracle Applications menu](#)" describes menu items and submenus of the **Open > Oracle Applications > Project Management** submenu.

Table 4-2. Project Management submenu in Oracle Applications menu

Menu Item or Submenu	Description
Create GAP Analysis request	Creates a new OraApps Application Issue request

Table 4-2. Project Management submenu in Oracle Applications menu, continued

Menu Item or Submenu	Description
Create Status Update request	Creates a new OraApps Status Update request
Reports submenu	
Oracle Apps IT Demand Summary	Runs the OraApps IT Demand Summary Report
Oracle Apps Critical Request Summary	Runs the OraApps Critical Open Request Summary Report
Administration submenu	
Manage GAP Analysis Process	Opens the OraApps GAP Analysis Process workflow in the PPM Workbench
Manage Status Update Process	Opens the OraApps Status Update Request workflow in the PPM Workbench

Deployment Management Submenu

"[Table 4-3. Deployment Management submenu in Oracle Applications menu](#)" describes menu items and submenus of the **Open > Oracle Applications > Deployment Management** submenu.

Table 4-3. Deployment Management submenu in Oracle Applications menu

Menu Item or Submenu	Description
Create Interface request	Creates a new OraApps Application Issue request
Create Conversion request	Creates a new OraApps Conversion request
Create Setup Change request	Creates a new OraApps Setup Change request
Create Report request	Creates a new OraApps Report request
Create Enhancement request	Creates a new OraApps Enhancement request
Open Package Workbench	Opens the Package Workbench
Administrations submenu	
Manage Interfaces Process	Opens the OraApps Interfaces Process workflow in the PPM Workbench

Table 4-3. Deployment Management submenu in Oracle Applications menu, continued

Menu Item or Submenu	Description
Manage Conversion Process	Opens the OraApps Conversion Process workflow in the PPM Workbench
Manage Setup Change Process	Opens the OraApps Setup Change Process workflow in the PPM Workbench
Manage Reports Process	Opens the OraApps Reports Process workflow in the PPM Workbench
Manage Enhancements Process	Opens the OraApps Enhancement Process workflow in the PPM Workbench
Manage Deployment Process	Opens the OraApps Customization/ Configuration Deployment workflow in the PPM Workbench

Instance Management Submenu

"[Table 4-4. Instance Management submenu in Oracle Applications menu](#)" describes menu items and submenus of the **Open > Oracle Applications > Instance Management** submenu.

Table 4-4. Instance Management submenu in Oracle Applications menu

Menu Item or Submenu	Description
Create Cloning request	Creates a new OraApps Cloning request
Open Package Workbench	Opens the Package Workbench
Reports submenu	
Compare Custom Database Setup	Runs the Compare Custom Database Setup Report
Compare Oracle Environments	Runs the Compare Oracle Environments Report
Administration submenu	
Manage Env. Cloning Request Process	Opens the OraApps Cloning Process workflow in the PPM Workbench
Manage Env. Cloning Process	Opens the OraApps 11i Cloning workflow in the PPM Workbench

Patch Management Submenu

Table 4-5 describes menu items and submenus of the **Open > Oracle Applications > Patch Management** submenu.

Table 4-5. Patch Management submenu in Oracle Applications menu

Menu Item or Submenu	Description
Open Package Workbench	Opens the Package Workbench
Reports submenu	
Compare Custom Database Setup	Runs the Compare Custom Database Setup Report
Compare Oracle Environments	Runs the Compare Oracle Environments Report
Patch Application Comparison	Runs the Patch Application Comparison Report
Patches Applied to an Env	Runs the Patches Applied to an Environment Report
Pending Patches	Runs the Pending Patches Report
OraApps Patch Detail	Runs the OraApps Patch Detail Report
OraApps Patch Analysis	Runs the OraApps Patch Analysis Report
Administration submenu	
Manage Patch Deployment Process	Opens the OraApps Patch Deployment workflow in the PPM Workbench

Oracle Applications Roles Menu

The **Open > Oracle Applications Roles** menu path has the following submenu options correspond to business roles:

- Project Sponsor
- Project Manager
- Track Owner
- Apps DBA

The menu items under the submenus include common tasks for each role, for example, creating requests or running reports. Users view only those menu items for which they have licenses and permissions.

Project Sponsor Submenu

Table 4-6 describes menu items and submenus of the **Open > Oracle Applications Roles > Project Sponsor** submenu.

Table 4-6. Project Sponsor submenu in Oracle Applications Roles menu

Menu Item or Submenu	Description
Create GAP Analysis request	Creates a new OraApps GAP Analysis request
Create Interface request	Creates a new OraApps Interface request
Create Conversion request	Creates a new OraApps Conversion request
Create Setup Change request	Creates a new OraApps Setup Change request
Create Report request	Creates a new OraApps Report request
Create Enhancement request	Creates a new OraApps Enhancement request
Create Status Update request	Creates a new OraApps Status Update request
Create Apps Issue Request	Creates a new OraApps Application Issue request
Reports submenu	
Oracle Apps IT Demand Summary	Runs the OraApps IT Demand Summary Report
Oracle Apps Critical Request Summary	Runs the OraApps Critical Open Request Summary Report
OraApps Apps Issues Detail	Runs the OraApps Apps Issues Detail Report
Administration submenu	
Manage GAP Analysis Process	Opens the OraApps GAP Analysis Process workflow in the PPM Workbench
Manage Status Update Process	Opens the OraApps Status Update Request workflow in the PPM Workbench

Project Manager Submenu

"[Table 4-7. Project Manager submenu in Oracle Applications Roles menu](#)" describes menu items and submenus of the **Open > Oracle Applications > Project Management** submenu.

Table 4-7. Project Manager submenu in Oracle Applications Roles menu

Menu Item or Submenu	Description
Create GAP Analysis request	Creates a new OraApps GAP Analysis request
Create Interface request	Creates a new OraApps Interface request
Create Conversion request	Creates a new OraApps Conversion request
Create Setup Change request	Creates a new OraApps Setup Change request
Create Report request	Runs the OraApps Critical Open Request Summary Report
Create Enhancement request	Creates a new OraApps Enhancement request.
Create Status Update request	Creates a new OraApps Status Update request
Create Apps Issue Request	Creates a new OraApps Application Issue request
Create Cloning request	Creates a new OraApps Cloning request
Reports submenu	
OraApps IT Demand Summary	Runs the OraApps IT Demand Summary Report
OraApps Critical Request Summary	Runs the OraApps Critical Open Request Summary Report
OraApps Apps Issues Detail	Runs the OraApps Apps Issues Detail Report
OraApps Issues Summary	Runs the OraApps Apps Issues Summary Report

Table 4-7. Project Manager submenu in Oracle Applications Roles menu, continued

Menu Item or Submenu	Description
Administration submenu	
Manage GAP Analysis Process	Opens the OraApps GAP Analysis Process workflow in the PPM Workbench
Manage Interfaces Process	Opens the OraApps Interfaces Process workflow in the PPM Workbench
Manage Conversion Process	Opens the OraApps Conversion Process workflow in the PPM Workbench
Manage Setup Change Process	Opens the OraApps Setup Change Process workflow in the PPM Workbench
Manage Reports Process	Opens the OraApps Reports Process workflow in the PPM Workbench
Manage Enhancements Process	Opens the OraApps Enhancement Process workflow in the PPM Workbench
Manage Status Update Process	Opens the OraApps Status Update Request workflow in the PPM Workbench
Manage Apps Issue Process	Opens the OraApps Application Issue workflow in the PPM Workbench
Manage Env. Cloning Request Process	Opens the OraApps Cloning Process workflow in the PPM Workbench
Manage Deployment Process	Opens the OraApps Customization/ Configuration Deployment workflow in the PPM Workbench

Track Owner Submenu

"[Table 4-8. Track Owner submenu in Oracle Applications Roles menu](#)" describes menu items and submenus of the **Open > Oracle Applications Roles > Track Owner** submenu.

Table 4-8. Track Owner submenu in Oracle Applications Roles menu

Menu Item or Submenu	Description
Create GAP Analysis request	Creates a new OraApps GAP Analysis request

Table 4-8. Track Owner submenu in Oracle Applications Roles menu, continued

Menu Item or Submenu	Description
Create Interface request	Creates a new OraApps Interface request
Create Conversion request	Creates a new OraApps Conversion request
Create Setup Change request	Creates a new OraApps Setup Change request
Create Report request	Runs the OraApps Critical Open Request Summary Report
Create Enhancement request	Creates a new OraApps Enhancement request.
Create Status Update request	Creates a new OraApps Status Update request
Create Apps Issue Request	Creates a new OraApps Application Issue request
Create Cloning request	Creates a new OraApps Cloning request
Open Package Workbench	Opens the Package Workbench
Reports submenu	
OraApps IT Demand Summary	Runs the OraApps IT Demand Summary Report
OraApps Critical Request Summary	Runs the OraApps Critical Open Request Summary Report
OraApps Apps Issues Detail	Runs the OraApps Apps Issues Detail Report
OraApps Issues Summary	Runs the OraApps Apps Issues Summary Report
Administration submenu	
Manage GAP Analysis Process	Opens the OraApps GAP Analysis Process workflow in the PPM Workbench

Table 4-8. Track Owner submenu in Oracle Applications Roles menu, continued

Menu Item or Submenu	Description
Manage Interfaces Process	Opens the OraApps Interfaces Process workflow in the PPM Workbench
Manage Conversion Process	Opens the OraApps Conversion Process workflow in the PPM Workbench
Manage Setup Change Process	Opens the OraApps Setup Change Process workflow in the PPM Workbench
Manage Reports Process	Opens the OraApps Reports Process workflow in the PPM Workbench
Manage Enhancements Process	Opens the OraApps Enhancement Process workflow in the PPM Workbench
Manage Status Update Process	Opens the OraApps Status Update Request workflow in the PPM Workbench
Manage Env. Cloning Request Process	Opens the OraApps Cloning Process workflow in the PPM Workbench
Manage Deployment Process	Opens the OraApps Customization/ Configuration Deployment workflow in the PPM Workbench

Apps DBA Submenu

"[Table 4-9. Apps DBA submenu in Oracle Applications Roles menu](#)" describes menu items and submenus of the **Open > Oracle Applications Roles > Track Owner** submenu.

Table 4-9. Apps DBA submenu in Oracle Applications Roles menu

Menu Item or Submenu	Description
Create Apps Issue Request	Creates a new OraApps Application Issue request
Create Cloning request	Creates a new OraApps Cloning request
Open Package Workbench	Opens the Package Workbench
Reports submenu	

Table 4-9. Apps DBA submenu in Oracle Applications Roles menu, continued

Menu Item or Submenu	Description
Compare Custom Database Setup	Runs the Compare Custom Database Setup Report
Compare Oracle Environments	Runs the Compare Oracle Environments Report
Patch Application Comparison	Runs the Patch Application Comparison Report
Patches Applied to an Env	Runs the Patches Applied to an Environment Report
Pending Patches	Runs the Pending Patches Report
OraApps Apps Issues Detail	Runs the OraApps Apps Issues Detail Report
OraApps Issues Summary	Runs the OraApps Apps Issues Summary Report
Administration submenu	
Manage Env. Cloning Request Process	Opens the OraApps Cloning Process workflow in the PPM Workbench
Manage Deployment Process	Opens the OraApps Customization/ Configuration Deployment workflow in the PPM Workbench

Preconfigured Dashboard Page and Portlets

You can add preconfigured Dashboard pages to your PPM Dashboard. Preconfigured pages include a set of portlets related to a function. These pages are generally configured for common usage but you can personalize them for your specific business needs. You can add only those pages for which you have been granted access by the PPM administrator.

The pages and portlets shown in this document are samples. Their content and configuration can vary depending on how the Extension is used by your business.

Many of the portlets have drill-down capability. For example, on the Oracle Apps Project Manager page, in the OraApps: Critical Activities portlet shown in "[Figure 4-1. OraApps: Critical Activities portlet](#)", users can click the number 4 in the # of Reqs column to view detailed information about the four OraApps Application Issue requests.

Figure 4-1. OraApps: Critical Activities portlet

Request Type	Application	# of Reqs
OraApps Application Issue	HR Application	1
OraApps Cloning Request	ERP Application	1
OraApps Report Request	ERP Application	1

Showing 1 to 3 of 3 Prev Next Maximize

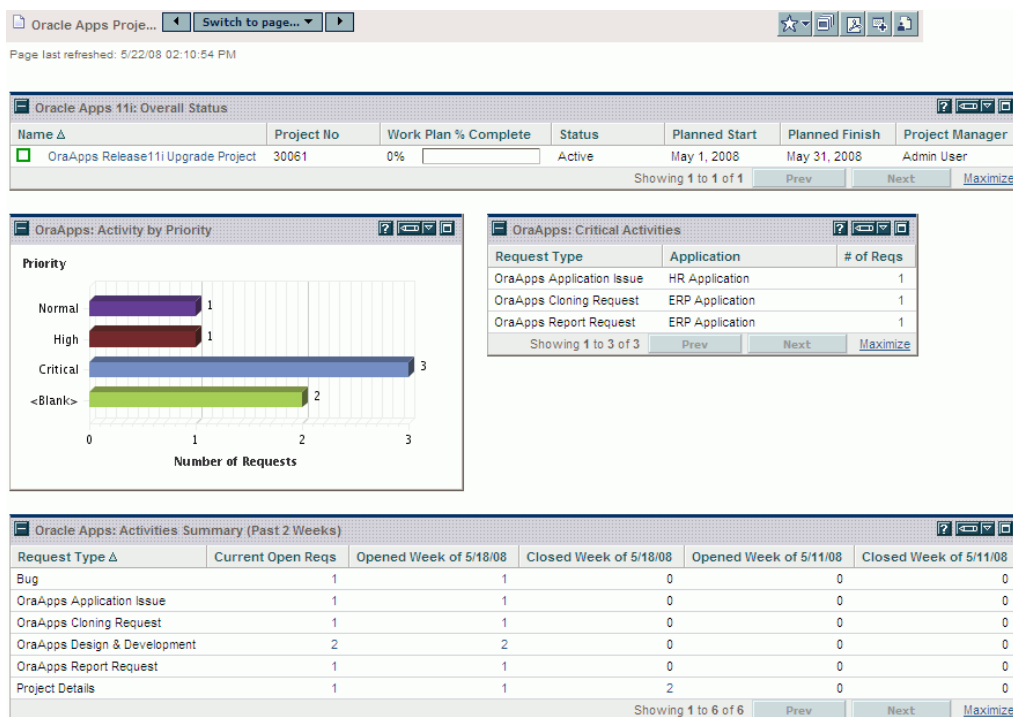
For information about adding the preconfigured Dashboard pages, see the *Getting Started* guide as necessary. The availability of particular preconfigured Dashboard pages depends on the security configuration established by your PPM Center administrator.

The following sections describe the preconfigured Dashboard pages and their individual portlets.

Oracle Apps Project Manager Dashboard Page and Its Portlets

The Oracle Apps Project Manager Dashboard page (shown in "Figure 4-2. Oracle Apps Project Manager preconfigured Dashboard page") allows project managers to view and access information needed to fulfill their business role. (For information about the project manager business role, see "Project Manager" on page 22.)

Figure 4-2. Oracle Apps Project Manager preconfigured Dashboard page



As described in the following sections, the Oracle Apps Project Manager Dashboard page includes the following portlets by default:

- Oracle Apps 11i: Overall Status portlet
- OraApps: Activity by Priority portlet
- OraApps: Critical Activities portlet
- Oracle Apps: Activities Summary (Past 2 Weeks) portlet

Oracle Apps 11i: Overall Status Portlet

The Oracle Apps 11i: Overall Status portlet includes information about the overall status of the Oracle Release 11i or Release 12 project, such as how close the project is to completion (by percentage), the project state, the scheduled start date, and the scheduled finish date. The portlet supports Release 12 even though its name was not changed.

When you click the **Edit** icon for this portlet, a page appears with, among other portlet filter fields, the default fields having the default values shown in ["Figure 4-2. Oracle Apps Project Manager preconfigured Dashboard page"](#).

Table 4-10. Oracle Apps 11i: Overall Status portlet filter fields

Field Name (*Required)	Default Value
Specific Projects	OraApps Release 11i Upgrade Project
*Project Health (green, yellow, red, and No Health check boxes)	Green, yellow, and red selected; No Health deselected

OraApps: Activity by Priority Portlet

The OraApps: Activity by Priority portlet uses a bar chart to summarize all open Oracle-related requests. The requests are grouped by priority, providing real-time data to help allocate resources to the most important tasks. Users can drill down on each entry to view specific requests.

When you click the **Edit** icon for this portlet, a page appears with, among other portlet filter fields, the default fields having the default values shown in ["Table 4-11. OraApps: Activity by Priority portlet filter field"](#).

Table 4-11. OraApps: Activity by Priority portlet filter field

Field Name (*Required)	Default Value
Request Type	All OraApps request types (see Appendix B, Request Types, on page 263)

OraApps: Critical Activities Portlet

The OraApps: Critical Activities portlet summarizes open critical Oracle-related requests, displaying information about the type and the total number of requests for each category. This information can help identify areas of the project requiring additional resources or attention. Users can drill down on each entry to view specific requests.

When you click the Edit icon for this portlet, a page appears with, among other portlet filter fields, the default fields having the default values shown in "[Table 4-12. OraApps: Critical Activities portlet filter fields](#)".

Table 4-12. OraApps: Critical Activities portlet filter fields

Field Name (*Required)	Default Value
Request Type	All OraApps request types (see Appendix B, Request Types, on page 263)
Priority	Critical
Include Closed	No

Oracle Apps: Activities Summary (Past 2 Weeks) Portlet

The Oracle Apps: Activities Summary (Past 2 Weeks) portlet displays Oracle-related request activity for the past two weeks. This includes information about the number of requests opened and closed during the last two weeks, as well as the number of open requests. Users can drill down on each total to view specific requests.

When you click the Edit icon for this portlet, a page appears with, among other portlet filter fields, the default fields having the default values shown in "[Table 4-13. Oracle Apps: Activities Summary \(Past 2 Weeks\) portlet filter fields](#)".

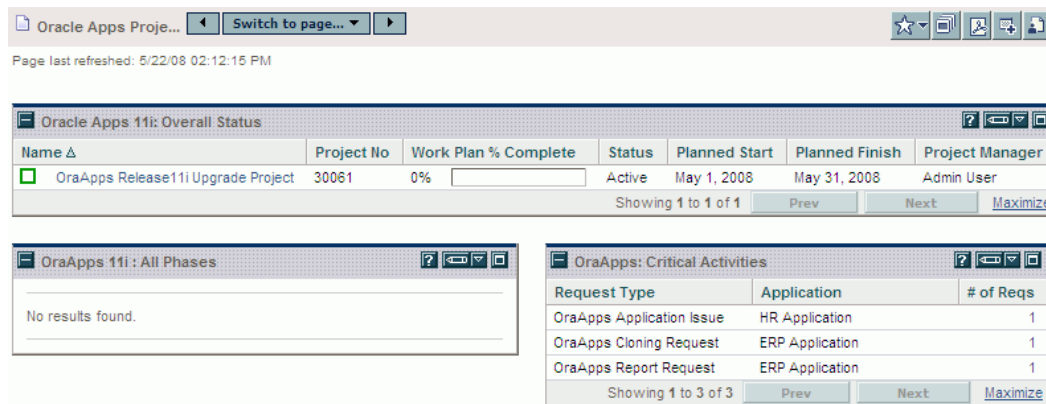
Table 4-13. Oracle Apps: Activities Summary (Past 2 Weeks) portlet filter fields

Field Name (*Required)	Default Value
Request Type	All OraApps request types (see Appendix B, Request Types, on page 263)
*Group By	Request Type

Oracle Apps Project Sponsor Dashboard Page and Its Portlets

The Oracle Apps Project Sponsor Dashboard page (shown in "[Figure 4-3. Oracle Apps Project Sponsor preconfigured Dashboard page](#)") allows project sponsors to view and access information needed to fulfill their business role. (For information about the project sponsor business role, see "[Project Sponsor](#)" on page 23.)

Figure 4-3. Oracle Apps Project Sponsor preconfigured Dashboard page



As described in the following sections, the Oracle Apps Project Sponsor Dashboard page includes the following portlets by default:

- Oracle Apps 11i: Overall Status portlet
- Oracle Apps 11i: All Phases portlet
- OraApps: Critical Activities portlet

Oracle Apps 11i: Overall Status Portlet

The Oracle Apps 11i: Overall Status portlet includes information about the overall status of the Oracle Release 11i or Release 12 project, such as how close the project is to completion (by percentage), the project state, the scheduled start date, and the scheduled finish date. The portlet supports Release 12 even though its name was not changed.

When you click the **Edit** icon for this portlet, a page appears with, among other portlet filter fields, the default fields having the default values shown in "[Table 4-14. Oracle Apps 11i: Overall Status portlet filter fields](#)".

Table 4-14. Oracle Apps 11i: Overall Status portlet filter fields

Field Name (*Required)	Default Value
Specific Projects	OraApps Release 11i Upgrade Project
*Project Health (green, yellow, red, and No Health check boxes)	Green, yellow, and red selected; No Health deselected

Oracle Apps 11i: All Phases Portlet

The Oracle Apps 11i: All Phases portlet displays the status of all phases of the project, such as how close each phase is to completion (by percentage), project state, and the scheduled finish date. The portlet supports Release 12 even though its name was not changed.

When you click the **Edit** icon for this portlet, a page appears with, among other portlet filter fields, the default fields having the default values shown in "[Table 4-15. Oracle Apps 11i: All Phases portlet filter fields](#)".

Table 4-15. Oracle Apps 11i: All Phases portlet filter fields

Field Name (*Required)	Default Value
Specific Summary Tasks	Phase I: Project Startup Phase II: Business Operations Analysis Phase III: Prototype, GAP Analysis, High Level Design Phase IV: Development and System Test Phase V: Integration and Performance Testing Phase VI: Simulations Phase VII: Cut Over Phase VIII: Post Implementation Support
*Health (green, yellow, red, and No Health check boxes)	Green, yellow, and red selected; No Health deselected

OraApps: Critical Activities Portlet

The OraApps: Critical Activities portlet summarizes open critical Oracle-related requests, displaying information about the type and the total number of requests for each category. This information can help identify areas of the project requiring additional resources or attention. Users can drill down on each entry to view specific requests.

When you click the **Edit** icon for this portlet, a page appears with, among other portlet filter fields, the default fields having the default values shown in "[Table 4-16. OraApps: Critical Activities portlet filter fields](#)".

Table 4-16. OraApps: Critical Activities portlet filter fields

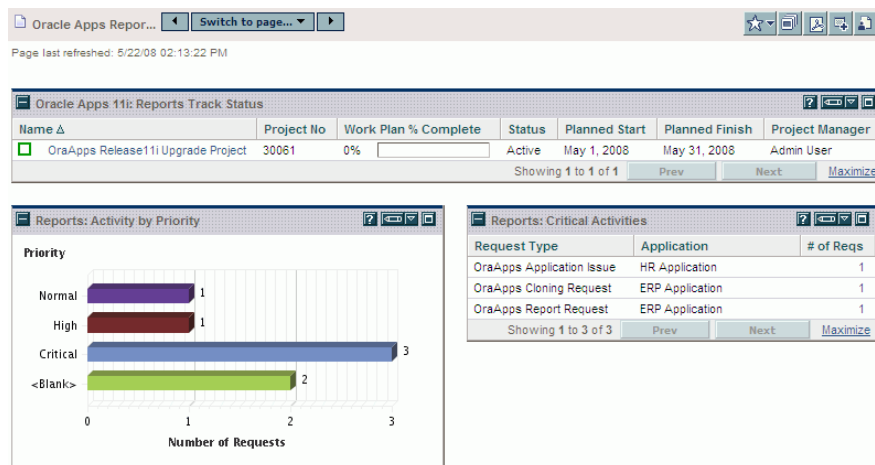
Field Name (*Required)	Default Value
Request Type	All OraApps request types (see Appendix B, Request Types, on page 263)
Priority	Critical
Include Closed	No

Oracle Apps Reports Track Owner Dashboard Page and Its Portlets

The Oracle Apps Reports Track Owner Dashboard page (shown in "Figure 4-4. Oracle Apps Project Manager preconfigured Dashboard page") allows track owners to view and access information needed to fulfill their business role. (For information about the track owner business role, see "Track Owner" on page 23.)

The Oracle Apps Reports Track Owner page is provided as a sample of a single track. Additional pages would be defined for other tracks.

Figure 4-4. Oracle Apps Project Manager preconfigured Dashboard page



As described in the following sections, the Oracle Apps Reports Track Owner Dashboard page includes the following portlets by default:

- Oracle Apps 11i: Reports Track Status portlet
- Reports: Activity by Priority portlet
- Reports: Critical Activities portlet

Oracle Apps 11i: Reports Track Status Portlet

The Oracle Apps 11i: Reports Track Status portlet displays the status of summary tasks belonging to the reports track. This information includes how close the projects are to completion (by percentage), scheduled start date, and scheduled finish date. The portlet supports Release 12 even though its name was not changed.

When you click the **Edit** icon for this portlet, a page appears with, among other portlet filter fields, the default fields having the default values shown in "[Table 4-17. Oracle Apps 11i: Overall Status portlet filter fields](#)".

Table 4-17. Oracle Apps 11i: Overall Status portlet filter fields

Field Name (*Required)	Default Value
Specific Projects	OraApps Release 11i Upgrade Project
*Health (green, yellow, red, and No Health check boxes)	Green, yellow, and red selected; No Health deselected

Reports: Activity by Priority Portlet

The Reports: Activity by Priority portlet uses a bar chart to display open Oracle-related requests that belong to the track. The requests are grouped by priority.

When you click the **Edit** icon for this portlet, a page appears with, among other portlet filter fields, the default fields having the default values shown in "[Table 4-18. Reporting Track: Activity by Priority portlet filter field](#)".

Table 4-18. Reporting Track: Activity by Priority portlet filter field

Field Name	Default Value
Request Type	OraApps Report Request

Reports: Critical Activities Priority

The Reports: Critical Activities portlet displays critical Oracle-related requests that belong to the track, including their type, and the total number of requests for each category.

When you click the **Edit** icon for this portlet, a page appears with, among other portlet filter fields, the default fields having the default values shown in "[Table 4-19. Reports: Critical Activities portlet filter fields](#)".

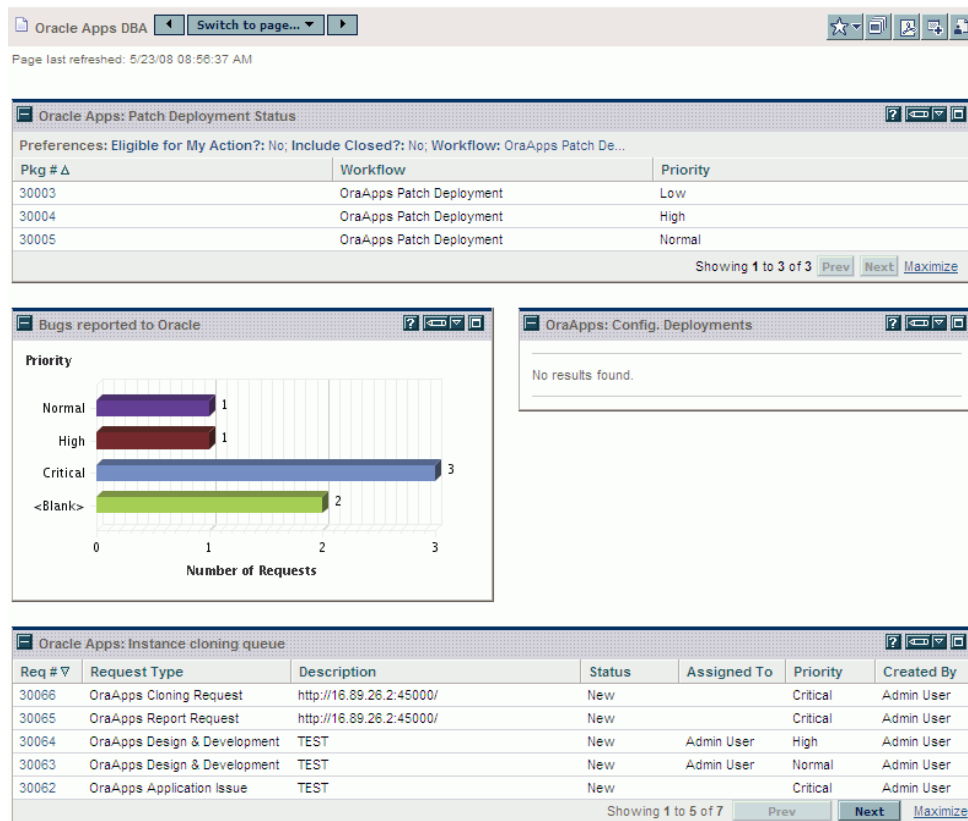
Table 4-19. Reports: Critical Activities portlet filter fields

Field Name	Default Value
Request Type	OraApps Report Request
Priority	Critical
Include Closed	No

Oracle Apps DBA Dashboard Page and Its Portlets

The Oracle Apps DBA Dashboard page (shown in "Figure 4-5. Oracle Apps DBA preconfigured Dashboard page") allows Oracle E-Business Suite DBAs to view and access information needed to fulfill their business role. (For information about the DBA business role, see "Database Administrator (DBA)" on page 24.)

Figure 4-5. Oracle Apps DBA preconfigured Dashboard page



As described in the following sections, the Oracle Apps Reports Track Owner Dashboard page includes the following portlets by default:

- Oracle Apps: Patch Deployment Status portlet
- Bugs reported to Oracle portlet

- OraApps: Config. Deployments portlet
- Oracle Apps: Instance cloning queue portlet

Oracle Apps: Patch Deployment Status Portlet

The Oracle Apps: Patch Deployment Status portlet displays general information about patch deployment packages.

When you click the Edit icon for this portlet, a page appears with, among other portlet filter fields, the default fields having the default values shown in "[Table 4-20. Oracle Apps: Patch Deployment Status portlet filter fields](#)".

Table 4-20. Oracle Apps: Patch Deployment Status portlet filter fields

Field Name	Default Value
Workflow	OraApps Patch Deployment
Include Closed	No

Bugs reported to Oracle Portlet

The Bugs reported to Oracle portlet uses a bar chart to display all open Oracle-related issues reported to Oracle and in an OA - Pending Oracle Resolution status. The requests are grouped by priority, concisely highlighting issues requiring follow-up. Users can drill down on each entry to view specific requests.

When you click the Edit icon for this portlet, a page appears with, among other portlet filter fields, the default fields having the default values shown in the following table.

Table 4-21. Bugs reported to Oracle portlet filter fields

Field Name	Default Value
Request Type	OraApps Application Issue
Status	OA - Pending Oracle Resolution

OraApps: Config. Deployments Portlet

The OraApps: Config. Deployments portlet displays a summary of the deployment activity for the past three weeks using the OraApps Customization/Configuration Deployment process. Information is grouped by object type and by week of deployment activity. Users can drill down on each entry to view specific packages.

When you click the Edit icon for this portlet, a page appears with, among other portlet filter fields, the default fields having the default values shown in "[Table 4-22. OraApps: Config. Deployments portlet filter fields](#)".

Table 4-22. OraApps: Config. Deployments portlet filter fields

Field Name (*Required)	Default Value
Workflow	OraApps Customization/Configuration Deployment
*Group By	Object Type Name

Oracle Apps: Instance cloning queue Portlet

The Oracle Apps: Instance cloning queue portlet displays the list of open cloning requests. Users can drill down on each entry to view specific requests.

When you click the Edit icon for this portlet, a page appears with, among other portlet filter fields, the default fields having the default values shown in "[Table 4-23. Oracle Apps: Instance cloning queue portlet filter fields](#)".

Table 4-23. Oracle Apps: Instance cloning queue portlet filter fields

Field Name	Default Value
Request Type	OraApps Report Request
Eligible for My Action	No
Include Closed	No

Managing Issues

This section includes the following topics:

- ["Overview of Managing Issues" below](#)
- ["Capturing and Tracking Issues" below](#)
- ["Processing User Requests" on the next page](#)
- ["Reports About Issues" on page 90](#)

Overview of Managing Issues

During an Oracle E-Business Suite implementation or upgrade, there is a constant influx of new problems, questions and requests. Each issue must be quickly addressed and resolved in order to prevent a growing backlog that can jeopardize the success of the project. Often these issues are captured in a variety of places—design documents, testing scripts, email discussions among team members, and help desks. Some problems, questions, and requests simply fall through the cracks and become visible only when the issue escalates.

The Extension provides the necessary tools for effective issue management. The Extension systematizes the process of issue analysis and resolution, so that the appropriate detailed data is gathered at the right time. The Extension recognizes and intelligently routes issues based on whether they are internal, external, or vendor (that is, an Oracle bug). Through automation and notifications, the right people work on the right issues at the right time. Then, when resolution means a change to the system, the process integrates with other Extension functionality so that development and deployment maintain the same level of control and visibility.

Throughout the life of an issue, you can monitor its status and progress through the PPM Center standard interface. Reports help assess both individual issues and the aggregate status of groups of issues. Once the implementation or upgrade is complete, you can use the issue management processes on an ongoing basis for maintenance.

This section provides conceptual and procedural information about issue management and the entities (request types, workflows, and report types) relating to issue management provided in the Extension.

Capturing and Tracking Issues

The OraApps Application Issue request type is used to capture and track the issues at an Oracle E-Business Suite implementation site. This request type provides a means for specifying information needed to resolve issues, and it works in conjunction with the OraApps Application Issue workflow.

Someone creates a request in order to resolve a specific problem. The request originator is prompted for the information needed to successfully identify and analyze the issue.

This information focuses on the Oracle environment in addition to project and business information. After providing all necessary information, the originator submits the request.

The following items are examples of the kind of information the request type uses:

- Oracle environment information
 - Application in which the problem occurred
 - Form being used
 - Operating system being used
- Project and business information
 - Business area affected
 - Perceived business benefit for resolving the issue
 - Source of the issue
 - Estimated technical impact on the organization
 - Attachments (for example, relevant documents and image files)

When all the relevant information for an issue has been specified for the OraApps Application Issue request type, the information can be analyzed and processed through to resolution.

For more information, see [OraApps Application Issue Request Type on page 267](#).

Processing User Requests

Once an OraApps Application Issue request type has been submitted, it is routed through a business process of approvals, decisions, and actions as defined in the OraApps Application Issue workflow. This workflow involves a number of steps that require review and analysis by users.

You can locate and open issues through the standard interface. The Extension provides preconfigured portlets based on your business role that help you manage Oracle-related requests. A project manager, for example, might use the OraApps: Critical Activities portlet to view open issues. You can also configure additional portlets or pages in the PPM Dashboard.

["Figure 5-1. OraApps Application Issue workflow"](#) shows the OraApps Application Issue workflow.

Figure 5-1. OraApps Application Issue workflow

OraApps Critical Requests Summary Report

The OraApps Critical Requests Summary Report displays a summary of all the Extension-related requests that have a priority level of critical. The report breaks down this information by the following categories:

- Total number of open critical requests
- Number of critical requests opened in the current week
- Number of critical requests closed in the current week
- Total number of closed critical requests

The report also lists detailed information for each issue. For more information, see ["OraApps Critical Requests Summary Report" on page 291](#).

OraApps Apps Issues Detail Report

The OraApps Apps Issues Detail Report provides details of requests created using the OraApps Application Issue request type. This report lets you filter for requests by values in the standard header fields and specific detail fields such as **TAR #**, **Business Benefit**, and **Environment**.

For more information, see ["OraApps Apps Issues Detail Report" on page 282](#).

OraApps Apps Issues Summary Report

The OraApps Apps Issues Summary Report provides summary information on requests for the OraApps Application Issue request type. This report lets you filter for requests by values in the standard header fields and specific detail fields such as **TAR #**, **Business Benefit**, and **Environment**. The Group By field lets you summarize requests using many of these detail fields.

For more information, see ["OraApps Apps Issues Summary Report" on page 287](#).

OraApps IT Demand Summary Report

The OraApps IT Demand Summary Report displays a summary of all Extension-related requests. This report gives the total counts for groups of issues that match the selection criteria. Users can group selected issues into as many as five categories and get the subtotals for each group.

Project managers for Oracle-related projects can use this report to determine priority, assigned group, and total number of project-related request types.

For more information, see ["OraApps IT Demand Summary Report" on page 293](#).

Managing Projects

This section includes the following sections:

- ["Overview of Managing Projects" below](#)
- ["Planning an Upgrade Strategy" on the next page](#)
- ["Performing a Gap Analysis" on page 94](#)
- ["Using the OraApps Release11i Upgrade Work Plan Template" on page 95](#)
- ["Reporting Status" on page 101](#)
- ["Configuring Extension Entities for Managing Projects" on page 103](#)

Overview of Managing Projects

Upgrading to any Oracle E-Business Suite release is a significant effort. The upgrade usually involves a project team crossing a variety of departments and functional and technical backgrounds, often including external system integrators. The upgrade can entail many architectural, functional, and process changes. Existing customizations and Extensions must be assessed and removed or revised. In addition, the upgrade might include implementation of new modules (for example, Oracle CRM modules).

All this complexity and effort require central control and wide visibility of the upgrade project. Tasks and deliverables must be closely monitored. Common processes must be enforced across many teams. Open issues related to the project must be tightly managed. Roles and responsibilities for each issue and task must be clearly defined.

The Extension includes content to help you manage your Oracle E-Business Suite upgrade or implementation project. The key entity for this effort, the OraApps Release11i Upgrade Work Plan Template, provides a sample outline you can modify to suit your organization's needs for upgrading to any Oracle release supported by the Extension (Release 11, Release 11i, and Release 12). The items in the work plan template have been consolidated from actual upgrade projects as well as other implementations and upgrades performed by Micro Focus and its partners, providing you with a significant jump-start to your project.

With Project Management and the OraApps Release11i Upgrade Work Plan Template, you can bring your Oracle project to fruition, allowing project team members to directly update the project as they go through their tasks, reviews, and approvals. With the project as a central information repository, team members can spend more time working on tasks and less time providing detailed status information on their own tasks and looking for status information about their dependent tasks.

Project Management can also serve as the integration point to tie together deliverables and information related to the project, as follows:

- Packages and requests that require their own detailed processes can be associated with the project entity.

- Issues raised from any source can be associated with appropriate project tasks. When a task requires its own detailed process of analysis and approvals, it can be integrated with a request entity such that there is a dependency between the two. For example, a gap analysis request can be linked to the project.
- Team members can use the PPM Center standard interface to access their current task list, reducing the potential for lost productivity. Track owners and project managers can quickly assess the status of their areas of control, so they can focus on areas requiring attention.
- Reports tailored for the Oracle E-Business Suite environment assist you in identifying situations requiring action, such as unassigned tasks that must be started or unresolved critical issues.
- The Extension includes the OraApps Status Update Request request type, which can quickly request project status. Automated executions and notifications reduce the need to manually track down status, allowing more time to be spent on tasks.

Planning an Upgrade Strategy

The OraApps Release11i Upgrade Work Plan Template, provided with the Extension, is modeled after processes that have been used in real-world upgrade projects in Oracle E-Business Suite systems. When planning an upgrade to any Oracle release supported by the Extension, open the OraApps Release11i Upgrade Work Plan Template and decide how these processes apply to your particular situation before beginning the upgrade project. The work plan template is a model or blueprint for upgrading, and it does not cover every possible configuration or situation.

Note:

The OraApps Release11i Upgrade Work Plan Template supports Oracle Release 12 as well as Release 11 and Release 11i, even though its name was not changed.

Planning an upgrade strategy includes planning a project and integrating Project Management and Demand Management, as discussed in the following sections.

Planning an Upgrade Strategy

The OraApps Release11i Upgrade Work Plan Template, provided with the Extension, is modeled after processes that have been used in real-world upgrade projects in Oracle E-Business Suite systems. When planning an upgrade to any Oracle release supported by the Extension, open the OraApps Release11i Upgrade Work Plan Template and decide how these processes apply to your particular situation before beginning the upgrade project. The work plan template is a model or blueprint for upgrading, and it does not cover every possible configuration or situation.

Note:

The OraApps Release11i Upgrade Work Plan Template supports Oracle Release 12 as well as Release 11 and Release 11i, even though its name was not changed.

Planning an upgrade strategy includes planning a project and integrating Project Management and Demand Management, as discussed in the following sections.

Integrating Project Management and Demand Management

Your upgrade strategy also includes integrating the Project Management capabilities with the issue tracking capabilities in Demand Management. Demand Management requests and Project Management projects and tasks can be linked to each other through informational relationships. This allows requests to be part of project initiatives and gives instant visibility to detailed activities that support the upgrade project.

You can link requests to projects using the following:

- **References** tab in the Task Details page, accessed from the Work Plan page
- **Reference Additions** section of the **References** tab in the Project Overview page

The PPM Dashboard lets you view references related to tasks and projects using the Request References portlet. Add this portlet to your PPM Dashboard and then personalize the portlet to show references that are relevant to upgrades.

Integrating Project Management and Demand Management, and then personalizing the PPM Dashboard to include the Request References portlet gives you quick access to information about the reference types and when the references were added.

Performing a Gap Analysis

An Oracle E-Business Suite upgrade project can result in significant changes to current business processes and in additions or customizations to the Oracle environment to accommodate the changed business processes.

Assessment of the differences between existing business processes and functionality or capability provided by a packaged application is referred to as “gap analysis.” Each identified difference, or “gap,” is analyzed to determine how it will be handled. For example, Oracle functionality might be extended or an existing business process might be changed.

Members of the upgrade project team who have the business role of project manager, track owner, or project sponsor can request a gap analysis using the OraApps GAP Analysis Request request type. This request type captures business information needed to analyze a gap and uses the OraApps GAP Analysis Process workflow to enforce a process that includes approval, assignment, and analysis steps for the gap analysis.

The project team uses information gathered from the gap analysis to help plan their upgrade tasks. The analysis gives the team a better picture of the resources, time, and effort needed to successfully complete the upgrade.

Steps in the OraApps GAP Analysis Process workflow include the following:

- A project manager assigns a priority and a business lead to the request.
- The business lead can either reject the request (which sends it back to the project manager) or accept it. If the request is accepted, the business lead assigns a team to the gap analysis.
- The gap analysis team determines where the gap is in the Oracle functionality. They also determine whether the gap can be closed by using a code, setup, or process change.
- The gap analysis team creates either an enhancement request or a setup change.
- The gap analysis request is closed.

For information on configuring this workflow process, see ["OraApps GAP Analysis Request Request Type" on page 104](#) and ["OraApps GAP Analysis Process Workflow" on page 105](#).

Chapter 6: Using the OraApps Release11i Upgrade Work Plan Template

The following sections provide an overview of the OraApps Release11i Upgrade Work Plan Template and describe its structure and how to customize work plans.

Overview of Work Plan Template

The OraApps Release11i Upgrade Work Plan Template helps maintain visibility and control of your Oracle E-Business Suite upgrade. The work plan template includes items typically performed during an upgrade. These items have been consolidated from actual upgrade projects, as well other implementations and upgrades performed by Micro Focus and its partners. The template works with any Oracle release supported by the Extension (Release 11, Release 11i, and Release 12).

The OraApps Release11i Upgrade Work Plan Template, described in this section, is a sample outline. You can modify the work plan template to suit your organization's needs for upgrading to any release supported by the Extension.

When you create a project, you must specify a project type, which can specify the work plan template to be used for projects of that type.

To access and edit a template that is based on the OraApps Release11i Upgrade Work Plan Template:

1. Log on to PPM.
2. From the menu bar, select **Open > Administration > Project Types & Templates > Manage Work Plan Templates**.
The Manage Work Plan Templates page opens.
3. Select **OraApps Release11i Upgrade Work Plan Template**, select another template that was previously copied from it, or make a new copy of it.

We recommend that you start with a copy of the original OraApps Release11i Upgrade Work Plan Template, then rename and revise the copy, so that the original work plan template remains available. Click Copy to copy the template.

4. Revise the basic fields for the template or, to revise details of the template, click **Edit Work Plan Template**.

Work Plan Template Structure

The OraApps Release11i Upgrade Work Plan Template contains eight phases that represent a typical Oracle E-Business Suite upgrade. Each phase is a summary task, and each of the summary tasks contains sample tasks for common functional areas such as demand planning, order management, and manufacturing. The OraApps Release11i Upgrade Work Plan Template is focused on the actual upgrade, but can be used as a summary task for projects with a bigger scope.

You can configure a copy of the work plan template to reflect your organization’s needs. For example, if you have not implemented Oracle Manufacturing, you can delete the manufacturing sections.

The OraApps Release11i Upgrade Work Plan Template is shown in "Figure 6-1. OraApps Release11i Upgrade Work Plan Template" (the column widths have been adjusted for this document). The data in the columns in the work plan template is associated with each task. The columns are described in "Table 6-1. OraApps Release11i Upgrade Work Plan Template columns".

Figure 6-1. OraApps Release11i Upgrade Work Plan Template

Seq	Name	Scheduled Dura...	Scheduled Effor...	Predecessors	Resources	Role	Activity
0	OraApps Release11i Upgrade Work Plan Template	2,990.00	0.00				
1	Oracle Apps : Release 11i	2,990.00	0.00				
2	Phase I: Project Startup	199.00	0.00				
3	Project Team Management	24.00	0.00				
4	Assemble Final Project Team	5.00	0.00				
5	Arrange Cube/Office Space	3.00	0.00	4			
6	Set up E-Mail Accounts	2.00	0.00	4			
7	Set up Network Accounts	3.00	0.00	6			
8	Set Up Phones	3.00	0.00	6			
9	Set Up Badges	2.00	0.00	6			
10	Develop Preliminary Training Schedule for ...	5.00	0.00	4			
11	Project Plan	1.00	0.00				
12	Phase I: Project Team Management Complete	0.00	0.00	3			
13	Project Plan	154.00	0.00				
14	Scope/Objective/Approach Document (inclu...	22.00	0.00				
15	Estimate Assumptions	22.00	0.00				
16	Project Order of Magnitude estimate	22.00	0.00				
17	Project Schedule	22.00	0.00				
18	Key Deliverable Templates	22.00	0.00				
19	Project Organization chart	22.00	0.00				
20	Quality Plan	22.00	0.00				
21	Phase I: Project Plan Complete						

Table 6-1. OraApps Release11i Upgrade Work Plan Template columns

Work Plan Template Column	Description
Seq	Sequence number of the tasks. There are 676 tasks in the work plan template as delivered. (Figure 6-1 shows the first 20.)
Name	Name of the parent or child task.
Scheduled Duration	Typical amount of time in days required to complete the task. When a project is created from the template, this value allows Project Management to schedule the task.
Scheduled Effort	Typical effort in hours required to complete the task in an average upgrade project. This value is used to show the roll-up of effort required to complete the project.
Predecessors	Typical effort in hours required to complete the task in an average upgrade project. This value is used to show the roll-up of effort required to complete the project.
Resources	Tasks that must completed before the start of the target task. Predecessors can be individual tasks or summary tasks. Critical predecessors are included in the work plan template to show the relationship among tasks.
Role	Names of the individual resources associated with the tasks or summary tasks.
	Role required of the resources to accomplish the task.
Activity	Activity associated with the task.

Note:

The rolled-up durations shown on the work plan template change once the project is created and resourced, as the scheduling engine takes into consideration dependencies and information unavailable to the work plan template.

The work plan template has the following phases, as described in the following sections:

- Phase I: Project Startup
- Phase II: Business Operations Analysis
- Phase III: Prototype, Gap Analysis, High Level Design
- Phase IV: Development and System Test
- Phase V: Integration and Performance Testing

Phase I: Project Startup

This phase consists of tasks pertaining to the overall structure of the upgrade. These tasks are done in the weeks leading to kickoff, and include the formal kickoff meeting with the project team. This phase lays the groundwork for the project so the team members understand the tasks involved and can efficiently complete the tasks that are assigned to them.

This phase includes milestones for the completion of each major grouping of tasks. You might want to add or modify these milestones based on your tracking needs.

Key deliverables for this phase are a project readiness document, a system architecture, a work plan, and a kickoff presentation.

Phase II: Business Operations Analysis

This phase clarifies and consolidates information about the current installation, identifies key business requirements for the upgraded system, and assesses the potential to meet those requirements in the upgraded Oracle E-Business Suite instance. Current installation information includes a catalog of customizations in place in the Oracle E-Business Suite instance, and processes and interface points for business functions to be newly incorporated into the Oracle E-Business Suite instance. The Oracle functionality is analyzed to determine how well it meets the business requirements, including allowing for the elimination of existing customizations or Extensions. In a typical upgrade, most of the time spent in this phase is focused on learning the new Oracle E-Business Suite release and identifying functional gaps.

Also included are milestones marking the completion of tasks for each track or sample functional area. You might want to add or modify these milestones for sign-off on your most critical or interdependent Oracle E-Business Suite modules.

The key deliverables for this phase are a gap list and as-is process flows.

Phase III: Prototype, GAP Analysis, High Level Design

This phase begins with a gap analysis for various tracks from Phase II. As gaps are identified, they are analyzed in greater detail in order to determine solutions. High-level designs are created for each solution, and any required process changes are incorporated in the to-be process documents. This includes customizations and Extensions, reports, interfaces, and conversions.

Typically a representative from each business area formally accepts designs. At this point, all remaining technical effort should be known. Generally these tasks are conducted per business functional area per gap or per group of interdependent gaps. Often reviewers include representatives from all dependent functional areas, as well as a technical reviewer to ensure technical feasibility and consistency.

The work plan template includes the following:

- Tasks for gap analysis
- Solution design and setup
- Profiles options for each track and sample functional area

Also included are milestones marking the completion of tasks for each track or sample functional area. You might want to add or modify these milestones based on your tracking needs. For example, you might want to include milestones for sign-offs of to-be process flows, prioritization of interfaces, or reports readiness.

Key deliverables for this phase are final to-be process flows, solution designs for all gaps, setup and profile options, and sign-off on the unit (business) test scripts.

Note:

The OraApps GAP Analysis Request request type is designed to manage the process each gap must pass through to complete successfully, by attaching requests using this request type to the appropriate tasks in the plan.

Phase IV: Development and System Test

At this phase the detailed designs are finalized for conversion, interfaces, reports, and customizations. Coding of conversion, interfaces, reports, and customizations is done, and unit tests are run.

The work plan template includes milestones marking the completion of for each track or sample functional area. You might want to add to or modify these milestones based on your tracking needs. For example, you might want to include milestones for sign-offs of exit criteria, successful completion of initial testing, or training plans.

The key deliverables for this phase are unit-tested interfaces, reports, conversions, customizations, enhancements, and sign-offs of exit criteria.

Note:

You can use the specialized development request types that are part of the Extension to facilitate development and increase visibility to development efforts on the project. These request types are as follows:

OraApps Conversion Request

OraApps Enhancement Request

OraApps Interface Request

OraApps Report Request

OraApps Setup Change Request

You can use the OraApps Customization/Configuration Deployment workflow to manage the code migrations for each gap by attaching packages using this workflow to the appropriate tasks in the plan.

Phase V: Integration and Performance Testing

In this phase all required test scripts for integration and performance testing are created. This might involve consolidation or adaptation of existing test scripts, generation of new test scripts to bridge between different processes, and design of sample test data. For performance test scripts, identification of representative transactions and loads should be completed. Baseline performance should also be defined, if possible.

Integration testing is performed to ensure that business processes, developed code, interfaces, and conversions function as expected. This is usually the first time these elements are tested end-to-end in the same instance. Performance testing is done to ensure that system performance is acceptable for users and to perform final tuning of programs. Disaster recovery processes should be defined.

The work plan template provides a single combined phase for integration and performance testing. Additional rounds of testing (such as acceptance testing) should be planned, and they can be added to the work plan template. Also included are milestones marking the completion of the following:

- Build
- Data population using conversions
- Functional testing
- Exit criteria sign-off for each round of testing
- Project management and infrastructure tasks associated with the phase

You might want to add to or modify these milestones based on your tracking needs.

The key deliverables for this phase are a fully operational system that is at least 90% complete and a document identifying production readiness.

Phase VI: Simulations

This phase is a dry run of the cutover event in order to streamline the time needed to implement the Oracle upgrade and minimize the impact to your business. This event is usually conducted around the clock, and it is used to set a baseline so that management knows throughout the cutover event how the project team's status compares to agreed upon schedules. The simulation is also used to ensure that the workload is balanced and that all tasks can be accomplished by the people assigned to them.

The work plan template includes milestones marking the completion of tasks for each track or sample functional area. You might want to add to or modify these milestones based on your tracking needs. For example, you might want to include milestones to mark completion of each simulated iteration, or to identify a formal "go-live" sign-off milestone after simulations are complete.

The key deliverable for this phase is the final cutover plan, including an estimate of the time required for the implementation of the Oracle upgrade.

Phase VII: Cut Over

This phase addresses the actual upgrade and cutover to the new production system. The plans refined in Phase VI (Simulations) are executed against the production environments.

The work plan template includes milestones to mark the completion of cutover tasks and formal go-live system acceptance. You might want to add to or modify these milestones based on your tracking needs.

The key deliverables from this phase are a committed go-live decision and signed acceptance of the production system.

Phase VIII: Post Implementation Support

Once the upgrade has been completed from a technical standpoint, it is common to go through a period of additional training and heightened support as users become accustomed to the new system.

The support tasks for functional, reporting, interface, customization support track, and conversion support tracks include the following:

- Issue resolution
- Bug fixes
- Knowledge transfer

Customizing the Work Plan Template

As with any work plan template, you can open and copy the OraApps Release11i Upgrade Work Plan Template, and revise the copy.

When a user creates a project, the user must specify its project type. A project type can specify many default project settings, including the work plan template the project uses. These settings apply to all projects of that type.

The Extension is not delivered with its own project type. However, you can create a project type, or copy and customize the Enterprise project type provided with PPM Center, and set its default work plan template to the OraApps Release11i Upgrade Work Plan Template or your revision of it. Then every new project of that project type will use that work plan template as a starting point.

For more information about project types and work plan templates, see the Project Management User Guide.

Reporting Status

The following sections provide guidance for the following entities related to reporting status:

- OraApps Status Update Request request type
- OraApps Status Update Request workflow
- Report types for project-related status

For more information on these entities, see the appendixes in this document.

OraApps Status Update Request Request Type

The OraApps Status Update Request request type automates the process of gathering and reviewing status reports. This request type lets managers request information, automatically notifies group members of requirements for information, and monitors the requirements to make sure that tasks are completed. When the tasks for completing a report have been done, reviewers are automatically notified. All involved parties have access to the latest version of the report.

This request type allows users to specify basic requestor information and determine which project teams must provide status updates. The requestor can indicate the required date for the information and which individual is directly responsible for the update from each team (this responsibility information can also be defaulted). Once the request is submitted, it goes through an update request process that routes the request to the appropriate teams and individuals. The status updates are attached to the request, so all individuals with appropriate access can use the request as the repository for the project status information for the given time period.

The OraApps Status Update Request request type is designed to work with the OraApps Status Update Request workflow, described in ["OraApps Status Update Request Workflow" below](#).

For more information, see ["OraApps Status Update Request" on page 233](#).

OraApps Status Update Request Workflow

The OraApps Status Update Request workflow provides a process to automatically notify group members, follow up with them until information is received, contact the appropriate reviewers, and distribute the latest version of the reports.

Once a status update request is made, the OraApps Status Update Request workflow splits into parallel branches for each project subteam. If a status update is requested from a specific subteam, the designated team lead is notified of the request. The team lead is then repeatedly reminded of this request until the information is provided to the request.

Once all the requested status updates are provided, the requestor is notified and can review the individual information directly from the request. The entire project management team or the entire project team can review the information, as well.

For more information, see ["OraApps Status Update Request Workflow" on page 271](#).

Reports About Project-Related Status

The following sections discuss the following reports on project-related status:

- OraApps Critical Requests Summary Report
- OraApps IT Demand Summary Report

OraApps Critical Requests Summary Report

The OraApps Critical Requests Summary Report displays a summary of Extension-related requests that have a priority level of critical. Project managers for Oracle projects must identify critical issues quickly so they can either add more resources or adjust the schedule.

The report breaks down the information by the following categories:

- Total number of open critical requests
- Total number of critical requests opened in the current week
- Total number of critical requests closed in the current week
- Total number of closed critical issues

The report also lists detailed information for each issue.

For more information, see ["OraApps Critical Requests Summary Report" on page 291](#)

OraApps IT Demand Summary Report

The OraApps IT Demand Summary Report displays a summary of all Extension-related requests. This report gives the total counts for groups of issues that match the selection criteria. Users can group selected issues into as many as five categories and get the subtotals for each group.

Project managers for Oracle-related projects can use this report to determine priority, assigned group, and total number of project-related request types.

For more information, see ["OraApps IT Demand Summary Report" on page 293](#).

Configuring Extension Entities for Managing Projects

The following sections discuss configuring request types, workflows, and report types for Oracle E-Business Suite project management.

Configuring Request Types

The following sections provide conceptual and procedural information about the following request types:

- OraApps GAP Analysis Request
- OraApps Status Update Request

OraApps GAP Analysis Request Request Type

The OraApps GAP Analysis Request request type initiates the process of requesting a gap analysis for an Oracle system. Project managers, track owners, and project sponsors can use this request type to create a gap analysis. Once a request for a gap analysis has been created, the request must be attached to relevant project tasks.

The OraApps GAP Analysis Request request type captures business details and works with the OraApps GAP Analysis Process workflow, which consists of approval, assignment, and analysis steps.

To configure the OraApps Status Update Request request type:

1. Review the security information on the workflow and modify it to meet your organization's structure and needs.
2. Review the security information on the request type (user access and field security) and modify it to meet your organization's structure and needs. For example, you might want to provide only specific groups of users with the ability to create requests.
3. In the **Rules** tab of this request type in the PPM Workbench, set the default workflow to the OraApps GAP Analysis Process workflow.
4. In the **Workflows** tab of this request type in the PPM Workbench, restrict the request type to use only this workflow.
5. Enable the request type.

For more information about this request type, see ["OraApps GAP Analysis Request Request Type" on page 223](#).

OraApps Status Update Request Request Type

To configure the OraApps Status Update Request request type:

1. Decide which of your subteams will do status reports, and for each subteam, do the following:
 - a. Create **Status** and **Team Lead** fields for subteams that are not already defined.
 - b. Delete any **Status** and **Team Lead** fields that are not relevant to your project.
2. Define the defaults for each Status field (that is, **Requested** or **Not Requested**).

3. Identify team leads for the project and define them as default values in the **Team Lead** fields.
4. Review the security information on the request type (User Access and Field Security) and modify it to meet your organization's structure and needs. For example, you might want to provide only specific groups of users with the ability to create status update requests.

For more information about this request type, see ["OraApps Status Update Request" on page 233](#).

Configuring Workflows

The following sections provide conceptual and procedural information about the following workflows:

- OraApps GAP Analysis Process
- OraApps Status Update Request

OraApps GAP Analysis Process Workflow

As described in ["Performing a Gap Analysis" on page 94](#), members of an upgrade project team can request a gap analysis using the OraApps GAP Analysis Request request type. This request type uses the OraApps GAP Analysis Process workflow to carry out the gap analysis.

Using information in the gap analysis results, the gap analysis team determines the best action as follows:

- If they can find matching functionality in Oracle, they send this information back to the business lead, who decides whether to accept the functionality match.
- If they determine that a code, setup, or process change is required for bridging the gap, they manually create an enhancement request or a setup change request.

After one of these actions is taken, the gap analysis is closed.

To configure the OraApps GAP Analysis Process workflow:

1. Review the security information in the workflow and modify it to meet your organization's structure and needs.
2. In the Rules tab of the OraApps GAP Analysis Request request type in the PPM Workbench, set the default workflow to the OraApps GAP Analysis Process workflow.
3. In the **Workflows** tab of this request type in the PPM Workbench, restrict the request type to use only this workflow.
4. Modify the workflow to define any additional processing notifications required.
5. Modify the workflow to reflect local business practices.

OraApps Status Update Request Workflow

To configure the OraApps Status Update Request workflow:

1. Verify the security groups for each step and modify them, if necessary, to fit your organization's needs.
2. Remove workflow branches for any status or team lead fields that have been removed from the request type.
3. Add branches, if required, as follows:
 - a. Copy existing branches.
 - b. Update the step names.
Be sure the name of the resolution step token matches the name of the request detail field token for the status field.
 - c. Update the notifications.

For more information about this workflow, see ["OraApps Status Update Request Workflow" on page 271](#).

Configuring Report Types

This section provides information about configuring the following report types included with the Extension:

- OraApps IT Demand Summary Report
- OraApps Critical Requests Summary Report

To configure these report types:

1. Copy the reference versions of the reports and give the copies the following names:
 - OraApps IT Demand Summary Report
 - OraApps Critical Requests Summary Report
2. Enable the reports for use.
3. Configure any security restrictions required.
By default, these reports are available for all users.
4. Use hidden fields to restrict the data provided by the reports.

For more information about these report types, see the following sections:

- ["OraApps IT Demand Summary Report" on page 293](#)
- ["OraApps Critical Requests Summary Report" on page 291](#)

Managing Changes

This section includes the following topics:

- ["Overview of Managing Changes" below](#)
- ["Requesting Changes" on the next page](#)
- ["Deploying Oracle Changes" on page 113](#)

Overview of Managing Changes

Oracle E-Business Suite has a flexible architecture that allows companies to shape the system to their business needs through both configuration and customization. In a typical implementation or upgrade, a significant amount of work goes into designing, developing, and rolling out new or updated configurations and changes. Even companies primarily trying to use only standard functionality find themselves with high volumes of configurations (for example, security setup) and customizations (for example, custom reports). After implementation, business processes continue to evolve, and Oracle systems must evolve as well.

The success of an Oracle implementation, an upgrade, or ongoing production support depends on a stable, efficient process for managing these configurations and customizations. Without such a process, determining which changes have been applied to which instances can seem to be a nearly impossible task. Further, the potential for inaccurate application of changes to a given instance is high. Limiting who can apply changes reduces inaccuracies, but requires considerable time from valuable people.

The Extension provides processes and automation to help you maintain visibility to and control of the changes to your system. This packaged content has been derived from firsthand, in-depth experience by Micro Focus and its partners in implementing, upgrading, and supporting Oracle E-Business Suite through Release 12. These components allow you to quickly get control of your changes, from initial requirements to final deployment to your production system.

Maintaining control over how changes are developed is important. The Extension includes request types and workflows that model various types of development for Oracle, such as interfaces and reports. This helps with efficient use of resources and reduces the likelihood that changes will need rework late in the development cycle. When changes must be deployed, the deployment workflow and object types in the Extension enforce best practices for deployments, testing, and approvals.

The Extension provides special handling for AOL and GL objects, integrating with the Object Migrator and GL Migrator products to migrate AOL and GL setup data between Oracle E-Business Suite instances.

For more information about Deployment Management, see the *Deployment Management User Guide* and the *Demand Management Configuration Guide*.

Requesting Changes

Request types and workflows provided by the Extension take users through the process of requesting, validating, and developing changes to an Oracle system. Users can request enhancements, new interfaces, and new reports, and can set up changes and conversions. The processes are initiated by creating a request of the appropriate request type and are moved through the necessary steps by the corresponding workflows.

The following sections discuss the following kinds of changes that users can request:

- Requesting new reports
- Requesting new interfaces
- Requesting conversions
- Requesting enhancements
- Requesting setup changes

Requesting New Reports

Oracle E-Business Suite includes standard reports that provide users with information about the system's setup and transaction activity. These reports do not always fully meet the needs of an organization. For example, you might need to include additional information not provided in standard reports, such as descriptive flexfield data, or corporate standards might dictate a different report format than Oracle provides. Regardless of the reason, in these cases new or modified reports must be developed.

In the Extension, users can request creation of a new report with the OraApps Report Request request type. This request type works in conjunction with the OraApps Reports Process workflow to enforce proven practices for report development, and to help manage IT workload effectively. Once the request is initiated, it captures the following information at the appropriate point in the process:

- Report name and description
- Whether a new report is needed
- The date by which the report must be completed
- Technical analysis
- Technical impact of the report

The workflow sends the request through approval, analysis, and assignment steps. The requestor creates the high-level requirements for the report, which is then signed off. After being prioritized by the business lead, the request goes to a team lead or track owner, who assigns the request to a developer. If the developer cannot find any existing reports that match the technical requirements, a new report is created. The developer prepares the detailed technical and functional designs for the report. When these are verified against the original design for the report, the report is developed. To deploy the

report through the testing and migration phases, a package is created that uses the OraApps Customization/Configuration Deployment workflow.

For more information about this request type, see ["OraApps Report Request Request Type" on page 228](#).

Requesting New Interfaces

In an Oracle E-Business Suite system, interfaces are used to extract data from and provide data to other systems.

The OraApps Interface Request request type is used to request new interfaces for the import and export of data from Oracle. This request type works in conjunction with the OraApps Interfaces Process workflow. Once the request is initiated, it captures business information such as the following:

- Interface name and description
- Date by which the interface must be completed
- Interface type
- Estimated effort
- Technical analysis
- Technical impact of the new interface

The workflow sends the request through approval, analysis, and assignment steps. The requestor creates the high-level requirements for the interface, which is then signed off. After being prioritized by the business lead, the request goes to a team lead or track owner, who assigns the request to a developer. If the developer cannot find any existing interfaces that match the technical requirements, a new interface is created. The developer prepares detailed technical and functional designs for the interface. When these are verified against the original design for the interface, the interface is developed. When development is completed, a package is created (using the OraApps Customization/Configuration Deployment workflow) to deploy the new code through the Oracle E-Business Suite instances for testing and validation.

For more information about this request type, see ["OraApps Interface Request Request Type" on page 226](#).

Requesting Conversions

Organizations implementing Oracle often use other legacy applications and third-party software programs. Similarly, additional business functions can be moved to Oracle after initial implementation. These other applications and software programs contain data that must be brought into Oracle. However, the data must go through a one-time conversion process so that it can be imported and used by Oracle. Whether third-party conversion tools or internally developed programs are used, development effort is required to

accomplish the data conversions. In the Extension, these activities can be managed using a conversion request for each set of entities to be unpacked into Oracle.

The process of requesting new conversions requires the OraApps Conversion Request request type, which works in conjunction with the OraApps Conversion Process workflow. Once the request is initiated, it captures business information such as the following:

- Conversion name and description
- The date by which the conversion must be completed
- Estimated effort in hours
- Technical analysis
- Technical effort required to complete the conversion

After being prioritized by the business lead, the request goes to a team lead or track owner, who assigns the request to a developer. If the developer cannot find any similar data that is being converted that matches the technical requirements, a new conversion is created. The developer prepares the detailed technical and functional designs for the conversion. When these are verified against the original design for the conversion, the conversion programs are developed. A package is created to deploy the conversion programs to the appropriate Oracle E-Business Suite instances for testing.

For more information about this request type, see ["OraApps Conversion Request Request Type" on page 216](#).

Requesting Enhancements

Although Oracle E-Business Suite includes extensive functionality across a broad spectrum of business areas, the standard functionality might not fully meet your organization's needs. For example, corporate policy might require that extensive additional data be captured at certain points in the manufacturing cycle, requiring an additional form for data capture, or you might need specialized logic incorporated in your purchase order approval process. When situations such as these arise, enhancements must be developed and implemented in your Oracle E-Business Suite instances.

The following sections discuss the following request types that are provided in the Extension for requesting enhancements:

- OraApps Enhancement
- OraApps Design & Development

OraApps Enhancement Request Request Type

The OraApps Enhancement Request request type provides a means of requesting new functionality or modifying existing functionality in an Oracle system. The request could be the result of developing new products or technology, market changes, or complying with

legal changes. In the course of a typical Oracle E-Business Suite upgrade, significant changes are made to existing customizations.

The technical requirements behind the change must be reviewed, approved, and prioritized so that resources are not wasted on work that does not have significant payback for the organization. To help you make these kinds of decisions, the OraApps Enhancement Request request type gathers the following information:

- Enhancement name
- Business area affected by the change
- Application requesting the enhancement
- Date by which the enhancement is needed
- Detailed description of the enhancement
- Analysis information needed for review and approval

The request becomes the central repository for information about the change process involved, including all notes and attachments. Everyone involved with the change process has access to the request so they can view requirements and justifications.

After all approvals have been gathered and the resulting change matches the initial technical requirements, the change is deployed to the appropriate instances.

For more information about this request type, see ["OraApps Enhancement Request Request Type" on page 221](#).

OraApps Design & Development Request Type

The OraApps Design & Development request type provides a simplified development process for cases where a more extensive process does not make sense, for example, a simple customization that requires porting only. The OraApps Design & Development request type gathers the following information for a simplified development process:

- Summary description
- Specific business areas affected
- Actual business benefit that results from the request
- Analysis of the request from the designated personnel

The request becomes the central repository for information about the change process involved, including all notes and attachments. Everyone involved with the change process has access to the request so they can view requirements and justifications.

The OraApps Design & Development request type works in conjunction with the OraApps Design & Development workflow. Once the change process has been initiated, the workflow routes the request through the following steps:

- The request goes through functional analysis, review, and sign-off.
- The request goes through technical design, review, and sign-off.

- The approved request goes into development.
- A package is created and released to deployment.
- Configurations and customizations are deployed through the standard deployment cycle.

For more information about this request type, see ["OraApps Design & Development Request Type" on page 218](#).

Requesting Setup Changes

An Oracle E-Business Suite implementation can involve many setup changes, depending on the module being implemented. These include application setups, such as setups for sets of books, payment terms, payable options, and accounting periods. These setup changes vary depending on the environment and operating units. Any setup changes approved by business leads must be made in Oracle before they can be used by application users.

However, an organization might not have the process in place to define and execute these setup changes. Any setup change process that currently exists might not be able to capture all the necessary information to successfully perform the setup change.

The OraApps Setup Change Request request type lets users capture important setup information. The OraApps Setup Change Request request type works in conjunction with the OraApps Setup Change Process workflow. This workflow puts the setup change request through a cycle of analysis, approval, and assignment. Once the setup change process has been initiated, the request captures business information such as the following:

- Setup name and description
- Application
- Requestor name
- Operating unit
- Environment name

The workflow contains the following steps:

- The functional lead analyzes the change and determines the estimated effort and dependencies to complete the change, if any. The technical lead also determines the estimated completion date for the setup change.
- The technical lead approves the setup change for the environment and assigns a resource.
- The assigned developer determines whether the setup change is a manual setup or an Micro Focus-migrated setup. A package can be created for migrations handled by Object Migrator or GL Migrator. Other migrations are handled manually.
- The assigned developer sets up the environment.
- For Micro Focus-migrated setups, the developer creates the package.

- The developer documents the setup changes.
- The request closes successfully.
- If the technical lead does not approve the request, the requestor provides more information and resubmits it.

For reference about this request type, see ["OraApps Setup Change Request Request Type" on page 230](#).

Deploying Oracle Changes

Over time, and especially when multiple development efforts are underway, it can be challenging to make sure you have visibility to and control over the changes made to a given Oracle E-Business Suite instance. The problem grows larger if you are supporting multiple testing instances. But it is critical to your business to maintain visibility to and integrity of your mission-critical Oracle E-Business Suite production instances.

Once development of new or modified functionality is complete, the changes must be deployed to the appropriate testing instances, validated, and then deployed to production. Often multiple changes must occur together to implement the new functionality. The Extension includes the OraApps Customization/Configuration Deployment workflow, providing proven practice expertise and standards to help you control your deployment processes. This workflow emphasizes the reviews required at the end of the deployment cycle (for example, a performance review) to make sure that the modification is fully tested and complies with your organization's standards.

- An Deployment Management package is created that specifies the customizations and configurations ready for deployment. The package then uses the following deployment process steps:
- Modifications are applied to the DEV environment to allow unit testing. AOL or GL objects bypass this step, since these online configurations would be provided manually in the DEV environment.
- The package moves through the deployment stages as a coordinated entity rather than an individual package. This avoids partial deployment of the overall change.
- Unit testing is performed by the assigned user. A successful test moves the package forward.
- The package goes through a technical review where actual code and scripts are examined. If the package passes the review, it moves to the next step in the workflow.
- The package is deployed from the DEV instance to the TEST instance. If the deployment is successful, the customization or configuration goes through QA testing. If QA testing is successful, the package moves to the next step.
- If the object type for the package is database-related, it is reviewed by a DBA. If the object type is not database-related, the package skips the DBA review and goes to the Performance Review step.

- The performance architecture team reviews the package before it goes to the production instance. If the package review is successful, it moves forward through the workflow.
- The package is deployed from the TEST instance to the PROD instance.
- When all the package lines have been successfully deployed to the PROD instance, the package is closed. If work must be done as part of the Rework Package step, the work is performed and the package starts again at the beginning of the deployment cycle.

The following sections discuss the following deployment subjects:

- Overview of AOL and GL migration
- AOL or GL migration execution
- AOL Concurrent Request Log or GL Concurrent Request Log
- Overview of FNDLOAD object types

Overview of AOL and GL Migration

This section describes how to execute Extension-enabled Object Migrator and GL Migrator entities from within Deployment Management. The Migrate Concurrent Program (AOL:Conc Prog object type) program is used as an example. The program fields are shown in "[Figure 7-1. Migrate Concurrent Program \(AOL:Conc Prog object type\) fields](#)".

Figure 7-1. Migrate Concurrent Program (AOL:Conc Prog object type) fields

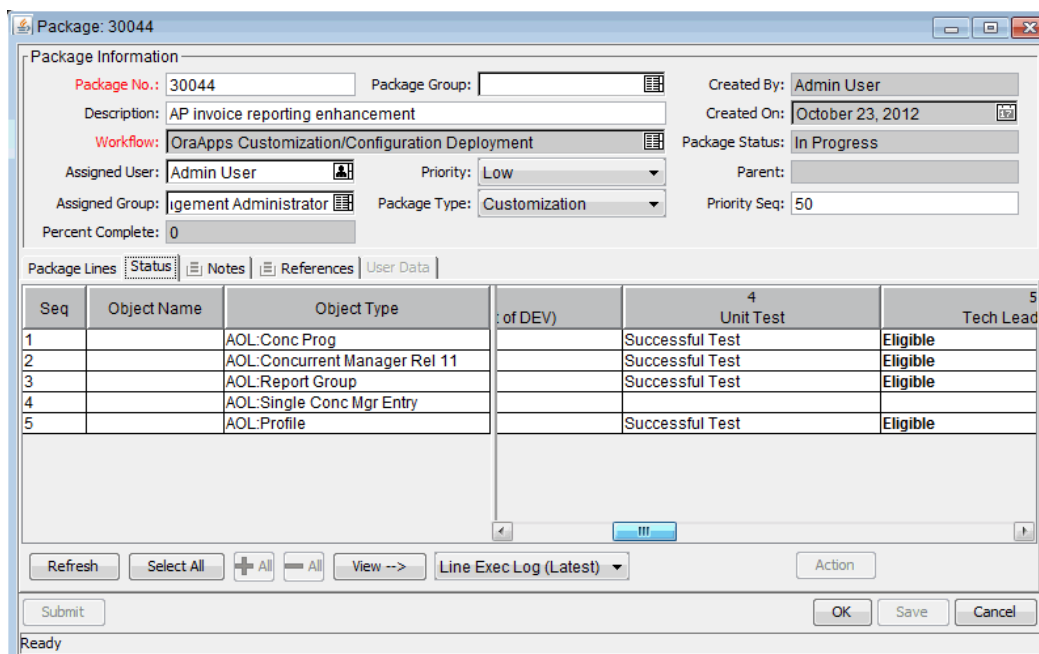
The screenshot shows a dialog box titled "Add Line" with a blue header. It contains the following fields and controls:

- Object Type Information:**
 - Object Type: AOL:Conc Prog
 - Sequence: 1
 - Application Code: None
- Parameters / User Data:**
 - Source Application: (text field)
 - Selection Type: Specific Object (dropdown menu)
 - Conc Program: (text field)
 - New User Program Name: (text field)
 - Conc Program From: (text field)
 - Conc Program To: (text field)
 - Version Label: (text field)
 - Version Desc: (text field)
 - Overwrite if Exists: Yes No
 - Partials Allowed: Yes No
- Buttons:** Clear, OK, Add, Cancel
- Status Bar:** 'AOL:Conc Prog' parameters loaded.

The Extension includes an object type for each specific migrator in Object Migrator and GL Migrator. The fields are similar to the fields used when running Object Migrator or GL Migrator directly from Oracle. Both the application name and the object name are validated against the initial environment in the workflow. The Version Label and Version Desc (description) fields are used only if the workflow is configured to use the Object Archive. You do not specify the source and destination database for each object. As the package line travels through the workflow, the workflow engine determines the source and destination database at each step.

"Figure 7-2. Package including AOL objects" shows a package using an AOL migration workflow.

Figure 7-2. Package including AOL objects



Some fields that are visible when Object Migrator runs in Oracle might be hidden by default in PPM Center. This reflects the most common usage in an automated deployment environment. As your organization may require, these fields can be displayed.

The Extension includes special handling of Object Migrator and GL Migrator migrations. When an action step in a workflow is executed, Deployment Management submits a concurrent request in Oracle to run Object Migrator or GL Migrator. When the action step completes, it prints the request ID for the concurrent request and displays the status of the concurrent request. The Oracle log and output files are attached to the PPM Center execution log.

AOL or GL Migration Execution

Every time you execute a migration step for a package line containing an AOL or GL entity, Deployment Management uses the associated workflow to determine the source and destination databases. Deployment Management then maps the values provided on the package line to the fields from the Concurrent Program definition of the specific Object Migrator or GL Migrator installation. After the mapping is complete, Deployment Management submits a concurrent request to Oracle to run the Object Migrator or GL Migrator installation.

If a package includes more than one AOL or GL object, Deployment Management automatically launches the AOL or GL migrations in order (as set in the object type). For example, all the Value Set migrations would have to complete before the Concurrent Program migration requests would start. If there are any migration failures, the PPM Server fails all dependent concurrent requests and updates the Deployment Management package appropriately.

Deployment Management monitors the status of the concurrent request. When the request is complete, Deployment Management updates the package line with the appropriate status and attaches the concurrent request output and log files to the PPM Center execution log.

Note:

If you are migrating multiple AOL or GL object package lines, Deployment Management automatically runs the concurrent requests serially, based on object dependencies.

If you use environment groups in your workflows, the workflows must be configured for serial execution in order for the AOL or GL object dependencies to be considered.

AOL Concurrent Request Log or GL Concurrent Request Log

You can view the report output of the Object Migrator or GL Migrator concurrent request directly from Deployment Management, which launches a Web browser session to display the report.

Using the Extension, Deployment Management is able to track and report on concurrent requests triggered by Micro Focus products. You can view the AOL Concurrent Request Log or the GL Concurrent Request Log for package activities through the Package Status window.

To view a Concurrent Request Log for a particular package line:

1. Log on to PPM.
2. From the menu bar, select **Open > Administration > Open Workbench**.

The PPM Workbench opens.

3. From the shortcut bar, select **Deployment Mgmt > Packages**.

The User Workbench opens.

4. Query and highlight the target package.

5. Click **Open**.

The Package: <#####> window for the target package number opens.

6. Click the **Status** tab.

7. Select the package line for which you want to view the Concurrent Request Log.

8. From the drop-down list at the bottom of the Package: <#####> window, select **Line Exec Log (Latest)**.

9. Click **View**.

The PPM Center execution log opens. The **Project and Portfolio Management Concurrent Request Output** section at the bottom of this screen contains embedded links to the Logfile with migration execution details and the Outfile with the migration report.

Overview of FNDLOAD Automation Object Types

The following object types automate the use of Oracle's FNDLOAD utility for migrations:

- The AR:Phone Country Codes object type is a setup entity that is used to migrate the standard country codes used for the phone number formats of various countries in Oracle's Financial Module Accounts Receivable (AR).
- The INV:Units of Measure object type is a setup entity that is used to migrate the Units of Measure (such as gallons, feet, and minutes) used for various inventory items in Oracle's Manufacturing Module Inventory (INV).

Caution:

If you use either object type as a template to build custom objects for other entities that you intend to use with FNDLOAD, you must fully understand the functionality of those entities and create custom FNDLOAD control (.lct) files for each one. You must ensure that the control file includes all the business logic and validations needed to maintain the referential integrity for Oracle entity IDs across the instances during a migration. Using a control file that does not include the necessary business logic and validations can lead to data corruption at the destination.

The FNDLOAD utility uses a control (.lct) file that defines individual entities. For these object types, the .lct file defines Phone Country Codes or Units of Measure. FNDLOAD extracts (exports) data from the source environment and creates an .ldt file, which is a text file whose format is defined by the .lct file.

Using the Oracle-defined .lct files, both the AR:Phone Country Codes object type and the INV:Units of Measure object type perform any of the following migration options you specify in the Migrator Action field of the object type:

- **Export.** The object type exports the Phone Country Codes or Units of Measure from the source and populates the .ldt file you specify. Package line execution provides the link to the FNDLOAD Execution Log file at the source.
- **Import.** The object type imports into the destination the Phone Country Codes or Units of Measure in the .ldt file you specify. The user must pass the .ldt file name for the data to the object type. Package line execution provides the link to the FNDLOAD Execution Log file at the destination.
- **Migrate.** The object type migrates the Phone Country Codes or Units of Measure from the source to the destination in one step. The object type exports the data from the source, creates the .ldt file, sends the .ldt file using FTP to the destination, and imports the .ldt file into the destination. Package line execution provides the links to the FNDLOAD Execution Log files at the source and destination.

For all of these migration options, consider the following:

- The object type expects the .lct files at the source and destination to be compatible.
- The AR:Phone Country Codes object type supports migration of all Phone Country Codes or a specific one.
- The INV:Units of Measure object type supports migration of all Units of Measure or a specific one.

Managing Instances

This section includes the following topics:

- ["Overview of Managing Instances" below](#)
- ["Cloning Requirements" on the next page](#)
- ["Cloning Instances" on page 121](#)
- ["Cleaning Up After Cloning" on page 123](#)
- ["Configuring Instance Management" on page 124](#)

Overview of Managing Instances

An Oracle system consists of one or more application file systems, an application database, and associated Oracle home directories and other installed entities. In addition, there are multiple supporting services required to make the system functional. These systems must be maintained and periodically cloned without risking the stability of the system.

Cloning is complex in Oracle E-Business Suite Release 11i, primarily because of the many processes and components involved. Simply replacing all the components of an existing instance does not provide a duplicate functioning system. Several configuration files must also be modified for the clone to work. In addition, the Oracle Rapid Install installation process makes use of Oracle Universal Installer, which writes information about the installation into a binary file. Copying the binary file to a different instance invalidates its contents. As a result, patches might not be applied correctly to those components.

Oracle publishes guidelines that identify the process by which cloning an instance should be performed. The process consists of a series of manual steps that include running cloning scripts provided by Oracle, as well as deleting and copying the database files and other application files.

Typically, an instance might need to be cloned for the following reasons:

- To effectively test patches against a production system without risk, a duplicate of the production system must be created and the patch applied to that copy. After it is clear that the patch does not cause any problems, the patch can then be applied to the actual production system.
- Periodically refreshing a test instance from an active production instance can provide more realistic results for test cases.
- An existing instance might need to be moved to a different machine.
- Multiple training instances might need to be created. These training instances can then be refreshed after a series of classes.

A given development instance might have become unstable due to multiple teams working on different customizations. The development instances must be deleted and

cloned with a copy from the Oracle 11i Reference instance. A typical set of tasks to complete this process would include the following:

- Deleting and backing up the development instance
- Extracting the configuration files from the development instance
- Copying the application code from the Oracle 11i reference instance to the development instance
- Copying the database files
- Restoring the saved configurations after the copy operation

Due to the complexity of the process used for cloning, there are several steps during which user error can cause major problems, including the steps in which database and application files must be deleted and overwritten. Manually executing the entire process increases the possibility that some crucial item might be overlooked.

The Extension provides a model workflow for cloning Oracle E-Business Suite Release 11i instances. This workflow enforces a process patterned after Oracle's recommended cloning process.

Automating the process using PPM Center products reduces risks and provides a means to replicate the cloning process. PPM Center also provides a history of the files and databases affected. Having an audit trail of the executions, decisions, justifications, and people responsible improves current and future cloning implementations by providing accurate and complete information, as well as forcing a standard process to which everyone in the organization adheres.

Caution:

The Extension supports cloning only Oracle Release 11i instances.

Note:

The Extension lets you automate the cloning process across multiple instances. After cloning is complete, the Extension reports all the customizations and patches that have been lost during the cloning process and must be applied again. The Extension can then automatically apply the customizations and patches to the newly cloned instance.

Caution:

The tools provided are templates that must be reviewed and configured for your environment before use. Contact Micro Focus Software Support to facilitate a successful implementation and see the Micro Focus Software Support Web site:

<https://softwaresupport.hpe.com>

Cloning Requirements

You must adhere to the following considerations for the cloning functionality in the Extension to work correctly:

- The source and target systems must have the same type of Oracle database (for example, VISION, fresh install), base language, default territory, APPL_TOP character set, server and node configurations, and platform.
- Perl must be installed on the target system to run the cloning scripts provided by Oracle.
- Adequate disk space must exist on the target system directories.
- The cloning patch from Oracle must be applied to the target system.
- The `config.txt` file should exist for the target system and should not be located under `$APPL_TOP`, `$OA_HTML`, `$JAVA_TOP`, or the database files directory, because these directories are purged during the cloning process.
- If a live instance is to be used as the source system, its application and database services must be shut down before initiating the cloning process and kept down until all the file copies are completed.
- If a cold backup is to be used as the source, the files or archives of files for `$APPL_TOP`, `$OA_HTML`, `$JAVA_TOP`, and database files should reside in different directories.
- A trace file to be used as a template for re-creating control files must exist in the directory where the source database files are stored. You can generate this file using the `kacc_ora_ctrlfile_template.sql` script delivered with the Extension. This script should be run against the source database. A trace file is created in the `$UDUMP` directory. Rename the trace file to `kacc_ora_ctrlfile.trc` and place it in the source database files directory.

These considerations are based on the process outlined in the *Cloning Oracle Applications Release 11i* white paper published by Oracle.

Cloning Instances

The OraApps Cloning Request request type and the OraApps Cloning Process workflow work together to streamline the process of initiating and tracking a request to have an Oracle E-Business Suite instance cloned. The request type provides a means for specifying information needed for the cloning. The definitions include the following information:

- Source environment
- Destination environment
- Option to clone the database portion of the system
- Date by which the cloning process must be completed

The requestor is also prompted for business information for the benefit of the business or technical analyst. This information includes the following:

- Business area affected
- Perceived business benefit of cloning

Business and technical analysts can record in the request type the results of their analysis and estimate the technical impact of the request. Requirements and justification details are attached to the request so that it serves as a central repository for information.

The OraApps 11i Cloning object type and the OraApps 11i Cloning workflow clone an Oracle instance by automating the execution of major steps in the process. Having all the environment information correctly set up through PPM Center causes only the appropriate directories to be cloned.

Information pertinent to the environments to be cloned is captured by the object type. This information is to be used during the execution of commands that clone the instance. Approval and analysis steps throughout the workflow cause the correct analysis and decisions to be made by the appropriate personnel before any execution is performed.

Caution:

Cloning your Oracle instance is a complex process, and misconfigurations can easily result in unexpected consequences. This object type and its associated workflow provide a template for cloning your instances, but they require tailoring for your environment. Contact Micro Focus Software Support to facilitate a successful implementation and see the Micro Focus Software Support Web site:

<https://softwaresupport.hpe.com>

The OraApps 11i Cloning object type supports cloning for any Oracle E-Business Suite Release 11i instance that does not use AutoConfig, which is a configuration option for Oracle E-Business Suite Releases from 11.5.1 through 11.5.6.

Cloning an instance through PPM Center allows the entire process to be monitored from initiation to completion. Information can be shared through references that can be attached to either the request or the package.

When a clone is needed, a request is created using the OraApps Cloning Request request type and the OraApps Cloning Process workflow. The required fields are specified by the requestor to provide the necessary information for the correct groups to be able to decide whether to approve the request. Once the request is submitted, the business analysts, team leads, and DBAs are also prompted to specify appropriate fields to provide information for the next group of decision makers. The fields might or might not be required, depending on the group eligible for that particular step in the workflow.

Once approvals from the various groups are received, a package is created to execute the clone. The package should be defined to use the OraApps 11i Cloning workflow and the OraApps 11i Cloning object type. Once the package line is submitted, the user is asked to determine whether the destination system exists and to install it if it does not.

The requirements for cloning are verified, and if any requirements are not met, the user is prompted to make required changes. Once all the requirements have been met, the user is advised to create a backup of the destination instance. The user can decide whether a backup of the destination is needed.

The cloning scripts are then run to shut down the application services and to save the destination configuration information to be used later in the process. If the database is to be cloned along with the application code, steps for deleting and copying application files and database files are executed. There are appropriate analysis and rerun steps to address errors that might occur during the course of these tasks.

Once the preceding steps have been successfully executed, the cloning scripts are run to start up the application services and restore the configuration files with the appropriate information specific to the target instance. The user is then asked to perform any final tasks that are required and to test the destination instance before closing the package line.

Once the package is closed, the request also closes with the appropriate status.

For more information about configuring this process, see ["OraApps Cloning Request Request Type" on page 126](#) and ["OraApps Cloning Process Workflow" on page 127](#).

For more information about performing cleanup tasks, see [Cleaning Up After Cloning](#).

For more information about the entities discussed in this section, see the following sections:

- ["OraApps Cloning Request Request Type" on page 213](#)
- ["OraApps Cloning Process Workflow" on page 243](#)
- ["OraApps 11i Cloning Object Type" on page 192](#)
- ["OraApps 11i Cloning Workflow" on page 276](#)

Cleaning Up After Cloning

The OraApps 11i Cloning workflow contains the following steps that take place after the cloning of an instance has been completed:

- The user is prompted to do additional tasks that might be required.
These tasks could include modifying the Web configuration file, updating the self-service fields, and relinking the Oracle executables.
For more information about these tasks, see the *Cloning Oracle Applications Release 11i* white paper published by Oracle.
- A test of the resulting destination instance is requested. If there are no problems, the package line is closed as successful.

After cloning an instance and responding to the prompts as previously described, an Environment Refresh of that instance must be performed to synchronize the Micro Focus data and determine what entities must be reapplied. After using the Refresh function, Deployment Management can update its audit history and its in-process packages. This update is performed using the Environment Refresh Workbench functionality in PPM Center.

An Environment Refresh does the following:

- Identifies open package lines that have migrated through the environment and are now inaccurate due to the software changes. For example, if you refresh the DEV and QA environments with the current PROD environment, the package lines become inaccurate.
- Updates the list of affected package lines, as necessary, including adding or deleting lines from the list.
- Updates the Deployment Management internal object inventory tables to reflect the physical refresh.

This operation copies the audit history from the source environment to the audit history of the refreshed environment. Since they are now physical matches of each other, their audit history should be the same.

- Updates the open package lines in the refresh list and resets them so they are eligible for migration into the refreshed environment again.

For more information about refreshing environments, see the *Deployment Management Configuration Guide*.

Configuring Instance Management

The following sections discuss the following subjects:

- Configuring environments
- OraApps Cloning Request request type
- OraApps cloning-related workflows
- OraApps 11i Cloning object type
- Special command `ksc_oa_copy_clone_scripts`

Configuring Environments

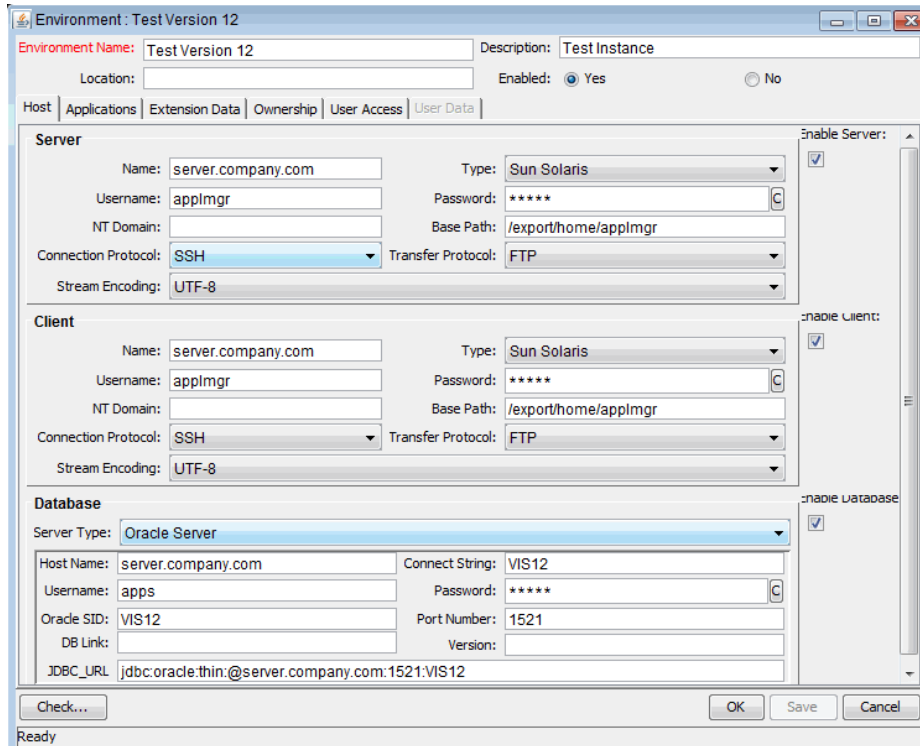
To configure environments, you provide server, client, and database environment information in the PPM Workbench Environment window's **Host** tab and its **Extension Data** tab, **Oracle Applications** subtab. For general configuration procedures, see "[Configuring Environments](#)" on page 41.

The tools provided are templates that must be reviewed and configured for your environment before use. Contact Micro Focus Software Support to facilitate a successful implementation and see the Micro Focus Software Support Web site:

<https://softwaresupport.hpe.com>

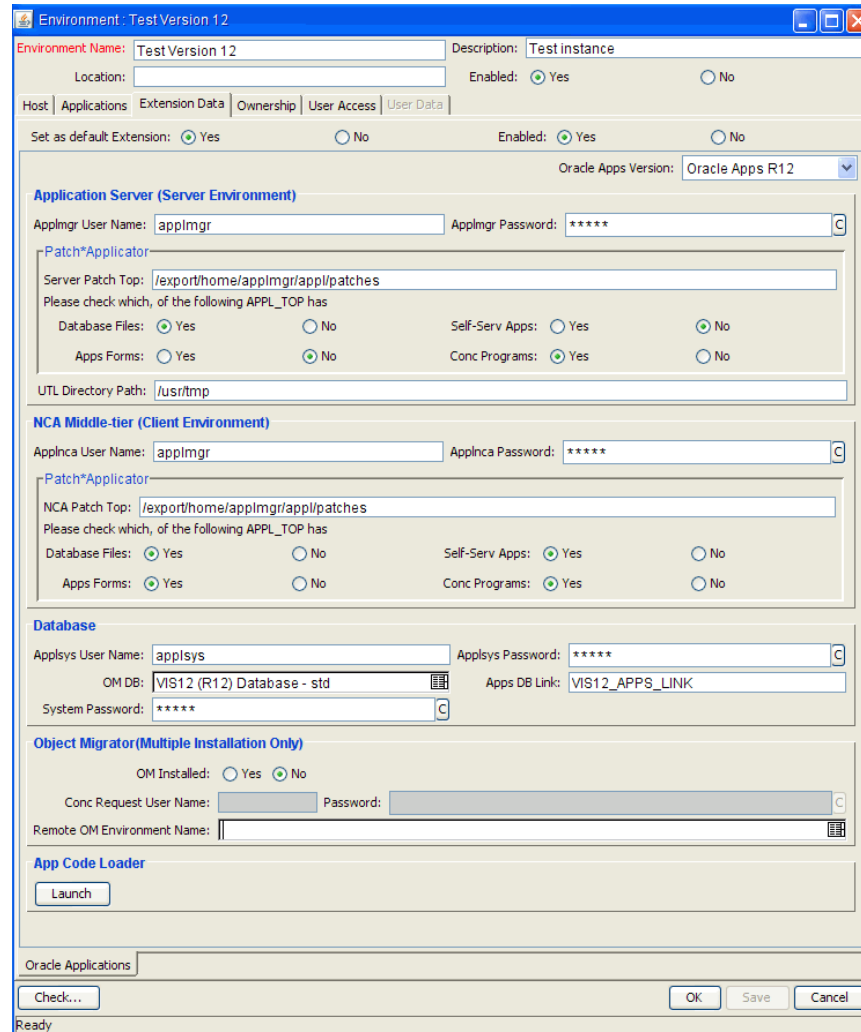
"[Figure 8-1. Host tab sample data](#)" shows the Host tab of a sample environment.

Figure 8-1. Host tab sample data



"Figure 8-2. Extension Data tab, Oracle Applications subtab sample data" shows the Extension Data tab, Oracle Applications subtab of a sample environment.

Figure 8-2. Extension Data tab, Oracle Applications subtab sample data



OraApps Cloning Request Request Type

The OraApps Cloning Request request type lets you streamline the process of initiating and tracking a request to have an Oracle E-Business Suite instance cloned. The request type captures the business details that justify the cloning process and works in conjunction with the Oracle Cloning Process workflow to go through approval and analysis steps. If cloning the instance is approved, an Deployment Management package is created that uses the OraApps 11i Cloning object type and the OraApps 11i Cloning workflow.

To configure the OraApps Cloning Request request type:

1. Use the Default Request Header Type.
2. Restrict the request type to allow only the OraApps Cloning Process workflow.
3. Set the OraApps Cloning Process workflow as the default workflow for the request type.

4. If you need tighter security for specific fields on the request type, configure field security setups for these fields. For example, you might want to configure the request type to allow only the DBA team to update the Analysis field.
5. (Optional) To limit who can create cloning requests, add user access restrictions.

Note:

The request type is defined to use participant security, so only users directly involved with the request can access it.

6. When you are finished configuring, enable the request type and the workflows.

For more information about the request type, see ["OraApps Cloning Request Request Type" on page 213](#).

OraApps Cloning-Related Workflows

The following sections provide information about the following workflows:

- OraApps Cloning Process
- OraApps 11i Cloning

OraApps Cloning Process Workflow

The OraApps Cloning Process workflow lets you streamline the process of cloning an Oracle E-Business Suite instance. This workflow works in conjunction with the OraApps Cloning Request request type and consists of approval and analysis steps for the appropriate resources. Once approved, the workflow creates a package in Deployment Management that completes the cloning process, using the OraApps 11i Cloning workflow, which is described in ["OraApps 11i Cloning Workflow" on page 276](#).

To configure the OraApps Cloning Process workflow:

1. Assign a security group to each workflow step so the appropriate user can act on each step.
The requestor (for example, developer, system administrator, or QA tester), business group managers, IT lead, and DBA team need authorizations for creating a cloning request.
2. (Optional) Add notifications to the workflow to help streamline the process.
3. Restrict the workflow so it can create packages using only the OraApps 11i Cloning workflow.
4. Enable the workflow.

For more information about the OraApps Cloning Process workflow, see ["OraApps Cloning Process Workflow" on page 243](#).

OraApps 11i Cloning Workflow

The OraApps 11i Cloning workflow works in conjunction with the OraApps 11i Cloning object type. When the OraApps Cloning Process workflow reaches the step where it must create the package and wait, the user creates a package that uses the OraApps 11i Cloning workflow and the OraApps 11i Cloning object type in its package lines.

The OraApps 11i Cloning workflow uses the following process:

- The workflow verifies the cloning requirements and begins copying PPM Center cloning scripts.
- The cloning scripts run in pre-clone mode to preserve the instance's configuration information.
- The database is shut down, any database files that must be deleted are deleted, and the files are copied from source to destination.
- Control files for the destination database are created.
- The database is started.
- Any concurrent requests copied from the source are cleaned up.
- Various code files are copied from the source backup directory to the destination.
- The cloning scripts are executed in post-cloning mode.

If invoked from a request, the request would get the final status and continue. To configure the OraApps 11i Cloning workflow:

1. Assign the correct security groups to each step of the workflow.
2. Restrict the workflow to use only the OraApps 11i Cloning object type.
3. Review the workflow and add steps unique to your business. This might also entail changes to the OraApps 11i Cloning object type.
4. Enable the workflow.

For more information about the OraApps 11i Cloning workflow, see "[OraApps 11i Cloning Workflow](#)" on page 276.

OraApps 11i Cloning Object Type

The OraApps 11i Cloning object type provides a means to clone an instance of Oracle E-Business Suite by automating the execution of major steps in the process. Information pertinent to the environments to be refreshed is captured in the object type. This information is to be used during the execution of commands that clone the instance. Only the databases and directories that are supposed to be cloned are affected. This object type works in conjunction with the OraApps Cloning Process workflow, described in "[OraApps Cloning Process Workflow](#)" on the previous page.

If you alter any of the default workflow step names in the OraApps 11i Cloning workflow, apply the correct modifications to the OraApps 11i Cloning object type command conditions, because the various commands execute based on the workflow step name.

Caution:

Cloning your Oracle instance is a complex process, and misconfigurations can easily result in unexpected consequences. This object type and its associated workflow provide a template for cloning your instances, but they require tailoring for your environment. Contact Micro Focus Software Support to facilitate a successful implementation and see the Micro Focus Software Support Web site:

<https://softwaresupport.hpe.com>

The OraApps 11i Cloning object type supports cloning for any Oracle E-Business Suite Release 11i instance that does not use AutoConfig, which is a configuration option for Oracle E-Business Suite Releases from 11.5.1 through 11.5.6.

For more information, see "[OraApps 11i Cloning Object Type](#)" on page 192.

ksc_oa_copy_clone_scripts Special Command

The `ksc_oa_copy_clone_scripts` command copies custom PPM Center versions of the Oracle cloning scripts to the destination server. The command is used by the OraApps 11i Cloning object type.

Managing Patches

This section includes the following topics:

- ["Overview of Managing Patches" below](#)
- ["Analyzing Patch Information" on the next page](#)
- ["Analyzing Patches" on page 133](#)
- ["Oracle E-Business Suite Implementation and Patching Scenarios" on page 139](#)
- ["Configuring Patch Management" on page 140](#)
- ["Maintaining PPM Center" on page 158](#)

Overview of Managing Patches

A significant number of patches are distributed by Oracle for each Oracle release, and the number per release continues to grow. For system stability, these patches must be applied to and verified in each Oracle E-Business Suite instance in use. Research is often required to determine prerequisite patching requirements or the impact of a given patch on custom-developed extensions and system functionality. Patches might be requested by different parts of the organization, and multiple patches might be in progress at a given time. Analyzing, applying, and tracking of patches across multiple Oracle E-Business Suite instances (and tiers) can pose a major management challenge. In addition, patches can require system downtime, so timely processing is important.

The Extension provides you with the tools you need for effective patch management. With minimal configuration, the workflows, object types, request types, and report types included in the Extension function together to manage common patching tasks. The Extension allows you to take control of your patch management functions by providing critical functionality such as the following:

- Automated management of ADPATCH and ADADMIN interactive sessions using a rule-based engine and notification-on-fail features
- Ability to analyze patch content details before or after patching, without reliance on scarce DBA resources
- Ability to merge patches for more efficient application
- Ability to patch single or multitier instances in serial or parallel
- Automated execution of post-patching activities, such as required ADADMIN functions, in conjunction with patch application
- Identification of bugs reported to Oracle and pending resolution
- Ability to track the patches received and the instances to which they are applied using processes that have been verified by Oracle customers

Analyzing Patch Information

Before applying a patch, it is common to analyze the patch contents to assess impact and potential risks. This analysis helps to determine when to apply the patch, where to apply it, whether functional analysis is required (for example, if the patch impacts functionality relied on by custom Extensions), what tests should be run to verify the patch, and which functional areas must approve applying it.

The file contents of a patch are listed in the patch's C-driver or U-driver file. For patches that contain a large number of changes or a large number of subpatches, deciphering the contents of the patch can take a considerable amount of time and manual effort, exacerbated by the fact that patch files might be controlled by a DBA team and not available directly to the person doing the analysis.

When bugs are found in a given Oracle instance, it might be necessary to identify whether specific existing patches have already been applied to the instance. The patches in question might be subpatches delivered in a master patch set. Quickly assessing whether a particular file or subpatch has been applied can save analysis time and help speed resolution of an Oracle TAR. Similarly, if a patch is a prerequisite for a bug fix, identifying that the patch has already been applied can save the effort of reapplying the patch.

The patch analysis feature of the Extension provides functionality that allows you to capture and analyze patch contents for any Oracle release supported by the Extension before or after patch application. Patch analysis provides an automated mechanism to record patch details during the patch application process, allowing this information to be leveraged during patch verification activities. Analyst users without configuration licenses for Deployment Management can load patches and run the analysis with a standard Deployment Management license through the PPM Dashboard.

A flexible mechanism is provided to allow users to report on patch data in a way that is meaningful to them. Patch analysis using the Extension consists of using the following entities, as described in the following sections:

- OraApps Patch Detail Report
- OraApps Patch Analysis Report
- OraApps Path Deployment workflow
- OraApps Patch Deployment Subworkflow
- OraApps Patch Data Capture Subworkflow

OraApps Patch Detail Report

The OraApps Patch Detail Report allows users to view patch contents, including files and subpatches, both before and after patches are applied to an environment. The format of the report is similar to the Micro Focus standard detail reports for entities such as object types. Users can locate patches based on patch contents (for example, files, bugs, and

environments) and can specify which data is displayed in the report. Users can load patch data prior to reporting or applying the patch, and they can delete patch data before applying the patch.

The C-driver or U-driver file of patches to be loaded are taken out of the standard patch repository, removing the need for users to physically access the patch archive, thereby reducing the load on DBAs to provide patch information to non-DBA users.

For more information about this report, see ["OraApps Patch Detail Report" on page 300](#).

OraApps Patch Analysis Report

The OraApps Patch Analysis Report allows users to view patch contents in a tabular format, which facilitates integrated data analysis. Users can locate patches based on patch contents or application status, and can specify the types of data displayed on the report and the display order. Using this report, users can quickly determine whether a given subpatch or file has been applied, and in which environments.

For more information about this report, see ["OraApps Patch Analysis Report" on page 297](#).

OraApps Patch Data Capture Subworkflow

The OraApps Patch Data Capture Subworkflow automates the capture of patch data during patch application. The subworkflow is invoked during each patch application, so that patch detail data is available when patches are validated for a given environment. The subworkflow verifies that the object type is Oracle Patch, enforces rules regarding when a patch is eligible for capture, captures patch details into the PPM Center patch repository, and associates patch details with the patched environment. The C-driver or U-driver file is taken from the patched Oracle E-Business Suite instance and used to determine patch details. For efficiency, once the patch has been successfully loaded for a given package line, the full patch is not analyzed if the patch already exists in the tables. A notification is issued if the capture of patch data fails for any reason. Logs are created to indicate whether the capture took place and to show the treatment of the bugs found in the patch.

When the PPM Center Environment Refresh functionality is used, patch detail data is automatically kept synchronized with environment contents information.

Patch details are captured for both standard and merged patches. In order to meaningfully report on merged patches, they should have been merged using the "merged_name" syntax, otherwise all merged patches are reported as a single patch.

For more information about this subworkflow, see ["OraApps Patch Deployment Workflow and Its Subworkflows" on page 153](#) and ["OraApps Patch Deployment Workflow and Its Subworkflows" on page 261](#).

For more information about merged patches, see ["OraApps Merge Patch Object Type" on page 151](#).

For descriptions of the various configuration processes involved in patch analysis, see ["Configuring Patch Management" on page 140](#).

Analyzing Patches

The following sections discuss the following subjects:

- Overview of applying patches
- Automating ADPATCH and ADADMIN
- Merging patches
- Reporting on patches

Overview of Applying Patches

The Extension automates Oracle's ADPATCH, ADADMIN, and ADMRGPCH utilities through the standard PPM Center Workflow and Execution engines. This allows you to apply and track patches across multiple Oracle instances using PPM Center workflows and object types.

Note:

The Extension is designed to apply only Oracle patches. The Extension cannot be used to apply Oracle database patches.

Once an Oracle patch has been identified and received from Oracle, the patch should be taken through a structured process of review and deployment to maintain visibility to the patch and reduce the risk of applying a bad patch to a critical environment. Review each patch for impact, pre- and post-patch application activities, and general relevance to your specific environment.

After the review process is complete, apply the patch to a "VANILLA" Oracle instance that does not include any in-house customization. This is a standard practice to isolate the Oracle patch and verify its correctness. If you find that the patch works in this environment and not in the other instances, then the problem might be due to customization or configuration. After deploying the patch successfully to the VANILLA instance, apply it to your development (DEV), testing (TEST), and production (PROD) environments, in turn, verifying success in each environment.

If the patch fails testing and verification in any of these instances, review it again. The purpose of re-review is to determine the problem with the patch, and if can be corrected, to move the patch through the full deployment cycle to all specified instances. If the patch cannot be fixed, cancel deployment.

The Extension provides the tools you need for effective patch management. With minimal configuration, PPM Center workflows, object types, and report types function together to automate common patching activities and processes as follows:

- Interactive ADPATCH and ADADMIN sessions are automatically initiated and managed using rule-based intelligence.
- Post-patch activities (such as performing ADADMIN functions) are included.
- Interested users are notified of patch failures, availability for verification, and rejection.
- Analysis of patch content is performed.

Note:

If you do not plan to maintain a central repository of patches, you must manually copy each patch to each of the target environments in which you plan to have Deployment Management apply it. You can do this in a controlled manner using a PPM Center workflow, or by modifying the patch object type to reference a source environment instead of Patch Stage (see Patch Stage Environment on page 194). The patch repository can also be NFS mounted to avoid having duplicate copies of the patch files in each instance.

Automating ADPATCH and ADADMIN

To apply patches in an Oracle environment using the ADPATCH and ADADMIN utilities:

1. Using Deployment Management, create a package.
2. Select the OraApps Patch Deployment workflow.

Note:

We recommend, but does not require, using the OraApps Patch Deployment workflow and the OraApps Patch Deployment Subworkflow. Object types related to patching can function with workflows you have created or configured at your site. However, capturing patch detail data requires the OraApps Patch Capture Subworkflow.

3. Add package lines and select the Oracle Patch object type.
The ADPATCH utility is invoked automatically by the object type.
The Oracle patch is copied from a central repository, and then unpacked in the Oracle instance where the patch will be applied. The ADPATCH prompts must be answered interactively, based on current field values and the setup of the environment being patched.
4. If your patch requires administration tasks, you can make these tasks occur automatically. Add a new package line and select one of the following object types to work in conjunction with the Oracle ADADMIN utility:
 - OraApps ADADMIN Compile APPS Schema object type
 - OraApps ADADMIN Compile Flexfields object type
 - OraApps ADADMIN Generate Jar Files object type
 - OraApps ADADMIN Generate Messages object type

For descriptions and configuration information about these object types, see **Configuring Object Types on page 203**.

5. Submit the package.

When the workflow executes, the Oracle ADADMIN utility is automatically invoked by the object type.

Limitations

The Extension is designed to run the ADPATCH and ADADMIN utilities, which utilize a standard prompt and response structure. This is usually sufficient to run a complete Oracle patch successfully. Unfortunately, Oracle does not have a guaranteed standard for the manner in which patches are generated and applied. Some patches require additional interaction other than simple questions and answers. The Extension does not support these more complex and unpredictable interactions.

For this reason, We strongly recommend that the first time you apply a patch, you apply it manually in your “sandbox” environment, after reading the README file provided with the patch. During this process, pay particular attention to any manual actions the patch requires beyond answering questions when prompted. If such manual actions are required for a patch, the Extension cannot be used to apply the patch. However, you can still generate a package line to track the history of the patch, and execution steps can be bypassed while the patch is manually applied.

Note:

While manually applying a patch to your sandbox environment, note the names of the driver files required to apply it. These names will be required later when you create a package line in Deployment Management for the patch to be automatically applied to other environments.

Execution Logs

For each environment to which you apply a patch, a log file of the ADPATCH output is produced. These log files typically reside local to the Oracle instance to which the patch was applied. This makes it very difficult to audit results across environments. Additionally, there are no records of pre- or post-application steps that can be associated with the patch.

The Extension solves this by combining all ADPATCH interactions into a single color-coded HTML log file that is accessible from a Web browser through the standard package line execution logs. The PPM Center execution log includes pre- and post-application step records, and links to associated information (including the README file for that particular patch).

For more information about the “OraApps ADADMIN” object types, see the following sections:

OraApps ADADMIN Compile APPS Schema Object Type on page 245

OraApps ADADMIN Compile Flexfields Object Type on page 248

OraApps ADADMIN Generate Jar Files Object Type on page 250

OraApps ADADMIN Generate Messages Object Type on page 252

Merging Patches

The OraApps Merge Patch object type automates the Oracle ADMRGPCH utility, allowing users to merge multiple compatible Oracle E-Business Suite Release 11i or Release 12 patches into a single patch. This decreases the number of individual patches to be tracked and applied and can decrease the total time dedicated to tracking and applying patches.

The OraApps Merge Patch object type takes specified patch archive files from the Patch Stage repository, merges them into a single patch in an Oracle E-Business Suite Release 11i or Release 12 instance using ADMRGPCH, and then returns the merged patch archive to the Patch Stage repository.

The OraApps Merge Patch object type produces a new patch archive with a user-specified name that you can use with Oracle patch-related object types to apply the patch. Details of the patch are captured by the patch analysis functionality in the Extension.

To merge patches in an Oracle environment that uses ADMRGPCH:

1. Place the patch archives to be merged in the Patch Stage area.
2. In Deployment Management, create a package.
3. Add a package line to the patch that uses the OraApps Merge Patch object type.
4. Select the workflow defined to merge patches. See ["Creating a Workflow to Merge Patches" on page 156](#).
5. Submit the package.

This procedure merges the individual patches into a single patch. You can then process the merged patch using the OraApps Patch Deployment workflow.

Caution:

ADMRGPCH does not merge patches of different releases, different platforms, or different parallel modes.

ADMRGPCH does not consider any dependencies regarding the sequence in which patches should be applied. You must be extremely careful in merging patches if there are sequence dependencies.

In Release 11i, Oracle added support for a new type of patch driver, the U-driver. Earlier versions of ADMRGPCH merge patches but do not merge the U-driver information, which can result in a potentially destructive merged patch. If you must merge patches that include U-drivers, you must use an instance that is at Release 11.5.9 or Release 11i AD.H patch level or later.

In Release 12, ADMRGPCH merges only patches that include U-drivers. Accordingly, the OraApps Merge Patch object type checks that all Release 12 patches to be merged include U-drivers.

Note:

The name you choose for the patch archive is also the name used for the patch itself. The name can contain no special characters except for the underscore (_).

For more information about the object type discussed in this section, see ["OraApps Merge Patch Object Type" on page 181](#).

For more information about the workflow discussed in this section, see ["OraApps Patch Deployment Workflow and Its Subworkflows" on page 261](#).

Reports About Patches

After applying all the patches to your Oracle environment, you can run reports (available through the Extension) that give you information about the impact these patches have had on your Oracle system. The following reports, as described in the following sections, provide specialized patch information that can help you in your post-patch analysis:

- OraApps Patch Detail Report
- OraApps Patch Analysis Report
- Patch Application Comparison Report
- Patches Applied to an Environment Report
- Pending Patches Report
- OraApps Patch Matrix Report
- OraApps Patches Applied for Specific Bug Report

OraApps Patch Detail Report

The OraApps Patch Detail Report provides details about selected patches. This report is also used to load patch data for unapplied patches. Users control the type of information displayed and the patches selected. If fields are selected to load patch data, the report type calls a special command to load the data before executing the report. If required, the report calls another special command to remove the patch data after report completion.

For more information about this report, see ["OraApps Patch Detail Report" on page 300](#).

OraApps Patch Analysis Report

The OraApps Patch Analysis Report is similar to the OraApps Patch Detail Report, except that the OraApps Patch Analysis Report does not allow users to load in patch data. Also, rather than providing information in a master/detail format, it consolidates all the information into a single table. This allows you to bring the information into spreadsheet programs such as Microsoft® Excel for additional reporting. You can select a variety of filter criteria, and you can configure which information to show in the table and in what order. For example, you might want to view only patch headers and included bug information, or just bug and file information.

For more information about this report, see ["OraApps Patch Analysis Report" on page 297](#).

For information about configuring this report, see ["OraApps Patch Analysis Report" on page 156](#).

Patch Application Comparison Report

The Patch Application Comparison Report is specialized for Oracle E-Business Suite patches. This report can be used to obtain a list of patches migrated to or applied to up to five different environments. The report lists each patch number and the last migration date for the patch into each environment, allowing you to determine whether a patch was applied out of order or was missed in one of the environments.

For more information about this report, see ["Patch Application Comparison Report" on page 308](#).

Patches Applied to an Environment Report

The Patches Applied to an Environment Report lists Oracle E-Business Suite patches migrated to or applied to a specific environment. You can use this report to view the history of your Deployment Management executions for these patches. You can determine whether an Oracle Application patch has been applied multiple times to the same environment. You can also use this report to obtain a list of all the patches applied in a specific date range or all the patches run after a specific patch was applied.

For more information about this report, see ["Patches Applied to an Environment Report" on page 310](#).

Pending Patches Report

The Pending Patches Report is specialized for Oracle E-Business Suite patches. This report lists all the package lines for Oracle patches that are waiting at a migration step (that is, the lines have been approved for a given environment and now the patch must be applied to it).

For more information about this report, see ["Pending Patches Report" on page 312](#).

OraApps Patch Matrix Report

The OraApps Patch Matrix Report generates an annotated matrix of patches and environments, identifying the environments that have had patches applied. If a patch has been applied to an instance, the report provides a link to the package that last applied the patch. If a patch has not been applied to an instance, the report provides links to all packages that could apply the patch.

For more information about this report, see ["OraApps Patch Matrix Report" on page 303](#).

OraApps Patches Applied for Specific Bug Report

The OraApps Patches Applied for Specific Bug Report lists the patches that have been applied to resolve the bug that a user specifies.

For more information about this report, see ["OraApps Patches Applied for Specific Bug Report" on page 306](#).

Oracle E-Business Suite Implementation and Patching Scenarios

A number of implementation architectures are supported by Oracle. Each is discussed briefly in this section to provide context for the patch application process in the Extension. The following architectures are considered here:

- Release 11

The Extension supports both server-side and middle-tier standard patches on UNIX® and Windows platforms.

Normally all three drivers (C-, D-, and G-) are to be applied at the server. Only `C<bugnumber>.drv` and `G<bugnumber>.drv` should be applied to the middle tier. Do not apply `D<bugnumber>.drv` or any database driver file to the middle tier.

If the server and middle tier share the same code tree, then do not apply any driver files to the middle tier.

- Release 11i and Release 12

With Release 11i or Release 12, for the first time on Windows, Oracle implements relinking of executables during patching. To allow this relinking, Oracle requires the installation of additional third-party software tools for C++ compilation and UNIX shell emulation. In order for PPM Center to function correctly for Windows, you must make sure that your Oracle environment variables (for example, `APPL_TOP`) are visible within a Bourne-style shell. If all settings are not immediately available, you might need to customize the patch-related object type to set the necessary values. This includes `PATH` settings, both for Oracle directories and third-party tools such as MKS Toolkit. If the host uses multiple UNIX emulators, the MKS Toolkit shell must be used for patch application.

The Extension supports both server-side and middle-tier standard patches on UNIX and Windows platforms.

Using a patchset to Release 11i or Release 12, Oracle added support for a new type of patch driver—the U- (Unified) driver, which combines the actions of the C- (Copy), D- (Database), and G- (Generate) drivers. The Extension supports the use of U-drivers when applied in accordance with the guidelines in the *11.5.9 Maintenance Procedures Guide* document from Oracle. Generally this requires applying the U-driver first to the admin server node, then to any other nodes defined for the instance.

Oracle's ADPATCH utility determines whether a given driver can be applied to a given physical server node.

Configuring Patch Management

The following sections discuss configuring the following for Oracle E-Business Suite patch management:

- System requirements
- OA - Oracle Patch Admin security group
- Environments
- Object types
- Workflows
- Report types
- Special commands

System Requirements

You should establish a patch repository where patch archives for all patches will reside, regardless of Oracle release. This repository is represented by the Patch Stage environment in PPM Center. A patch repository is used as the source of patch archives for pre-patching detail loads and for patch application activities.

Considerations for creating the patch repository include the following:

- The patch repository should be accessible by a telnet, an ssh, or a bash client and must allow file transfer and directory creation.
- The repository should have adequate available space to extract the C-driver for a given patch. This occurs during pre-patching load of patch detail data.
- The server on which the repository is located must have an unzip utility installed that supports the format of the Oracle patch archives located on the server.

Each Oracle instance that will be accessed for patch-related activities must meet the following requirements:

- A location where patches are unpacked on the Oracle instance must be accessible by a telnet, an ssh, or a bash client and must allow file transfer.

- The patch must be unpacked in the patching area before the ADPATCH utility is called and the patch detail data is captured during patching. By default, the patch-related object type moves the archive from the patch repository and unpacks the archive in this directory (as specified by the value of PATCH_TOP in the environment definition for the instance).
- Operating system sessions must be able to run a configuration file (for example, \$APPL_TOP) to set an environment context for the Oracle instance. For details, see the following note.

Note:

The Oracle Patch object type expects to call an environment context file to set the Oracle context. By default, for UNIX environments the object type assumes that the file name is the Oracle SID name.

For Windows environments, to create an environment context file that is compatible with the bash shell, you can use the kacc_oa_windows_config.sh script (located in the <PPM_Home>/scripts directory, where <PPM_Home> represents the path where your PPM Center instance is installed). You must modify the appropriate object type commands so they call this file to set the environment context information for the destination instance before the object type invokes ADPATCH or ADADMIN.

Configuring the OA -Oracle Patch Admin Security Group

The OA - Oracle Patch Admin security group is used by default in the OraApps Patch Deployment workflow and the OraApps Patch Data Capture Subworkflow. Normally, users in this group are DBAs responsible for patching and instance administration.

If this group is appropriate to your organization, you should add access grants for it. Usually these include standard Demand Management and Deployment Management functions, plus environments, workflows, environment refreshes, and releases. You should also add view grants for environments, workflows, environment refreshes, and releases.

Configuring Environments

The following sections discuss configuring the following environments:

- Patch Stage environment
- Patch destination environment
- SERVER_ENV_NAME environment

Patch Stage Environment

The patch management functionality in the Extension assumes that all patches reside in a patch repository represented by the Patch Stage environment, and this environment is distinct from all Oracle environments that require patching. The environment exists solely as a repository from which patches are copied and later applied to target environments, or where C-drivers or U-drivers are extracted from existing patch archives to allow capture of patch details before patching. The patch repository is not release-specific.

The patch repository is used during patching, for loading of patch details before patching, and when patches are merged.

To create a Patch Stage environment:

1. Log on to PPM.
2. From the menu bar, select **Open > Administration > Open Workbench**.
The PPM Workbench opens.
3. From the shortcut bar, select **Environments > Environments**. The Environment Workbench opens.
4. Click **New Environment**.
The Environment window opens to the **Host** tab by default.
5. In the Environment Workbench, create an environment named Patch Stage.

Note:

It is important that this environment be named Patch Stage, because this is the exact environment name that the object type looks for when copying the patch to the target environment. Exact spelling and case are important. You can change the standard object types if you want to follow a different naming convention.

6. Complete the fields you must specify for the Host tab in the Environment window for the Patch Stage environment, as shown in Table 9-1.
An example Host tab for the Patch Stage environment is shown in "[Figure 9-1. Sample Patch Stage environment, Host tab](#)" on page 144 .
7. Click the **Extension Data** tab and then, at the bottom of the Environment window, click the **Oracle Applications** subtab.
8. Complete the fields you must specify for the **Extension Data** tab, **Oracle Applications** subtab, as shown in "[Table 9-1. Environment window required fields for Patch Stage environment](#)".

Table 9-1. Environment window required fields for Patch Stage environment

Field Name	Value or Description
Environment Name	Patch Stage
Host tab, Server section	
Enable Server	Select this check box so that fields in the Server section can be configured.
Name	Host name of the server to be used as a server-side patch repository.
Username	Valid Username that has access to the patch repository. This user must have access to an unzip utility that supports the Oracle patch archives located on the server, and must also have access to create directories and files within the repository directory.
Password	Password for the Username. Base Path Path to the directory where Oracle Server patches are located.
Host tab, Client section	
Enable Client	Select this check box so that fields in the Client section can be configured.
Name	Host name of the client (or middle tier) to be used as a middle-tier patch repository.
Username	A valid Username that has access to the patch repository.
Password	Password for the Username.
Base Path	Path to the directory where Oracle middle-tier patches are located.
Extension Data tab, Oracle Applications subtab	
Enabled	Select the Yes option to make the fields on this subtab available to be configured.
Extension Data tab, Oracle Applications subtab, Application Server (Server Environment) section	
Server Patch Top	Path to the directory where Oracle patches are located. This is used during patch merge.

Note:

The directory you specify in the **Base Path** field for the server and client should be the directory where all the patch archive files are placed. Typically these are compressed files named <file name/number>.Z OR <file name/number>.zip from which the patch

files are extracted. Windows users should configure a virtual FTP server directory for the base path if one does not already exist.

Figure 9-1. Sample Patch Stage environment, Host tab

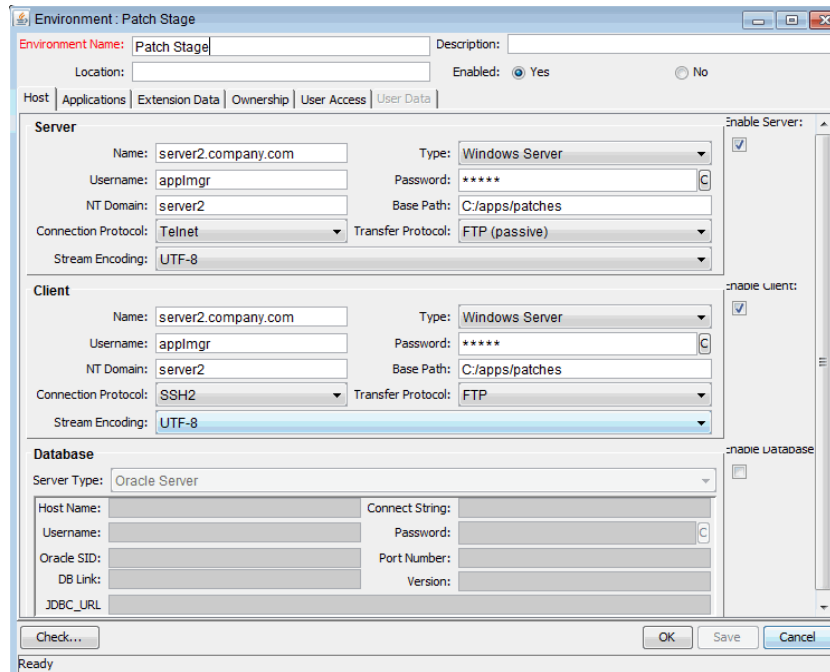


Figure 9-2. Sample Patch Stage environment, Extension Data tab, Oracle Applications subtab

Note:

If you do not plan to maintain a central repository of patches, you must manually copy each patch to each of the target environments in which you plan to have Deployment Management apply it. You can do this in a controlled manner using a PPM Center workflow, or by modifying the patch object type to reference a source environment instead of Patch Stage. The patch repository can also be NFS mounted to avoid having duplicate copies of the patch files in each instance.

Patch Destination Environment

Before you use the Extension to apply any patches, you must configure each destination environment by completing its Host tab and the Extension Data tab, Oracle Applications subtab, as follows:

1. In the Environment Workbench, select the destination environment.
2. Complete the required fields in the Host tab in the Environment window for the patch destination environment, as shown in "[Table 9-2. Environment window required fields for patch destination environment](#)".

3. Click the Extension Data tab and then the Oracle Applications subtab at the bottom of the Environment window.
4. Complete the fields you must specify for the Extension Data tab, Oracle Applications subtab, as shown in "[Table 9-2. Environment window required fields for patch destination environment](#)".

An example Extension Data tab, Oracle Applications subtab for an Oracle E-Business Suite Release 11 environment is shown in "[Figure 9-3. Sample patch destination environment, Extension Data tab, Oracle Applications subtab](#)".

Note:

The server and client user must have read/write permissions for the PATCH_TOP directory.

Table 9-2. Environment window required fields for patch destination environment

Field Name	Value or Description
Host tab, Server section	
Enable Server	Select this check box so that fields in the Server section can be configured.
Name	Host name of the server to be used as a server-side patch repository.
Type	Type of server.
Username	Required during initial copy of the patch file for backend patches.
Password	Required during initial copy of the patch file.
Base Path	Required during initial copy of the patch file.
Connection Protocol	Connection protocol to use.
Transfer Protocol	Transfer protocol to use.
Host tab, Client section	
Enable Client	Select this check box so that fields in the Client section can be configured.
Name	Host name of the client (or middle tier) to which middle-tier patches will be applied. This information is used when forms-type patches are applied.
Username	Required during initial copy of the patch file.

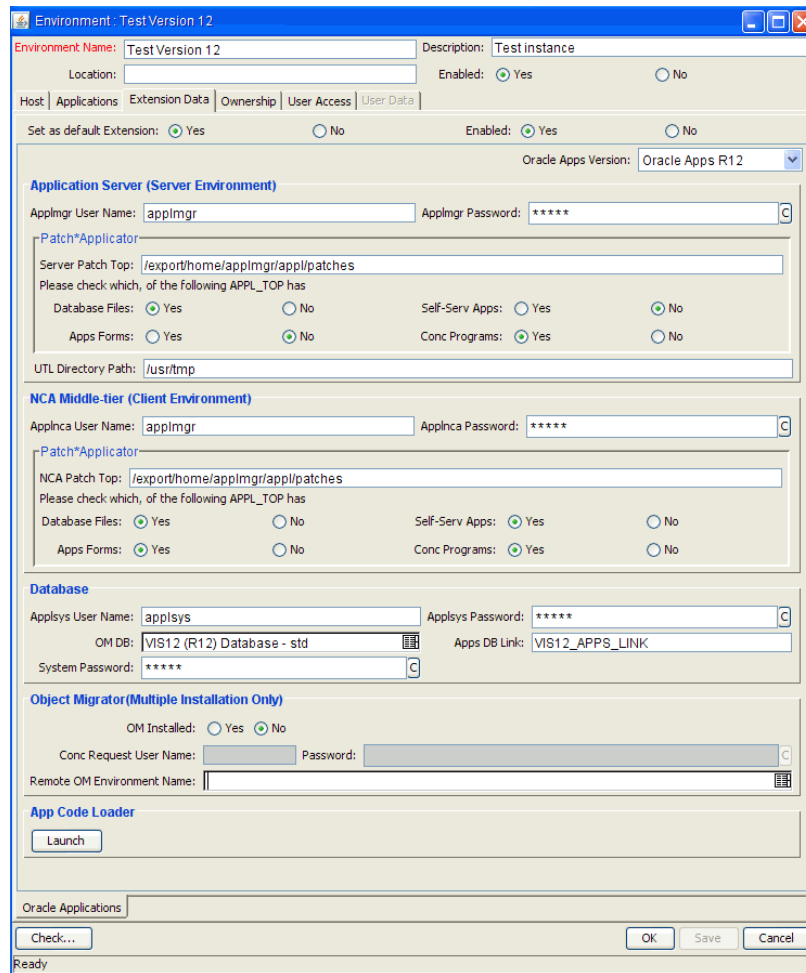
Table 9-2. Environment window required fields for patch destination environment, continued

Field Name	Value or Description
Password	Required during initial copy of the patch file for middle-tier patches.
Base Path	Required during initial copy of the patch file.
Extension Data tab, Oracle Applications subtab	
Enabled	Select the Yes option to make the fields on this subtab available to be configured.
Oracle Apps Version	Version of Oracle E-Business Suite running in this environment—Oracle Apps R11 (for Oracle E-Business Suite instances at Release 11 or Release 11i), or Oracle Apps R12.
Extension Data tab, Oracle Applications subtab, Application Server (Server Environment) section	
Applmgr User Name	Owner of the Oracle file system objects on the server computer, and the user that executes the AD utilities. This is normally APPLMGR.
Applmgr Password	Password for APPLMGR.
Server Patch Top	Location where all server-side patches being applied to the environment are copied and extracted from their archive formats. For example: APPL_TOP/Patches or /u1/apps/prod.
Database Files	Whether or not the files for installing or upgrading the database are installed in the APPL_TOP area of the server host. This value is used during execution of the ADPATCH and ADADMIN utilities on Release 11, and to determine whether patch detail information will be captured for a given Release 11, Release 11i, or Release 12 patch. Patch detail information is captured only for database tiers.
Self-Serv Apps	Whether or not the Java™ and HTML files for self-service applications will be installed in the APPL_TOP area of the server host.
Apps Forms	Whether or not the Oracle Forms files are installed in the APPL_TOP area of the server host.
Conc Programs	Whether or not the concurrent programs files are installed in the APPL_TOP area of the server host.

Table 9-2. Environment window required fields for patch destination environment, continued

Field Name	Value or Description
Extension Data tab, Oracle Applications subtab, NCA Middle-tier (Client Environment) section	
Applmgr User Name	Owner of the Oracle file system objects on the server computer, and the user that executes the AD utilities. This is normally APPLMGR.
Applmgr Password	Password for APPLMGR.
Server Patch Top	Location where all server-side patches being applied to the environment are copied and extracted from their archive formats. For example: APPL_TOP/Patches or /u1/apps/prod.
Database Files	Whether or not the files for installing or upgrading the database are installed in the APPL_TOP area of the server host. This value is used during execution of the ADPATCH and ADADMIN utilities on Release 11, and to determine whether patch detail information will be captured for a given Release 11, Release 11i, or Release 12 patch. Patch detail information is captured only for database tiers.
Self-Serv Apps	Whether or not the Java™ and HTML files for self-service applications will be installed in the APPL_TOP area of the server host.
Apps Forms	Whether or not the Oracle Forms files are installed in the APPL_TOP area of the server host.
Conc Programs	Whether or not the concurrent programs files are installed in the APPL_TOP area of the server host.
Extension Data tab, Oracle Applications subtab, Database section	
Applsys User Name	Database owner of the Oracle Applications Object Library (AOL). This is normally applsys.
Applsys Password	Password for applsys.
System Password	Password for the database system user.
Apps DB Link	The database link defined in the PPM Center schema that connects to the APPS account of the Oracle instance. This is used in rare cases during patch detail data capture after patch application. This might already be defined if you are using AOL object types to integrate with Object Migrator.

Figure 9-3. Sample patch destination environment, Extension Data tab, Oracle Applications subtab



SERVER_ENV_NAME Environment

You must configure an environment whose name matches the `SERVER_ENV_NAME` parameter in the `server.conf` file of PPM. This environment represents the PPM Center instance, and it is used during certain FTP and connection functions.

On the Host tab, the Username should be the user who owns the PPM Center instance. The Base Path should be the location of the PPM Center instance (`<PPM_Home>`), where `<PPM_Home>` represents the path where your PPM instance is installed.

Configuring Object Types

The following sections discuss configuring the following object types:

- Oracle Patch object type
- OraApps Merge Patch object type
- Object types that automate the use of the ADADMIN utility

Caution:

In Release 11i, Oracle added support for a new type of patch driver, the U-driver. Earlier versions of ADMRGPCH merge patches but do not merge the U-driver information, which can result in a potentially destructive merged patch. If you must merge patches that include U-drivers, you must use an instance that is at Release 11.5.9, Release 11i AD.H patch level or later, or Release 12.

Oracle Patch Object Type

The Oracle Patch object type is used for applying and tracking patches across multiple Oracle instances for any Oracle release supported by the Extension.

This object type obtains the patch file from a central repository and then copies and unpacks it in the Oracle environment that must be patched. The Oracle Patch object type invokes the Oracle ADPATCH utility to run the patch. The object type responds to ADPATCH prompts dynamically, eliminating the need for the user to answer the prompts interactively. A log of the interactive session is captured and remains available in PPM Center.

For Oracle E-Business Suite Release 11.5.10 or later (including Release 12), AutoConfig can be called automatically from ADPATCH to instantiate and run new templates contained in a patch. However, AutoConfig can modify environment files. If the patched instance runs on Windows, you might need to modify the environment context file that the Oracle Patch object type uses. If AutoConfig is run automatically during patch application, the object type adds the following warning to the Package Execution Log: "AutoConfig was executed by this patch. Environment files for Windows may need to be modified." To create an environment context file for the Oracle E-Business Suite instances running on Windows servers, see the note after ["If your Oracle instances reside on Windows, add logic to call an environment context file to set the required environment variables at the bash level. For details, see the following note."](#)

To configure the Oracle Patch object type:

1. Configure the object type for any special processing required at your site. For example, you might want to execute a site-specific script that gathers database statistics before patching.
2. Review the default field values and display settings, and adjust them to meet your needs. For example, if your Oracle environments are multilingual, you might want to display the Apply Multi-Lingual Patch Now field or change its default value.

3. If your Oracle instances reside on Windows, add logic to call an environment context file to set the required environment variables at the bash level. For details, see the following note.

Note:

The Oracle Patch object type expects to call an environment context file to set the Oracle context. By default, for UNIX environments the object type assumes that the file name is the Oracle SID name.

For Windows environments, to create an environment context file that is compatible with the bash shell, you can use the `kacc_oa_windows_config.sh` script (located in the `<PPM_Home>/scripts` directory, where `<PPM_Home>` represents the path where your PPM Center instance is installed). You must modify the appropriate object type commands so they call this file to set the environment context information for the destination instance before the object type invokes `ADPATCH` or `ADADMIN`.

4. To capture auditing information in greater detail, consider making the Patch Drivers field the “Object Revision” for the object type.

Note:

Review any special processing added to the Oracle Patch object type to determine whether it should also be applied to the OA -Capture Patch Workflow step source. See ["OA -Capture Patch Workflow Step Source" on page 155](#).

For more information about this object type, see ["Oracle Patch Object Type" on page 178](#).

OraApps Merge Patch Object Type

The OraApps Merge Patch object type allows users to merge multiple compatible Oracle E-Business Suite Release 11i or Release 12 patches into a single patch using Oracle’s standard `ADMRGPCH` utility. This decreases the number of individual patches to be tracked and applied, and it can decrease the total time dedicated to tracking and applying patches.

For important introductory information and limitations, see [Merging Patches on page 185](#).

The OraApps Merge Patch object type executes a file to set the required Oracle environment information. By default the object type expects the file name to be the Oracle SID as defined for the environment. You must amend the object type if the default mechanism is not appropriate.

For more information about this object type, see ["OraApps Merge Patch Object Type" on page 181](#).

Object Types that Automate Use of the ADADMIN Utility

The Extension provides a number of object types that automate the use of the ADADMIN utility for Oracle E-Business Suite releases supported by the Extension. ["Table 9-3. OraApps ADADMIN-related object types"](#) lists the OraApps ADADMIN-related object types provided with the Extension. Review the default field values and display settings for these object types and adjust them to your needs.

Table 9-3. OraApps ADADMIN-related object types

Object Type	Description
OraApps ADADMIN Compile APPS Schema	Recompiles one or more APPS schemas in an Oracle E-Business Suite instance, using the ADADMIN utility, and dynamically responds to ADADMIN prompts based on environment and package line information. The object type then navigates the ADADMIN menus, selecting the correct options to compile the APPS schemas.
OraApps ADADMIN Compile Flexfields	Recompiles all flexfields defined in an Oracle E-Business Suite instance, using the ADADMIN utility. This function might be required after patching or other changes to make sure the flexfield views are accurate.
OraApps ADADMIN Generate Jar Files	Generates either out of date or all (FORCE option) jar files in an Oracle E-Business Suite instance, using the ADADMIN utility. This task can be run at any time, but is usually run after applying forms patches. Use of this object type is supported for Oracle Release 11i and Release 12, but not for Release 11.
OraApps ADADMIN Generate Messages	Generates binary message files for some or all applications in an Oracle E-Business Suite instance, using the ADADMIN utility. This task is usually run after applying a patch that requests message regeneration. Oracle uses the message files to display online messages to users. The option to generate messages for specific applications and languages is not available in all versions of ADADMIN.

Note:

The Oracle Patch object type expects to call an environment context file to set the Oracle context. By default, for UNIX environments the object type assumes that the file name is the Oracle SID name.

For Windows environments, to create an environment context file that is compatible with the bash shell, you can use the `kacc_oa_windows_config.sh` script (located in the `<PPM_Home>/scripts` directory, where `<PPM_Home>` represents the path where your PPM Center instance is installed). You must modify the appropriate object type commands so they call this file to set the environment context information for the destination instance before the object type invokes `ADPATCH` or `ADADMIN`.

For more information about these object types, see the following sections:

- ["OraApps ADADMIN Compile APPS Schema Object Type" on page 184](#)
- ["OraApps ADADMIN Compile Flexfields Object Type" on page 186](#)
- ["OraApps ADADMIN Generate Jar Files Object Type" on page 188](#)
- ["OraApps ADADMIN Generate Messages Object Type" on page 190](#)

Configuring Workflows

The following sections discuss the following workflow configuration activities:

- OraApps Patch Deployment workflow and its OraApps Patch Deployment Subworkflow and OraApps Patch Data Capture Subworkflow
- OA - Capture Patch workflow execution step source
- Creating a workflow to merge patches

OraApps Patch Deployment Workflow and Its Subworkflows

The OraApps Patch Deployment workflow takes a patch through a proven process of review, application, detail data capture, and verification. If the patch successfully completes these steps, it goes through the full deployment cycle to all specified instances.

The workflow calls the OraApps Patch Deployment Subworkflow to ensure that the Oracle instance is in maintenance mode if the instance is at Release

11.5.10 or later, and to apply the patch (see ["OraApps Patch Deployment Subworkflow" on the next page](#)). The workflow calls the OraApps Patch Data Capture Subworkflow to capture the details of the patch as the patch is applied to an Oracle instance (to configure this subworkflow, see ["OraApps Patch Data Capture Subworkflow" on page 155](#)).

To configure the OraApps Patch Deployment workflow:

1. Verify security settings and modify them, as required.
By default, the Assigned To User, Assigned To Group, Created by User, and OA - Oracle Patch Admin security groups can act on the workflow steps.
The calls to the subworkflows called by this workflow do not include or require security by default, but you can configure authorized security groups.
2. Modify the workflow to reflect the actual environments in use. For each environment you add, you must duplicate the logic of the Apply to <environment> step, the Capture Patch (<environment>) step, and the Patch Verification (<environment>) step.
3. Add destination environment values to the Apply to <environment> step and the Capture Patch (<environment>) step. These steps should occur in a pair for each targeted environment and should have exactly the same destination environment settings.
4. The workflow includes Micro Focus-supplied notifications that inform users when they must take action. Customize these notifications as required. You might want to make the notifications reflect the destination environment rather than the generic DEV/TEST/PROD references, or you might want to change the recipients.
5. Make sure that the Oracle Patch object type is the only object type that has access to this workflow.

Note:

For each migration step in the OraApps Patch Deployment workflow, the destination environment is the target environment to which the patch will be deployed. For each target environment, the patch archive is always retrieved from the patch repository (Patch Stage environment) and copied to that environment.

For more information about the OraApps Patch Deployment workflow, see "[OraApps Patch Deployment Workflow and Its Subworkflows](#)" on page 261.

OraApps Patch Deployment Subworkflow

The OraApps Patch Deployment Subworkflow is called by the OraApps Patch Deployment workflow to apply the patch when the user acts on the subworkflow's Apply Patch step.

Note:

Environments are specified in the parent OraApps Patch Deployment workflow, not in this subworkflow.

If the Oracle E-Business Suite destination instance is at Release 11.5.10 or later (including Release 12), it must be in maintenance mode before the patch can be applied. In this case, the subworkflow determines the mode of the destination instance, sets the instance to maintenance mode, applies the patch, and resets the instance to its original mode. For releases earlier than 11.5.10, commands regarding the mode of the instance are not executed.

For more information about the OraApps Patch Deployment Subworkflow, see ["OraApps Patch Deployment Workflow and Its Subworkflows" on page 261](#).

OraApps Patch Data Capture Subworkflow

The OraApps Patch Data Capture Subworkflow is called by the OraApps Patch Deployment workflow. This subworkflow captures the details of the patch being applied to a given instance, and associates the patch with the environment. The detailed patch data captured by the subworkflow is taken only when the Oracle Patch object type is in use, by default.

Note:

Environments are specified in the parent OraApps Patch Deployment workflow, not in this subworkflow.

To configure the OraApps Patch Data Capture Subworkflow:

1. Verify security settings.
2. Add security on the Record Patch (IMMED) step, if required.
3. By default, the Assigned To User, Assigned To Group, Created by User, and OA - Oracle Patch Admin security groups can act on the workflow steps.
4. If patching object types do not use the standard name (Oracle Patch), add the names of Release 11, Release 11i, or Release 12 object types to the Check Object Type step. Object types listed here will have patch detail data captured if additional conditions are met, whereas other object types will not.
5. If special handling is required, copy and modify the OA - Capture Patch Workflow step source (described in ["OA -Capture Patch Workflow Step Source" below](#)). If a copy is made, the workflow step must be removed and re-added to point to the target step source. Changes might include special logic to locate the patch, or special logic to identify when to capture patch details (for example, capturing data for every tier). Make sure that any additional setups needed for the specialized logic are completed.
6. The Record Patch (IMMED) step sends a notification to the OA - Oracle Patch Admin security group if the step fails. Modify the recipients to meet your needs. Modify the notification text or conditions as required.

For more information about the OraApps Patch Data Capture Subworkflow, see ["OraApps Patch Data Capture Subworkflow" on page 132](#) and ["OraApps Patch Deployment Workflow and Its Subworkflows" on page 261](#).

OA -Capture Patch Workflow Step Source

Any special processing added to the Oracle Patch object type should be evaluated and applied to the OA - Capture Patch workflow (execution) step source, as appropriate. For example, the C-driver or U-driver is expected to reside in a specific location. If the object type is customized to use a different location, the step source should be similarly changed.

The step source verifies that the following criteria are met before attempting to capture patch detail data:

- The type of patch is Backend.
- The environment is Release 11, Release 11i, or Release 12.
- The environment is a DB tier.
- The patch is being executed.
- The package line includes a C-driver or U-driver.

Although these criteria fit most needs, the step source can be modified to reflect different criteria, if required. For example, an organization might want to capture patch data for all tiers, not just the database tier.

Creating a Workflow to Merge Patches

If you want to use the OraApps Merge Patch object type to merge multiple Release 11i or Release 12 patches, you must create a new workflow.

The workflow should include a standard execution step (for example, DLV Execution with Reset). In this step, the source environment should be the Patch Stage environment, and the destination environment should be the Oracle environment where you want to run ADMRGPCH to merge the patch files.

Add appropriate security to the workflow. We recommend that you limit the workflow to using only the OraApps Merge Patch object type.

For more information about merging patches, see ["Merging Patches" on page 136](#).

Configuring Report Types

The following sections discuss configuring the following report types:

- OraApps Patch Analysis Report
- OraApps Patch Detail Report

OraApps Patch Analysis Report

The OraApps Patch Analysis Report allows users to view patch contents in a tabular format, which facilitates integrated data analysis. Users can locate patches based on patch contents or application status, and can specify the types of data displayed on the report and the display order. Using this report, users can quickly determine whether a given subpatch or file has been applied, and in which environments.

When you configure an OraApps Patch Analysis Report, consider that the timeouts on the command steps reflect average-sized patch needs. Increase the timeouts as appropriate to your site and patch analysis style. Of course, larger patches and broader reporting require more time.

Note:

Take special care when selecting running fields for the OraApps Patch Analysis Report. A report that includes thousands of records (for example, all files and bugs in a maintenance pack) drains database resources and produces a report of questionable usefulness.

For more information about this report, see ["OraApps Patch Analysis Report" on page 138](#) and ["OraApps Patch Analysis Report" on page 297](#).

OraApps Patch Detail Report

The OraApps Patch Detail Report lists details of selected patches.

When you configure an OraApps Patch Detail Report, consider the following:

- The timeouts on the command steps reflect average-sized patch needs. Increase the timeouts, as appropriate for the customer site and patch analysis style. For example, if MegaPatches, FamilyPacks, and MaintenancePacks are commonly analyzed before application, the timeout for the load and purge steps might need to be increased. The timeout for the report itself probably would need to be increased also.
- If two distinct user groups are expected to use the Patch Detail Report, it might be worthwhile to make a second copy of the report type and hide the load-related fields from users who would never load patches for analysis.

We recommend that file details of maintenance patches never be displayed in this report, because there can be tens of thousands of files in a given patch.

For more information about this report, see ["OraApps Patch Detail Report" on page 300](#).

Special Commands

The following sections discuss the following special commands:

- `ksc_oa_capture_patch`
- `ksc_oa_purge_temp_patch`

`ksc_oa_capture_patch`

The `ksc_oa_capture_patch` command extracts the C-driver or U-driver from the patch archive to the PPM Server, validates the C-driver or U-driver, and calls a program to load detailed patch data based on the contents. The command is used by the OraApps Patch Detail Report when the report's Load Data option is set to Yes.

`ksc_oa_purge_temp_patch`

The `ksc_oa_purge_temp_patch` command removes patch detail data for the loaded patch if the patch has been applied in any environments. The command is used by the

OraApps Patch Detail Report when the report's Remove Data after Run option is set to Yes.

Maintaining PPM Center

The following sections discuss the following subjects:

- Maintaining the PPM Server
- Maintaining the PPM Center database

Maintaining the PPM Server

Some patch-related considerations for maintaining the PPM Server are as follows:

- Patch data can take up considerable space. For example, the C-driver file for the Release 11.5.7 Maintenance Pack is 42 megabytes. Make sure there is adequate space on the PPM Server to save these files during processing. In addition, with large patch files, the log file for processing and the report output for detail listings can also be quite large. The PPM Server needs adequate space to hold these files.
- Be sure to regularly monitor the space available for temporary processing, as well as for report and log storage.
- Deleting reports that are no longer required conserves disk space, especially when a patch was loaded for reporting, since the load log file might also be large.

Maintaining the PPM Center Database

Some patch-related considerations for maintaining the PPM Center database are as follows:

- If users are commonly loading patches for analysis before application and then purging the data, the patch tables can easily become fragmented. Monitor regularly for fragmentation problems, and take the actions required to periodically clean up tables and indexes.
- Patch data can optionally be stored in a separate tablespace from other PPM Center objects, and We strongly recommend doing so. Creating separate tablespaces reduces the likelihood that space constraints will affect other processing in PPM Center and reduces system resource contention when processing patch and other data.
- If PPM Center is running with a database that uses cost-based optimization, make sure that the patch-related database objects are analyzed regularly, especially after large data loads or purges.

Appendix A: Object Types

This section provides reference information about the Oracle-specific object types provided in the Extension. The object types are listed and defined in "[Table A-1. Object types included in the Extension](#)" on the next page.

You can view or modify an object type as follows:

1. Log on to PPM.
2. From the menu bar, select **Open > Administration>Open Workbench**.
The PPM Workbench opens.
3. From the shortcut bar, select **Deployment Mgmt>Object Types**.
The Object Type Workbench opens.
4. (Optional) Select a particular Extension from the **Extension** drop-down list.
5. Click **List**.
6. Select the object type of interest and click **Open**.
7. Edit the object type as desired. (On the **Fields** tab, the list of fields in the **Prompts** column is alphabetized.)

Subsequent figures in this section show screens you can use to revise object types in conjunction with adding package lines. You can access these screens as follows:

1. Log on to PPM.
2. From the shortcut bar, select **Deployment Mgmt>Object Types**.
The Object Type Workbench opens.
3. From the shortcut bar, select **Deployment Mgmt > Packages**.
The Package Workbench opens.
4. Add a new or open an existing package, as necessary.
5. Select a workflow.
6. Add a line.
7. Select the object type of interest.

Reference Object Types

Reference object types cannot be edited, but you can copy and rename them and edit the copies to meet your needs. You can also use existing non-reference object types as is or configure them further to meet your needs.

List of Object Types

The Oracle-related object types fall into the following categories:

- **Application data migration object types.** These object types are used to migrate application setup or configuration data. They integrate Object Migrator or GL Migrator

programs with PPM Center, or they automate standard Oracle utilities.

- **Application patching and administration object types.** These object types are used to apply patches and perform standard administration tasks.
- **Instance management object types.** These object types are used to perform instance refreshes.
- **FNDLOAD automation object types.** These object types are templates used to create custom object types to perform functional setup migration by automating the FNDLOAD utility.
- **Run/Patch file system file migration object types:** These object types are used to migrate files between the EBS run file systems or the EBS patch file systems.

"Table A-1. Object types included in the Extension" lists and defines the object types included in the Extension. Each is described in detail in one of the following documents, as indicated in the footnotes:

- Subsequent sections of this appendix
- *Object Migrator Guide*
- *GL Migrator Guide*

"Table A-1. Object types included in the Extension" lists and defines the object types included in the Extension. Each is described in subsequent sections.

Table A-1. Object types included in the Extension

Object Type Name	Description
Application data migration object types	
AOL:Conc Prog ^a	Automates the AOL:Conc Prog migration using Object Migrator.
AOL:Concurrent Manager ^a	Automates the AOL:Concurrent Manager migration using Object Migrator.
AOL:Desc Flex ^a	Automates the AOL:Desc Flex migration using Object Migrator.
AOL:Folder ^a	Automates the AOL:Folder migration using Object Migrator.
AOL:FSG Set ^a	Automates the AOL:FSG Set migration using Object Migrator.
AOL:Function ^a	Automates the AOL:Function migration using Object Migrator.
AOL:GUI Menu ^a	Automates the AOL:GUI Menu migration using Object Migrator.

Table A-1. Object types included in the Extension, continued

Object Type Name	Description
AOL:Message ^a	Automates the AOL:Message migration using Object Migrator.
AOL:Printer Def ^a	Automates the AOL:Printer Def migration using Object Migrator.
AOL:Profile ^a	Automates the AOL:Profile migration using Object Migrator.
AOL:QuickCode ^a	Automates the AOL:QuickCode migration using Object Migrator.
AOL:Report Group ^a	Automates the AOL:Report Group migration using Object Migrator.
AOL:Report Set ^a	Automates the AOL:Report Set migration using Object Migrator.
AOL:Resp ^a	Automates the AOL:Resp migration using Object Migrator.
AOL:Single Conc Mgr Entry ^b	Adds a single specialization rule to an existing Concurrent Manager definition.
AOL:Single GUI Menu Entry ^b	Adds a single entry to an existing GUI Menu definition.
AOL:Single Report Group Unit ^b	Adds a single unit to an existing Report Group definition.
AOL>User ^a	Automates the AOL>User migration using Object Migrator.
AOL:Value Set ^a	Automates the AOL:Value Set migration using Object Migrator.
GL:Budget Orgs ^c	Automates the GL:Budget Orgs migration using GL Migrator.
GL:Consolidations ^c	Automates the GL:Consolidations migration using GL Migrator.
GL:Cross Validation Rules ^c	Automates the GL:Cross Validation Rules migration using GL Migrator.

Table A-1. Object types included in the Extension, continued

Object Type Name	Description
GL:JE Sources ^c	Automates the GL:JE Sources migration using GL Migrator.
GL:Journal Categories ^c	Automates the GL:Journal Categories migration using GL Migrator.
GL:Mass Allocation/Budgets ^c	Automates the GL:Mass Allocation/ Budgets migration using GL Migrator.
GL:Summary Templates ^c	Automates the GL:Summary Templates migration using GL Migrator.
OraApps AKLOAD Migrate AK Setups ^b	Migrates Web setup data using the Oracle AKLOAD utility.
OraApps Oracle Workflow ^b	Migrates an Oracle workflow definition from one environment to another using the Oracle WFLOAD utility.
OraApps ADADMIN Compile APPS Schema ^b	Uses ADADMIN to compile the APPS schema in the destination environment.
OraApps ADADMIN Compile Flexfields ^b	Uses ADADMIN to compile all flexfields in the destination environment.
OraApps ADADMIN Generate Jar Files ^b	Uses ADADMIN to generate some or all .jar files in the destination environment. (Not supported for Oracle Release 11.)
OraApps ADADMIN Generate Messages ^b	Uses ADADMIN to generate some or all message files in the destination environment.
OraApps Merge Patch ^b	Allows multiple Oracle patches to be merged into a single patch.
Oracle Patch ^b	Applies an Oracle patch.
Instance management object type	
OraApps 11i Cloning ^b	Automates the cloning process for an Oracle E-Business Suite Release 11i system.
FNDLOAD automation object types	

Table A-1. Object types included in the Extension, continued

Object Type Name	Description
AR:Phone Country Codes	Automates the migration of the Country Phone Codes used within the Accounts Receivable module.
INV:Units of Measure	Automates the migration of the Units of Measure used within the Inventory module.
Run/Patch file system file migration object types	
OA - Run System File Migration	Migrates files between the two EBS run file systems.
OA - Patch System File Migration	Migrate files between the two EBS patch file systems.
<ul style="list-style-type: none"> a. Documented in the Object Migrator Guide. b. Documented in subsequent sections of this appendix. c. Documented in the GL Migrator Guide. 	

Application Data Migration Object Types

The Extension includes an object type corresponding to each specific migrator in Object Migrator (for example, Concurrent Program, Value Set, and Menu) and each specific migrator in GL Migrator (for example, Consolidations and Summary Templates). The Extension also includes object types for patch application, AD administration, workflow migrations and Web setup data. Some object types are specific to particular Oracle releases.

The following sections provide reference information on selected application data migration object types as indicated in "[Table A-1. Object types included in the Extension](#)" on page 160.

AOL and GL Object Types

The following sections provide a description, migration order information, and field definitions for the Oracle E-Business Suite Object Library (AOL) and Oracle General Ledger (GL) object types.

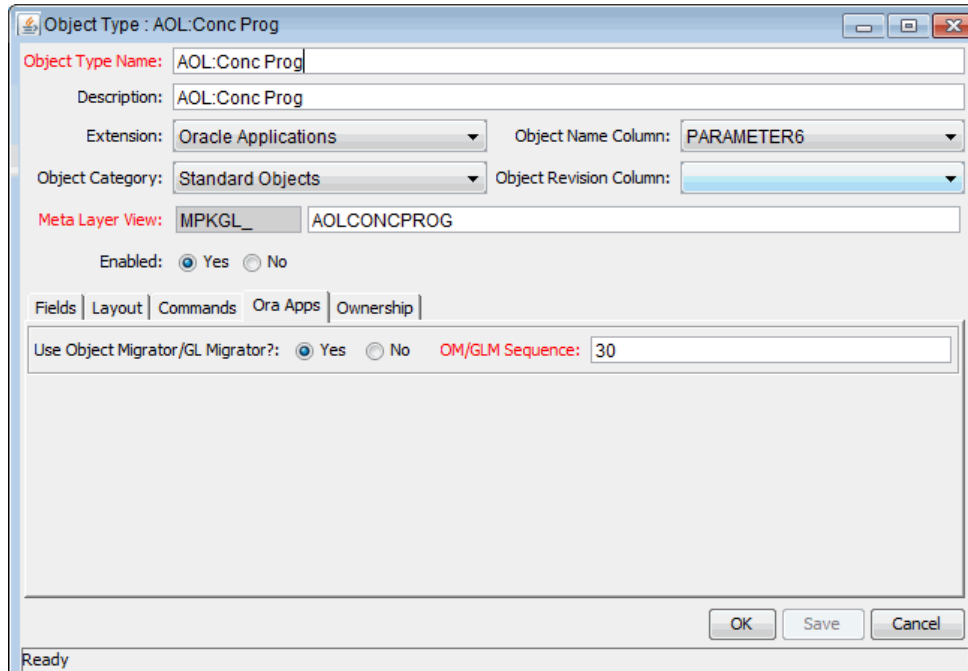
Description

The Extension includes object types that integrate Object Migrator or GL Migrator programs with PPM. Object Migrator and GL Migrator handle the management and migration of AOL and GL setup entities.

Migration Order

The order of and dependencies among object types (for example, Concurrent Programs are dependent on Value Sets) are specified by the **OM/GLM Sequence** field on the **Ora Apps** tab in the Object Type window in the PPM Workbench. "Figure A-1. Object Type window, Ora Apps tab" shows an example.

Figure A-1. Object Type window, Ora Apps tab



The screenshot shows a window titled "Object Type : AOL:Conc Prog". The "Object Type Name" is "AOL:Conc Prog" and the "Description" is "AOL:Conc Prog". The "Extension" is "Oracle Applications" and the "Object Name Column" is "PARAMETER6". The "Object Category" is "Standard Objects" and the "Object Revision Column" is empty. The "Meta Layer View" is "MPKGL_" and "AOLCONCPROG". The "Enabled" radio buttons are "Yes" (selected) and "No". The "Ora Apps" tab is selected, and the "OM/GLM Sequence" field is set to "30". The "Use Object Migrator/GL Migrator?" radio buttons are "Yes" (selected) and "No". The "OK", "Save", and "Cancel" buttons are at the bottom right. The status bar at the bottom left says "Ready".

The value you specify in the **OM/GLM Sequence** field determines the order of submission for Oracle-related requests when multiple package lines are submitted for execution as a batch. Typical dependencies are reflected in the values provided, but you can modify them as necessary.

Field Descriptions

The object type definition contains fields corresponding to each of the fields used by Object Migrator and GL Migrator, although the field layout differs between them.

You can view the fields for each specific migrator in the following ways:

- By adding a package line using the PPM Workbench as described in and as shown in the example for the AOL:Conc Prog object type in "Figure A-2. AOL:Conc Prog object type"
- By querying a specific migrator in the Define Concurrent Programs form in Oracle, as shown in "Figure A-3. Migrator fields from Define Concurrent Programs form in Oracle"

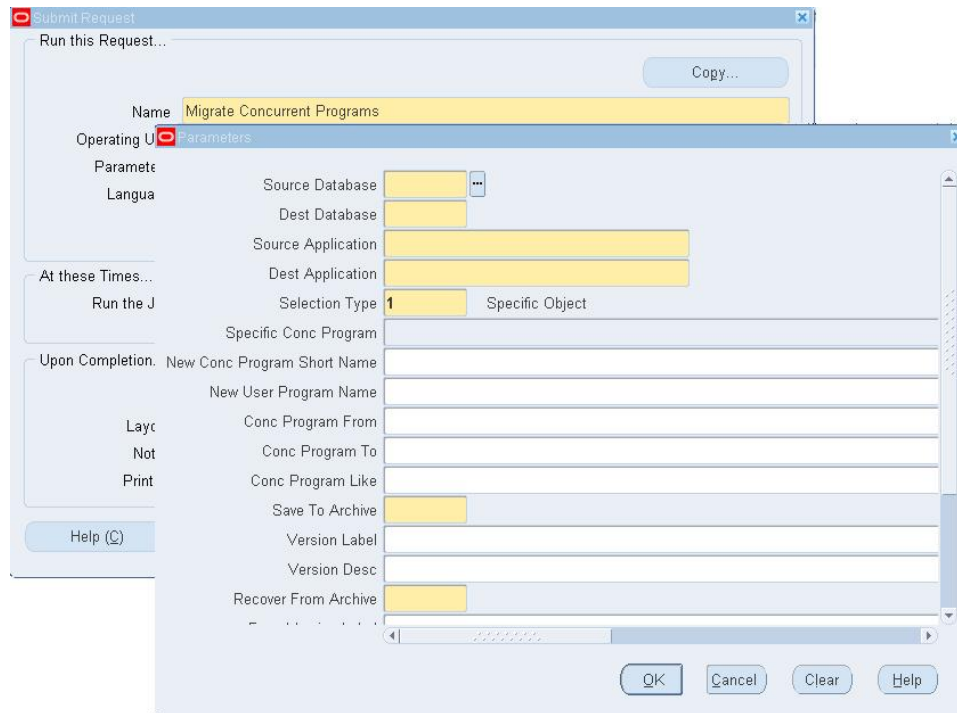
Note:

By default, some of the fields used by Object Migrator and GL Migrator might not be displayed by PPM Center. (For example, the **Compare Only** field is not displayed.) In most cases this represents the most common usage or most efficient way of integrating with PPM Center, allowing the feature of PPM Center to determine some values.

Figure A-2. AOL:Conc Prog object type

The screenshot shows a dialog box titled "Add Line" with a blue border. At the top, there is a section for "Object Type Information" containing a text field for "Object Type" (set to "AOL:Conc Prog"), a text field for "Sequence" (set to "1"), and a dropdown menu for "Application Code" (set to "None"). Below this is a tabbed interface with "Parameters" and "User Data" tabs. The "Parameters" tab is active and contains several fields: "Source Application" (text field), "Selection Type" (dropdown menu set to "Specific Object"), "Conc Program" (text field), "New User Program Name" (text field), "Conc Program From" (text field), "Conc Program To" (text field), "Version Label" (text field), and "Version Desc" (text field). At the bottom of the parameters section are two sets of radio buttons: "Overwrite if Exists" with "Yes" selected, and "Partials Allowed" with "No" selected. At the very bottom of the dialog are four buttons: "Clear", "OK", "Add", and "Cancel". A status bar at the bottom of the dialog reads "'AOL:Conc Prog' parameters loaded."

Figure A-3. Migrator fields from Define Concurrent Programs form in Oracle



AOL:Single Conc Mgr Entry Object Type

The following sections provide a description, configuration consideration, and field descriptions for the AOL:Single Conc Mgr Entry object type.

Description

The AOL:Single Conc Mgr Entry object type allows users to add a single specialization rule to an existing Concurrent Manager definition. The component can be Concurrent Program, User, Oracle ID, Request Type, or Combined Rule. This is useful for minor ongoing maintenance activity that might not follow a classic promotion process.

The concurrent manager to which the item is added, as well as the entry definition values, are all chosen from the source environment. The same items must exist in the target environment for the migration to be successful.

Configuration Consideration

The value of the **Apps DB Link** field in the Environment window, **Extension Data** tab, **Oracle Applications** subtab must be a valid database link from the PPM Center schema.

Field Descriptions

Figure A-4 shows the default screen when adding a package line that uses the AOL:Single Conc Mgr Entry object type. Table A-2 provides field descriptions for the object type.

Figure A-4. AOL:Single Conc Mgr Entry object type

Table A-2. AOL:Single Conc Mgr Entry object type fields

Field Name (*Required)	Description
*Conc Mgr Application	Name of the application to which the Concurrent Manager belongs.
*Conc Mgr Name	Name of the Concurrent Manager to which an entry is added.
*Action Type	Action Type of the entry being added— Allow or Disallow . The default is Allow .
*Program Type	Type of entry being added— Concurrent Program (the default), Request Type , Combined Rule , Oracle ID , or User .
Program Application	Short name of the application to which the entry belongs. Not applicable for Oracle ID and user entries.
*Program Name	Specific name identifying the object to add.

AOL:Single GUI Menu Entry Object Type

The following sections provide a description, configuration consideration, and field descriptions for the AOL:Single GUI Menu Entry object type.

Description

The AOL:Single GUI Menu Entry object type allows users to add a single component to an existing GUI menu. The component can be a function or a submenu.

The GUI menu to which to add the item and the item definition values are chosen from the source environment. The same menu and items must exist in the target environment for the migration to be successful.

Configuration Consideration

The value of the **Apps DB Link** field in the Environment window, **Extension Data** tab, **Oracle Applications** subtab must be a valid database link from the PPM Center schema.

Field Descriptions

Figure A-5 shows the default screen when adding a package line that uses the AOL:Single GUI Menu Entry object type. Table A-3 provides field descriptions for the object type.

Figure A-5. AOL:Single GUI Menu Entry object type

The screenshot shows a dialog box titled "Add Line" with a blue border. At the top, it says "Object Type Information". Below this, there are three fields: "Object Type" with the value "AOL:Single GUI Menu Entry", "Sequence" with the value "1", and "Application Code" with a dropdown menu showing "None". Below these are two tabs: "Parameters" and "User Data". Under the "Parameters" tab, there are five text input fields: "GUI Menu Name", "Menu Entry Seq", "Prompt", "Sub Menu Name", and "Function Name". At the bottom of the dialog are four buttons: "Clear", "OK", "Add", and "Cancel". A status bar at the very bottom says "'AOL:Single GUI Menu Entry' parameters loaded."

Table A-3. AOL:Single GUI Menu Entry object type fields

Field Name (*Required)	Description
*GUI Menu Name	Name of the GUI Menu to which an entry can be added.
*Menu Entry Seq	Sequence to assign to the entry. This should be a number and should not be in use on the menu in the destination.
*Prompt	Prompt the user sees when looking at the menu.
Sub Menu Name	Name of the GUI submenu to be added to the menu. This field should be blank if you are adding a function.
Function Name	Name of the function to add to the menu. This field should be blank if you are adding a Submenu.

AOL:Single Report Group Unit Object Type

The following sections provide a description, configuration considerations, and field descriptions for the AOL:Single Report Group Unit object type.

Description

The AOL:Single Report Group Unit object type allows users to add a single component to an existing report group. The component can be an application, concurrent program, or report set.

Use this object type when adding report group components to an Oracle environment at any release supported by the Extension.

Configuration Consideration

The value of the **Apps DB Link** field in the Environment window, **Extension Data** tab, **Oracle Applications** subtab must be a valid database link from the PPM Center schema.

The report group to which the item is added and the item definition values are all chosen from the source environment.

Field Descriptions

Figure A-6 shows the default screen when adding a package line that uses the AOL:Single Report Group Unit object type. Table A-4 provides field descriptions for the object type.

Figure A-6. AOL:Single Report Group Unit object type

The screenshot shows a dialog box titled "Add Line" with a close button in the top right corner. The main area is titled "Object Type Information" and contains the following fields:

- Object Type:** AOL:Single Report Group Unit
- Sequence:** 1
- Application Code:** None

Below this is a tabbed interface with "Parameters" selected. The "Parameters" tab contains the following fields:

- Report Group Application:** (empty)
- Report Group:** (empty)
- Unit Type:** Concurrent Program
- Unit Application:** (empty)
- Unit Name:** (empty)

At the bottom of the dialog are buttons for "Clear", "OK", "Add", and "Cancel". A status bar at the very bottom reads: "AOL:Single Report Group Unit' parameters loaded."

Table A-4. AOL:Single Report Group Unit object type fields

Field Name (*Required)	Description
*Report Group Application	Application to which the report group belongs
*Report Group Application	Application to which the report group belongs
*Report Group	Report group to which the item will be added
*Unit Type	Type of item being added—Application, Concurrent Program (the default), or Report Set
*Unit Application	Application to use for the new item
*Unit Name	Name of the new item
Function Name	Name of the function to add to the menu. This field should be blank if you are adding a Submenu.

OraApps AKLOAD Migrate AK Setups Object Type

The following sections provide a description, configuration considerations, and field descriptions for the OraApps AKLOAD Migrate AK Setups object type.

Description

Oracle provides a mechanism for users to define the layout and content of a Web page that is integrated with Oracle E-Business Suite. This is accomplished by defining the relevant data elements in a series of forms in the AK (Common Modules) module. The definition process takes a long time due to the amount of input required and the performance of the forms used to input the data.

The OraApps AKLOAD Migrate AK Setups object type allows users to automate the migration of custom Web page configuration data using the Oracle AKLOAD utility, thereby saving effort, increasing auditability, and reducing data entry errors. The following types of data can be migrated:

- Attributes
- Objects
- Regions
- Flows

This object type should be used only to migrate custom data.

Caution:

The AKLOAD utility uses ID values to migrate data. These IDs must match between source and destination instances. If non-matching IDs are found, the execution fails. You can choose to override this behavior, but this is not recommended, since it can lead to unintended results.

Configuration Consideration

When the object type runs, it does the following:

- Verifies whether all the applications that are in the source environment have the same ID values at the destination.
- Causes the execution to fail if the **Check for Application Differences** field is set to **Yes** and differences are found.

Prints a warning message in the execution log and allows the execution to continue if the **Check for Application Differences** field is set to **No** and differences are found.

- Connects to the source environment and extracts the AK data. This can include creation of a temporary directory and script. The execution fails if errors are found in the AKLOAD log.
- Connects to the destination environment, determines whether the extract directory exists, and creates the directory if it does not exist.
- Moves the extract file from source to destination.
- Connects to the source environment and removes the extract file if the **Clean the Extract Files** field is set to **Yes**.
- Connects to the destination environment and checks whether the extract file contains any non-application raw ID values.
- Causes the execution to fail if the **Check for Raw IDs** field is set to **Yes** and raw IDs are found.
Prints a warning message in the execution log and allows the execution to continue if the **Check for Raw IDs** field is set to **No** and raw IDs are found.
- Imports the AK data. This can include creation of a temporary directory and script. The execution fails if errors are found in the AKLOAD log.
- Connects to the destination environment and removes the extract file if the **Clean the Extract Files** field is set to **Yes**.

Caution:

Only custom data should be migrated with this object type. The object type does not validate or prevent migration of standard or Micro Focus-supplied data.

By default, this object type is configured to connect using JDBC. The object type can be configured to use other protocols supported by AKLOAD.

Data should be migrated between Oracle instances that are the same point release and patch level. The object type does not check the Oracle version.

Field Descriptions

"[Figure A-7. OraApps AKLOAD Migrate AK Setups object type](#)" shows the default screen when adding a package line that uses the OraApps AKLOAD Migrate AK Setups object type. "[Table A-5. OraApps AKLOAD Migrate AK Setups object type fields](#)" provides field descriptions for the object type.

Figure A-7. OraApps AKLOAD Migrate AK Setups object type

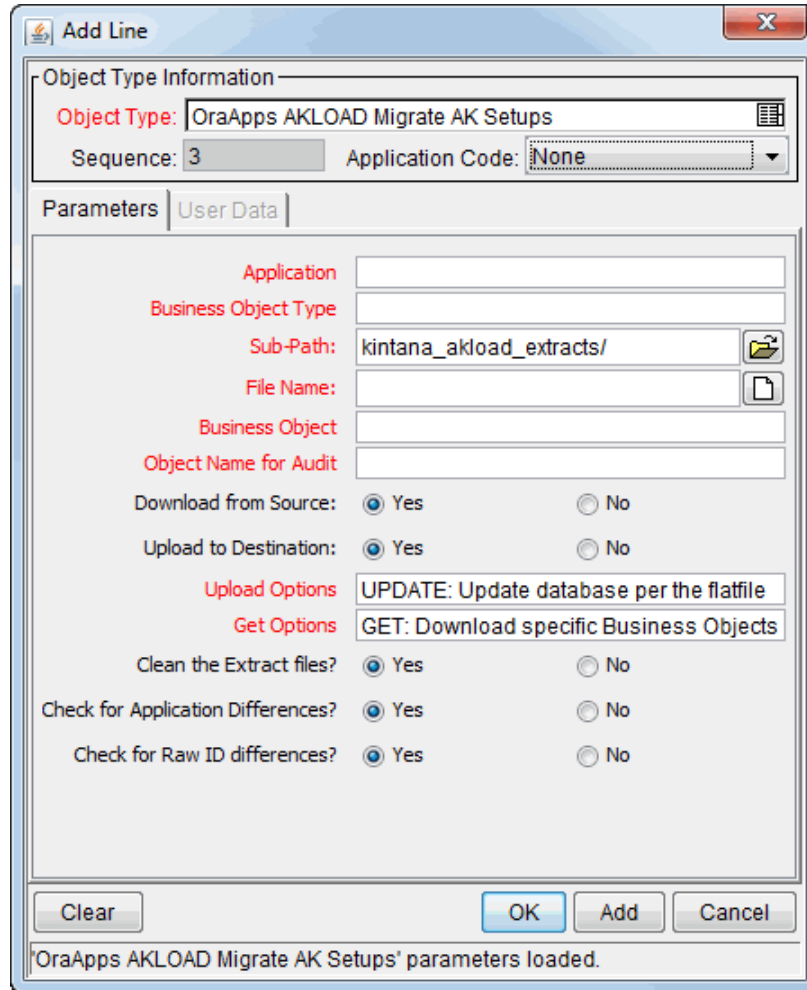


Table A-5. OraApps AKLOAD Migrate AK Setups object type fields

Field Name (*Required)	Description
*Application	Application from which objects are downloaded and uploaded.
*Business Object Type	Type of business object being migrated— REGION , FLOW , ATTRIBUTE , or OBJECT . This value must be reselected when the Application selection changes.
*Sub-Path	Subdirectory under the base path where the extract file should go.
*File Name	Extract file name. (.jlt extension).

Table A-5. OraApps AKLOAD Migrate AK Setups object type fields, continued

Field Name (*Required)	Description
*Business Object	Specific business entity being migrated. You can select more than one entity from the specified Application . This value of this field depends on the values for Application and Business Object Type , and you must re-select it when either of these values change.
*Object Name for Audit	Name that will be used as an audit trail for the AKLOAD migration. This name appears in the environment contents and history reporting for the object type. By default, the value selected will be the same as for the Business Object field, but delimited with commas rather than semicolons.
Download from Source	Export is performed only if this field is set to Yes .
Upload to Destination	Import is performed only if this field is set to Yes .
*Upload Options	Upload options for import.
*Get Options	Download options for import.
Clean the Extract files	Option to remove the extract files from the source and destination after successful migration.
Check for Application Differences	If this field is set to Yes , all Source Applications' short names and application IDs are validated against the destination DB, and an error results if they do not match.
Check for Raw ID differences	If this field is set to Yes , all object_id and menu_id values in the extract file are validated against the destination DB, and an error results if they do not match.

Note:

Any database views upon which your Web setup depends should already exist in the destination instance. Existence of the database views is not validated by the object type.

Migration of security data is not supported. The security information can be migrated using the AOL:Resp migrator.

OraApps Oracle Workflow Object Type

The following sections provide a description, configuration considerations, and field descriptions for the OraApps Oracle Workflow object type.

Description

The OraApps Oracle Workflow object type allows users to migrate embedded Oracle workflow definitions between Oracle E-Business Suite instances. By using PPM Center to migrate these definitions, users can increase the security, control, and visibility of these changes, and integrate with a source code control system.

This object type uses the standard Oracle WFLOAD utility to accomplish the upload or download of the workflow definition. The following migration types are supported (all three types can be integrated with source code control tools):

- **Upload from File.** The object moves an existing workflow definition file from the source environment to the destination environment and loads the file into the destination database. The file is left at the destination.
- **Download to File.** The workflow definition is downloaded from the source database to the source server.
- **Upload from Database.** The workflow definition is downloaded from the source database, moved to the destination server, and then uploaded to the destination database. The temporary file used to transport the definition is removed.

Note:

The workflow versions in the source and destination should be the same. This is not validated by the object type.

Configuration Consideration

Any environment defined in PPM Center that will be a source or target of a workflow migration must have a valid value in the **UTL_FILE_DIR** field in the Environment window, **Extension Data** tab, Oracle Applications subtab. This field defines the location where workflows will be downloaded to or uploaded from, and should correspond to a **UTL_FILE_DIR** entry in the environment database's *init.ora* file.

Each usage option can be configured for use with a source control tool. The **Upload from File** option can be integrated to retrieve the file from a source control system. The **Download to File** option can be configured to check the file into a source control system. The **Upload from Database** option can be configured to check in the downloaded definition to a source control system. If this is required, the derivation and storage of a temporary file name should be removed.

Field Descriptions

Figure A-8 shows the default screen when adding a package line that uses the OraApps Oracle Workflow object type. Table A-6 provides field descriptions for the object type.

Figure A-8. OraApps Oracle Workflow object type

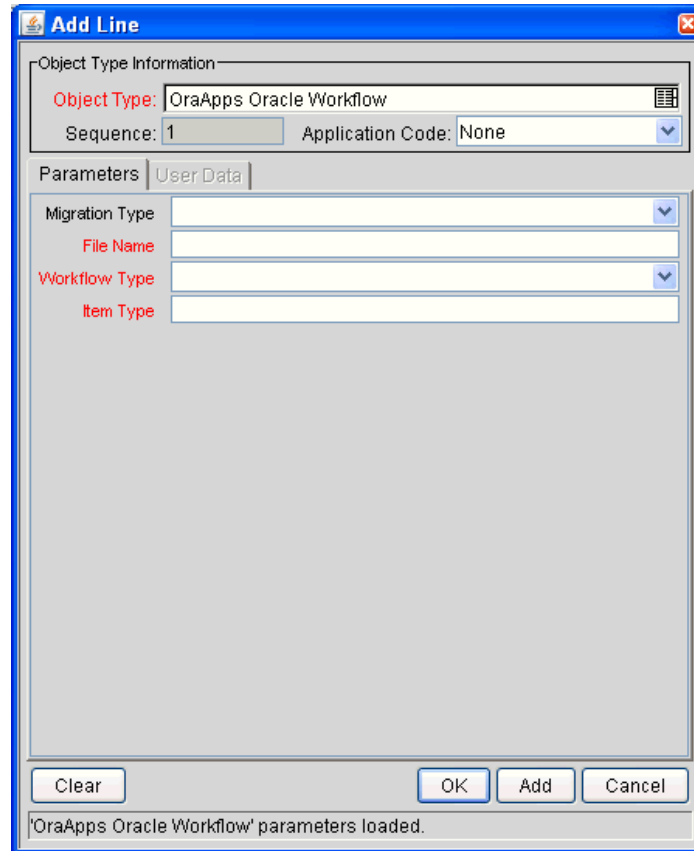


Table A-6. OraApps Oracle Workflow object type fields

Field Name (*Required)	Description
Migration Type	Type of migration— Upload from File , Upload from Database , or Download to File .
*File Name	Type of business object being migrated— REGION , FLOW , ATTRIBUTE , or OBJECT . This value must be reselected when the Application selection changes.
*Workflow Type	Type of workflow— Oracle Application Seeded Workflow , New Custom Workflow , or Changed to Custom Workflow .
*Item Type	Workflow Item Type (for example, OEOL/OEOH). Multiple values are allowed and must be space delimited.

Patch and Administration Object Types

With a small amount of configuration, you can automate a large portion of the patch application and tracking process using the Extension patch management tools. The Extension automates Oracle’s ADPATCH and ADADMIN utilities through the standard PPM Center workflow and execution engines, with sophisticated extras such as rules-based intelligence.

The Extension applies Oracle server-side and middle-tier patches that use the ADPATCH program.

The following sections provide reference information on the patch and administration object types listed in ["Table A-1. Object types included in the Extension"](#).

Oracle Patch Object Type

The following sections provide a description and field descriptions for the Oracle Patch object type.

For Oracle E-Business Suite Release 11.5.10 or later (including Release 12), AutoConfig can be called automatically from ADPATCH to instantiate and run new templates contained in a patch. However, AutoConfig can modify environment files. If the patched instance runs on Windows, you might need to modify the environment context file that the Oracle Patch object type uses. If AutoConfig is run automatically during patch application, the object type adds the following warning to the Package Execution Log: "AutoConfig was executed by this patch. Environment files for Windows may need to be modified." To create an environment context file for the Oracle E-Business Suite instances running on Windows servers, see the following Description section.

Description

The Extension contains the Oracle Patch object type for applying patches. For more information, see ["Managing Patches" on page 130](#).

Note:

Oracle's product documentation includes specific instructions for applying patches using unified drivers to multitier Oracle E-Business Suite instances. If the workflows you are using to apply patches to your Oracle E-Business Suite instances conflict with Oracle's guidelines, you should modify the workflows before applying patches with unified drivers.

The Oracle Patch object type expects to call an environment context file to set the Oracle context. By default, for UNIX environments the object type assumes that the file name is the Oracle SID name.

For Windows environments, to create an environment context file that is compatible with the bash shell, you can use the `kacc_oa_windows_config.sh` script (located in the `<PPM_Home>/scripts` directory, where `<PPM_Home>` represents the path where your PPM Center instance is installed). You must modify the appropriate object type commands so they call this file to set the environment context information for the destination instance before the object type invokes ADPATCH or ADADMIN.

Field Descriptions

Figure A-9 shows the default screen when adding a package line that uses the default Oracle Patch object type. Table A-7 provides field descriptions for the object type descriptions for the displayed fields and for some fields that are hidden by default.

Figure A-9. Oracle Patch object type

Table A-7. Oracle Patch object type fields

Field Name (*Required)	Description
*Patch Type	The OraApps Backend Server option causes the source or destination environment’s server information to be used for the patch. The OraApps Forms Server option causes the source or destination environment’s client information to be used for the patch.
*Bug No	Bug number for which the patch is relevant. Used for auditing.
*Patch Archive	File from which the patch is to be extracted.

Table A-7. Oracle Patch object type fields, continued

Field Name (*Required)	Description
Tar Number	Tracking number from Oracle Support.
Copy Patch To Dest	Option to copy the patch files to each environment being patched. Set to No if the patch already exists in each physical environment.
Execute Patch @Dest	Option to execute the patch in the destination environment.
Restart	In the event of patch failure, indicates if ADPATCH should continue from the point of failure, or restart.
Drv Files (.drv, *.drv ...)	Names of the driver files to be used in the patch application, separated by commas or spaces.
*Driver Execution Options	Indicates which portions of the patch driver to execute based on the kind of driver specified. Specific portions of the driver can be chosen only when a U-driver is specified, and should be chosen in accordance with Oracle's patch instructions. Options are All Portions , Copy Portion , Database Portion , or Generate Portion .
*Log File Name	Name of the file in the Oracle instance to which the patch application output will be written.
*No. of Parallel Workers	Used by the Oracle ADPATCH utility to determine the number of parallel processes used to handle the patch.
*Batch Size	Used by the Oracle ADPATCH utility to determine processing batch size.
Email Adpatch Failures	Standard with ADPATCH. Option to send email notifications of failures. This is not required if notifications have been set up for the workflow, which is the recommended way of sending email notifications with Micro Focus products.
Email Recipients (spc delim)	Email recipients receiving ADPATCH failure notifications. Space delimited.
Apply Multi-Lingual Patch Now (Hidden by default)	The default value is No . Select Yes to apply this multilingual patch. Oracle's ADPATCH utility requests confirmation when applying patches to multilingual environments, because multiple patch files might be required.

Table A-7. Oracle Patch object type fields, continued

Field Name (*Required)	Description
Terminate on Worker Failure	The default value is No . If the value is No , PPM Center continues to wait for outside intervention to fix failed workers. Select Yes to have the patch execution fail if all workers fail.
Append Patch Data (Hidden by default)	Option to evaluate the entire C-driver file and U-driver file to derive patch detail data if the patch already exists in the detail tables. Initially Yes , set this field to No after the patch is captured the first time for the package line.
Reapply Patch	Option to reapply the patch if it is already applied to the destination environment. Default is No . To reapply the patch, set this field to Yes .
Tolerate Applications System Name conflict	Option to allow the ADPATCH utility to continue without error if the Applications System Name stored in the database and the name in the APPL_TOP file system are different, which usually indicates a misconfiguration. Default is No .

OraApps Merge Patch Object Type

The following sections provide a description, configuration considerations, and field descriptions for the OraApps Merge Patch object type.

Description

The OraApps Merge Patch object type allows users to merge multiple compatible Oracle Release 11i or Release 12 patches into a single patch using the Oracle standard ADMRGPCH utility. This decreases the number of individual patches to be tracked and applied, and can decrease the total time dedicated to tracking and applying patches.

This object type takes specified patch archive files from the Patch Stage central repository, uses ADMRGPCH to merge them into a single patch in an Oracle Release 11i or Release 12 instance, and then returns the merged patch archive to the Patch Stage repository.

For information about important considerations for merging patches, see ["Merging Patches" on page 136](#).

The object type produces a new patch archive with a user-specified name that can be used with the "Oracle Patch" object types to apply the patch.

AD Merge Patch does not merge patches of different releases, different platforms, or different parallel modes.

AD Merge Patch does not consider any dependencies regarding the order in which patches should be applied. Users must be careful regarding the patches they merge together if there are application-order dependencies.

Configuration Consideration

The patch archives to be merged must exist in the patch stage area, and the Patch Stage environment must be defined. An environment must be defined where ADMRGPCH can run. If patches with Unified drivers will be merged, the environment should support Unified drivers.

Since merging patches is a one-time-only occurrence, it might be useful to create a simple workflow just to accomplish the merge.

The Merged Archive name assigned will be used by Patch Applicator to apply the merged patch, and will identify the patch in the `Environment History` file. You should select a meaningful name.

Field Descriptions

Figure A-10 shows the default screen when adding a package line that uses the OraApps Merge Patch object type. Table A-8 provides field descriptions for the object type.

Figure A-10. OraApps Merge Patch object type

Table A-7. Oracle Patch object type fields

Field Name (*Required)	Description
*Archives to be Merged	List of archives that must be merged. File names must be separated by commas or spaces. The list can contain up to a total of 200 characters. (A shorter length might be required depending upon the limitations of your operating system, because the list is used in operating system commands.)
*Merged Archive (New Bug#)	Name of the Merged Patch without file extension. This value should be a valid UNIX file name and should not contain any spaces or control characters. This value is used as a Bug Number when the Merged Patch archive is being applied through Patch Applicator, and also for patch detail reporting.
*Merged Archive Type	The type of the merged archive to be created (for example, zip or tar).
Tar Number	List of tar numbers for which patches are being merged. For Information and reporting purpose only.

Table A-7. Oracle Patch object type fields, continued

Field Name (*Required)	Description
*Overwrite Flag	Option to overwrite an existing merged file.
*Partial Allowed	Option to create the merged patch if some of the patch archives in the list of archives could not be found in the patch stage area.

OraApps ADADMIN Compile APPS Schema Object Type

The following sections provide a description, configuration considerations, and field descriptions for the OraApps ADADMIN Compile APPS Schema object type.

Description

By invoking the ADADMIN utility, the OraApps ADADMIN Compile APPS Schema object type recompiles one or more APPS schemas in an Oracle instance at Release 11, Release 11i, or Release 12. The object type responds to ADADMIN prompts dynamically, based on environment and package line information, eliminating the need for the user to answer the prompts interactively. The object type then navigates the ADADMIN menus, selecting the correct options to compile the APPS schemas.

ADADMIN spawns parallel workers to recompile invalid entities in the APPS schemas. This task can be run at any time, but is usually run after applying packages that alter packages in the APPS schema, or after installing custom packages that need compilation. If more than one APPS schema is defined, ADADMIN compiles all the schemas. After executing, the object type exits ADADMIN. A log of the interactive session is attached to the PPM Center execution log.

Configuration Consideration

You must set Oracle context information before invoking ADADMIN.

Note: The Oracle Patch object type expects to call an environment context file to set the Oracle context. By default, for UNIX environments the object type assumes that the file name is the Oracle SID name.

For Windows environments, to create an environment context file that is compatible with the bash shell, you can use the `kacc_oa_windows_config.sh` script (located in the `<PPM_Home>/scripts` directory, where `<PPM_Home>` represents the path where your PPM Center instance is installed). You must modify the appropriate object type commands so they call this file to set the environment context information for the destination instance before the object type invokes ADPATCH or ADADMIN.

Field Descriptions

"[Figure A-11. OraApps ADADMIN Compile APPS Schema object type](#)" shows the default screen when adding a package line that uses the OraApps ADADMIN Compile APPS Schema object type. "[Table A-9. OraApps ADADMIN Compile APPS Schema object type fields](#)" provides descriptions for the displayed fields and for some fields that are hidden by default.

Figure A-11. OraApps ADADMIN Compile APPS Schema object type

The screenshot shows a dialog box titled "Add Line" with a close button in the top right corner. The "Object Type Information" section contains:

- Object Type: OraApps ADADMIN Compile APPS Schema
- Sequence: 1
- Application Code: None

 Below this is a tabbed interface with "Parameters" selected. The "Parameters" section includes:

- Environment tier: Ora Apps Backend Server (dropdown)
- No. of Parallel Workers: 4 (text input)
- Invoker Rights Mode: Incremental (dropdown)
- Restart?: Yes (radio), No (radio, selected)
- Log File Name: adadmin_[PKG.NUMBER].log (text input)
- Batch Size: 1000 (text input)
- Email Adadmin Failures?: Yes (radio), No (radio, selected)
- Email Recipients (spc delim): applmgr (text input)
- Terminate on Worker Failure?: Yes (radio), No (radio, selected)

 At the bottom are buttons for "Clear", "OK", "Add", and "Cancel". A status bar at the very bottom reads: "'OraApps ADADMIN Compile APPS Schema' parameters loaded."

Table A-9. OraApps ADADMIN Compile APPS Schema object type fields

Field Name (*Required)	Description
*Environment tier	The OraApps Backend Server option causes the object to use the environment's server information. The OraApps Forms Server option causes the object to use the environment's client information.
Disregard init.ora errors (Hidden by default)	Select Yes to continue running ADADMIN if it detects potential issues with init.ora. The default value is No .
*No. of Parallel Workers	Number of workers to run in parallel.
Invoker Rights Mode	Mode for invoker rights.

Table A-9. OraApps ADADMIN Compile APPS Schema object type fields, continued

Field Name (*Required)	Description
Restart	Option to restart an interrupted session. We strongly recommend retaining the default option of No .
*Log File Name	Log name to be created in Oracle instance.
*Batch Size	Size of the batch to run.
Email Adadmin Failures	Option to have Oracle send an email if ADADMIN fails.
Email Recipients (spc delim)	List of users who should get email notices. Space delimited.
Terminate on Worker Failure	If the value is No (the default), PPM Center continues to wait for outside intervention to fix failed workers. Select Yes to have the patch execution automatically fail if all workers fail.

OraApps ADADMIN Compile Flexfields Object Type

The following sections provide a description, configuration considerations, and field descriptions for the OraApps ADADMIN Compile Flexfields object type.

Description

By invoking the ADADMIN utility, the OraApps ADADMIN Compile Flexfields object type recompiles all flexfields defined in an Oracle instance at Release 11, Release 11i, or Release 12. This function might be required after patching or other changes in order to make sure that the flexfield views are accurate.

Field Descriptions

"[Figure A-12. OraApps ADADMIN Compile Flexfields object type](#)" shows the default screen when adding a package line that uses the OraApps ADADMIN Compile Flexfields object type. "[Table A-10. OraApps ADADMIN Compile Flexfields object type fields](#)" provides descriptions of the object type's displayed fields and some of the fields that are hidden by default.

Figure A-12. OraApps ADADMIN Compile Flexfields object type

The screenshot shows a dialog box titled "Add Line" with a blue border. It contains the following fields and controls:

- Object Type Information:**
 - Object Type: OraApps ADADMIN Compile Flexfields
 - Sequence: 1
 - Application Code: None
- Parameters / User Data:**
 - Environment tier: Ora Apps Backend Server
 - Tolerate Compilation Errors?: Yes No
 - Restart?: Yes No
 - Log File Name: adadmin_[PKG.NUMBER].log
 - Batch Size: 1000
 - Email Adadmin Failures?: Yes No
 - Email Recipients (spc delim): applmgr
- Buttons:** Clear, OK, Add, Cancel
- Status Bar:** 'OraApps ADADMIN Compile Flexfields' parameters loaded.

Table A-10. OraApps ADADMIN Compile Flexfields object type fields

Field Name (*Required)	Description
*Environment tier	The OraApps Backend Server option causes the object to use the environment’s server information. The OraApps Forms Server option causes the object to use the environment’s client information.
Disregard init.ora errors (Hidden by default)	Select Yes to continue running ADADMIN if it detects potential issues with init.ora. The default value is No .
Tolerate Compilation Errors	Option to treat compilation errors as a failure (No), or to tolerate them and allow execution to complete successfully even if compilation errors are encountered (Yes).
Restart	Option to restart an interrupted session. We strongly recommend retaining the default option of No .
*Log File Name	Log name to be created in Oracle instance.
*Batch Size	Size of the batch.

Table A-10. OraApps ADADMIN Compile Flexfields object type fields, continued

Field Name (*Required)	Description
Email Adadmin Failures	Option to have Oracle send an email if ADADMIN fails.
Email Recipients (spc delim)	List of users who should get email notices.

OraApps ADADMIN Generate Jar Files Object Type

The following sections provide a description and field descriptions for the OraApps ADADMIN Generate Jar Files object type.

Description

By invoking the ADADMIN utility, the OraApps ADADMIN Generate Jar Files object type generates either out of date or all (FORCE option) jarfiles in an Oracle Release 11i or Release 12 instance. This task can be run at any time, but is usually run after applying forms patches.

Note:

This ADADMIN function is supported on Release 11i and Release 12 only. Attempting to administer a Release 11 environment with this object type results in a failed execution.

Field Descriptions

"[Figure A-13. OraApps ADADMIN Generate Jar Files object type](#)" shows the default screen when adding a package line that uses the OraApps ADADMIN Generate Jar Files object type. "[Table A-11. OraApps ADADMIN Generate Jar Files object type fields](#)" provides field descriptions for the object type.

Figure A-13. OraApps ADADMIN Generate Jar Files object type

The screenshot shows a dialog box titled "Add Line" with a blue border. It contains the following fields and options:

- Object Type Information:**
 - Object Type: OraApps ADADMIN Generate Jar Files
 - Sequence: 1
 - Application Code: None
- Parameters / User Data:**
 - Environment tier: Ora Apps Backend Server
 - Force generation of ALL Jars?: Yes No
 - Restart?: Yes No
 - Log File Name: adadmin_[PKG.NUMBER].log
 - Batch Size: 1000
 - Email Adadmin Failures?: Yes No
 - Email Recipients (spc delim): applmgr
- Buttons:** Clear, OK, Add, Cancel
- Status Bar:** 'OraApps ADADMIN Generate Jar Files' parameters loaded.

Table A-11. OraApps ADADMIN Generate Jar Files object type fields

Field Name (*Required)	Description
*Environment tier	The OraApps Backend Server option causes the object to use the environment’s server information. The OraApps Forms Server option causes the object to use the environment’s client information.
Force generation of ALL Jars	Option to force generation of all jar files. Selecting Yes forces generation of all jar files.
Restart	Option to restart an interrupted session. We strongly recommend retaining the default option of No .
*Log File Name	Log name to be created in Oracle instance.
*Batch Size	Size of the batch.

Table A-11. OraApps ADADMIN Generate Jar Files object type fields, continued

Field Name (*Required)	Description
Email Adadmin Failures	Option to have Oracle send an email if ADADMIN fails.
Email Recipients (spc delim)	List of recipients for email notices.

OraApps ADADMIN Generate Messages Object Type

The following sections provide a description and field descriptions for the OraApps ADADMIN Generate Messages object type.

Description

The OraApps ADADMIN Generate Messages object type uses the Oracle ADADMIN utility to generate binary message files for applications in an Oracle instance at Release 11, Release 11i, or Release 12. This task is usually run after applying a patch that requests message regeneration, but the task can be run on an incremental basis. Oracle uses the message files to display online messages to users. Some versions allow the user to specify the products (applications) and languages for which to generate messages, while earlier versions always generate messages for all languages.

Field Descriptions

"[Figure A-14. OraApps ADADMIN Generate Messages object type](#)" shows the default screen when adding a package line that uses the OraApps ADADMIN Generate Messages object type. "[Table A-12. OraApps ADADMIN Generate Messages object type fields](#)" provides field descriptions for the object type.

Figure A-14. OraApps ADADMIN Generate Messages object type

The screenshot shows a dialog box titled "Add Line" with a tabbed interface. The "Object Type Information" tab is active, showing "Object Type: OraApps ADADMIN Generate Messages", "Sequence: 1", and "Application Code: None". The "Parameters" tab is also visible, showing various configuration options: "Environment tier" (Ora Apps Backend Server), "Restart?" (No), "Log File Name" (adadmin_[PKG.NUMBER].log), "Batch Size" (1000), "Products (inv ap ...)" (all), "Languages (US DK ...)" (all), "No. of Parallel Workers" (4), "Terminate on Worker Failure?" (No), "Email Adadmin Failures?" (No), and "Email Recipients (spc delim)" (applmgr). At the bottom, there are "Clear", "OK", "Add", and "Cancel" buttons. A status bar at the very bottom reads "'OraApps ADADMIN Generate Messages' parameters loaded."

Table A-12. OraApps ADADMIN Generate Messages object type fields

Field Name (*Required)	Description
*Environment tier	The OraApps Backend Server option causes the object to use the environment’s server information. The OraApps Forms Server option causes the object to use the environment’s client information.
Force generation of ALL Jars	Option to force generation of all jar files. Selecting Yes forces generation of all jar files.
Restart	Option to restart an interrupted session. We strongly recommend retaining the default option of No .
*Log File Name	Log name to be created in Oracle instance.
*Batch Size	Size of the batch.

Table A-12. OraApps ADADMIN Generate Messages object type fields, continued

Field Name (*Required)	Description
*Products (inv ap...)	List of products (application codes) for which to generate messages, or generates messages for all products. Multiple values should be space delimited. The default value is all . Applicable only if ADADMIN allows generation of messages for specific products.
*Languages (US DK...)	List of languages (language codes) for which to generate messages, or generates messages in all installed languages. Multiple values should be space delimited. The default value is all . Applicable only if ADADMIN allows generation of messages for specific languages.
*No. of Parallel Workers	Number of parallel processes used to handle the administration activity. Applicable only if ADADMIN uses parallel workers for message generation.
Terminate on Worker Failure	Yes if Patch Applicator should force a termination when all workers fail, or No if Patch Applicator should continue to wait.
Email Adadmin Failures	Option to have Oracle send an email if ADADMIN fails.
Email Recipients (spc delim)	List of recipients for email notices.

Instance Management Object Type

The following section provides reference information on the instance management object type listed in "[Table A-1. Object types included in the Extension](#)" on page 160.

OraApps 11i Cloning Object Type

The following sections provide a description, configuration considerations, and field descriptions for the OraApps 11i Cloning object type.

Description

The OraApps 11i Cloning object type is used in the process of cloning instances. Using Oracle E-Business Suite requires management of a number of ongoing operating instances and temporary implementation and upgrade instances. These instances must be efficiently created, maintained, and periodically cloned without risking the stability of the system and disruption of day-to-day activities.

The OraApps 11i Cloning object type provides a means to clone an Oracle E-Business Suite instance by automating the execution of major steps in the process. Information pertinent to the environments to be refreshed is captured in the object type. This information is to be used during the execution of commands that clone the instance. Only the databases and directories targeted to be cloned are affected. This object type works in conjunction with the OraApps 11i Cloning workflow.

Caution:

Cloning your Oracle instance is a complex process, and misconfigurations can easily result in unexpected consequences. This object type and its associated workflow provide a template for cloning your instances, but they require tailoring for your environment. Contact Micro Focus Software Support to facilitate a successful implementation and see the Micro Focus Software Support Web site:

<https://softwaresupport.hpe.com>

The OraApps 11i Cloning object type supports cloning for any Oracle E-Business Suite Release 11i instance that does not use AutoConfig, which is a configuration option for Oracle E-Business Suite Releases from 11.5.1 through 11.5.6.

When the object type is run, it executes the following actions based on which step the workflow is currently on:

- Verifies if all cloning requirements are met. This includes checking for adequate disk space, the existence of Perl on the destination machine, whether the Oracle patch for cloning has been applied to the destination instance, and whether the files required for handling the database exist.
- Copies and executes cloning scripts in pre-clone mode. This preserves configuration information regarding the destination instance and shuts down the various application services.
- Generates a list of database files to be deleted and shuts down the destination database in preparation for the clone.
- Deletes all database files in the list.
- Copies all database files from the source to the destination.
- Re-creates control files needed to be able to get the database in working order.
- Restarts the database.
- Cleans up concurrent request tables that might contain executions relevant only to the source instance.
- Deletes and copies application files from the source to the destination.
- Copies and executes cloning scripts in post-clone mode. This restores the configuration information of the destination instance and restarts the various application services.

The process implemented by this object type and its associated workflow follow the guidelines set in the *Cloning Oracle Applications Release 11i* white paper published by Oracle.

Configuration Consideration

If any of the standard workflow step names in the OraApps 11i Cloning workflow are altered, you must apply the correct modifications to the OraApps 11i Cloning object type command condition, because the various commands execute based on the workflow step name.

Caution:

Cloning your Oracle instance is a complex process, and misconfigurations can yield unexpected consequences. This object type and its associated workflow provide a template for cloning your instances, but requires tailoring for your environment.

Contact Micro Focus Software Support to facilitate a successful implementation and see the Micro Focus Software Support Web site:

<https://softwaresupport.hpe.com>

Field Descriptions

Figure A-15 shows the default screen when adding a package line that uses the OraApps 11i Cloning object type. Table A-13 provides field descriptions for the object type.

Figure A-15. OraApps 11i Cloning object type sample data

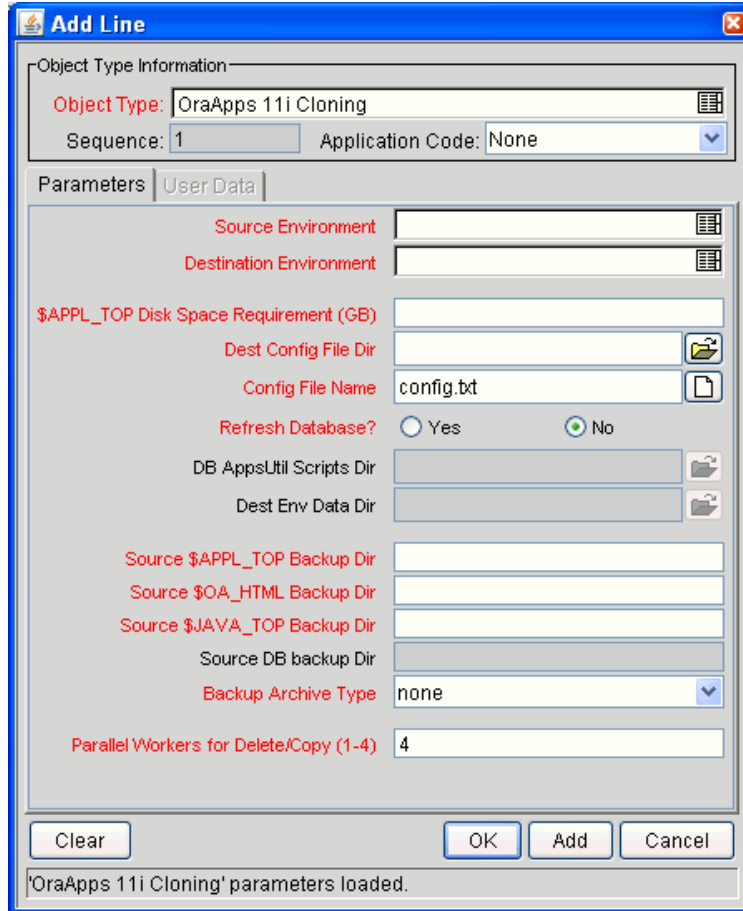


Table A-13. OraApps 11i Cloning object type fields

Field Name (*Required)	Description
*Source Environment	Source environment to be used for cloning.
*Destination Environment	Destination environment to be used for cloning.
*\$APPL_TOP Disk Space Requirement (GB)	Amount of space (in gigabytes) required at the destination.
*Dest Config File Dir	Location of the installation configuration file used by the destination instance.
*Config File Name	Name of the installation configuration file.
*Refresh Database	Option to refresh the database as well as the application code.

Table A-13. OraApps 11i Cloning object type fields, continued

Field Name (*Required)	Description
DB AppsUtil Scripts Dir	Location of database script files addbctl.sh and addlnctl.sh. These files are used to stop and start the listener and database.
Dest Env Data Dir	Destination data file directory.
*Source \$APPL_TOP Backup Dir	Directory where the source \$APPL_TOP code backup files are stored.
*Source \$OA_HTML Backup Dir	Directory where the source \$OA_HTML code backup files are stored.
*Source \$JAVA_TOP Backup Dir	Directory where the source \$JAVA_TOP code backup files are stored.
Source DB backup Dir	Directory where the source database backup files are stored.
*Backup Archive Type	Type of archiving used to back up source files—none, tar, or zip.
*Parallel Workers for Delete/Copy (1-4)	Number of parallel workers to be used for delete and copy processes.

FNDLOAD Automation Object Types

The following sections provide reference information on the FNDLOAD automation object types listed in ["Table A-1. Object types included in the Extension" on page 160](#).

AR:Phone Country Codes Object Type

The following sections provide a description, configuration considerations, and field descriptions for the AR:Phone Country Codes object type.

Description

Working with Oracle's FNDLOAD utility, the AR:Phone Country Codes object type automates the migration of the Phone Country Codes used within the Accounts Receivable module.

For further introductory information about this object type, see ["Overview of FNDLOAD Automation Object Types" on page 117](#).

Configuration Consideration

For introductory information about configuring this object type, see ["Overview of FNDLOAD Automation Object Types" on page 117](#).

Caution:

If you use this object type as a template to build custom objects for other entities that you intend to use with FNDLOAD, you must fully understand the functionality of those entities and create custom FNDLOAD control (.lct) files for each one. You must ensure that the control file includes all the business logic and validations needed to maintain the referential integrity for Oracle entity IDs across the instances during a migration. Using a control file that does not include the necessary business logic and validations can lead to data corruption at the destination.

Field Descriptions

"[Figure A-16. AR:Phone Country Codes object type](#)" shows the default screen when adding a package line that uses the AR:Phone Country Codes object type. "[Table A-14. AR:Phone Country Codes object type fields](#)" provides field descriptions for the object type.

Figure A-16. AR:Phone Country Codes object type

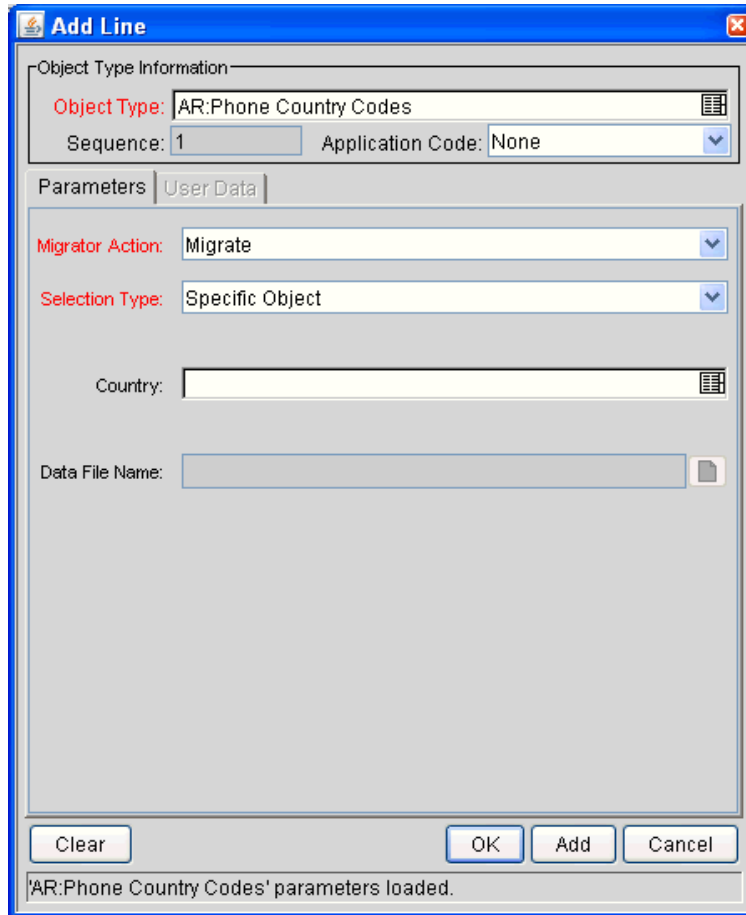


Table A-14. AR:Phone Country Codes object type fields

Field Name (*Required)	Description
*Migrator Action	Action to be performed—Migrate, Export, or Import.
*Selection Type	Option to perform the Migrator Action on a Specific Object (Phone Country Code) or on All Objects.
Country	Name of the phone country code on which to perform the Migrator Action if Selection Type is Specific Object. Disabled if Selection Type is All Objects.
Data File Name	FNDLOAD data file name (.ldt) if Migrator Action is Export or Import. Disabled if Migrator Action is Migrate.

INV:Units of Measure Object Type

The following sections provide a description, configuration considerations, and field descriptions for the INV:Units of Measure object type.

Description

Working with Oracle's FNDLOAD utility, the INV:Units of Measure object type automates the migration of the Units of Measure used within the Inventory module.

For further introductory information about this object type, see ["Overview of FNDLOAD Automation Object Types" on page 117](#).

Configuration Consideration

For introductory information about configuring this object type, see ["Overview of FNDLOAD Automation Object Types" on page 117](#).

Caution:

If you use this object type as a template to build custom objects for other entities that you intend to use with FNDLOAD, you must fully understand the functionality of those entities and create custom FNDLOAD control (.lct) files for each one. You must ensure that the control file includes all the business logic and validations needed to maintain the referential integrity for Oracle entity IDs across the instances during a migration. Using a control file that does not include the necessary business logic and validations can lead to data corruption at the destination.

Field Descriptions

["Figure A-17. INV:Units of Measure object type"](#) shows the default screen when adding a package line that uses the INV:Units of Measure object type. ["Table A-15. INV:Units of Measure object type fields"](#) provides field descriptions for the object type.

Figure A-17. INV:Units of Measure object type

Table A-15. INV:Units of Measure object type fields

Field Name (*Required)	Description
*Migrator Action	Action to be performed—Migrate, Export, or Import.
*Selection Type	Option to perform the Migrator Action on a Specific Object (Phone Country Code) or on All Objects.
Country	Name of the phone country code on which to perform the Migrator Action if Selection Type is Specific Object. Disabled if Selection Type is All Objects.
Data File Name	FNDLOAD data file name (.ldt) if Migrator Action is Export or Import. Disabled if Migrator Action is Migrate.

Run/Patch File System File Migration object types

EBS12.2 has two file systems: run file system and patch file system. The two file systems are located in two folders: *fs1* and *fs2*. The locations of the two file systems are

frequently cutover between the two folders when the EBS system upgrades or performs other operations. Sometimes the run file system is located in f_{s1} and the patch file system is located in f_{s2} . But sometimes the run file system switches to f_{s2} and the patch file system is located in f_{s1} . Files in run or patch file systems are not dedicated in a fixed position. You must change the position of the files when migrating the files in the EBS system after the run and patch file system cutover.

Run/Patch File System File Migration object types enable you to migrate files in the EBS system without the need to change the file locations after the run and patch file system cutover.

OA- Run System File Migration object type

The following sections provide the description, configuration considerations, and field descriptions for the OA - Run System File Migration object type.

Description

The OA- Run System File Migration object type allows users to automatically migrate files in run file system between the two EBS systems.

Configuration consideration

- Enable both the **Server** and **Client** in the **Host** tab of the Environment window.
- In the **Application** tab, set **Server Base Path** to the f_{s1} path location and **Client Base Path** to the f_{s2} path location for each application.

Field descriptions

"[Figure A-18. OA:Run System File Migration object type](#)" below shows the default screen when adding a package line that uses the OA- Run System File Migration object type. "[Table A-16. OA:Run System File Migration object type fields](#)" on the next page provides field descriptions for the object type.

Figure A-18. OA:Run System File Migration object type

The screenshot shows a dialog box titled "Add Line" with a close button (X) in the top right corner. The dialog is divided into two main sections: "Object Type Information" and "Parameters".

Object Type Information:

- Object Type:** OA - Run System File Migration (with a list icon)
- Sequence:** 1 (text input)
- Application Code:** CLM (dropdown menu)

Parameters:

- Source Env File Dir:** /u01/install/APPS (text input)
- File Location:** Client (text input with list icon)
- Sub-Path:** log/ (text input with folder icon)
- File Name:** "test.txt" (text input with document icon)
- Dest Env File Dir:** /u01/install/APPS (text input)

At the bottom of the dialog, there are buttons for "Clear", "OK", "Add", and "Cancel". A status bar at the very bottom reads: "OA - Run System File Migration' parameters loaded."

Table A-16. OA:Run System File Migration object type fields

Field Name (*Required)	Description
Source Env File Dir	The directory of file <code>EBSapps.env</code> in the source environment. Commands will be executed to source to this env file to get the <code>run</code> file system location. If the value is empty, the env file is considered to be located in the Source environment's Server Base Path . If the env does not exist in this directory, retrieving the list of file location will get an error.

Field Name (*Required)	Description
*File Location	The location of the run file system. The value is either Server or Client . The value will be automatically loaded after you click the Select button on the right side. If the run file system of source environment is in <i>fs1</i> , Server will be selected. If the run file system of source environment is in <i>fs2</i> , Client will be selected.
*Sub-Path	The sub path of the files that you are going to migrate.
*File Name	The file that you are going to migrate.
Dest Env File Dir	The directory of the file <code>EBSapps.env</code> in the destination environment. Commands will be executed to source to this env file to get the run file system location. If the value is empty, the env file is considered to be located in the Destination environment's Server Base Path . If the env does not exist in this directory, migration will fail.

OA- Patch System File Migration object type

The following sections provide the description, configuration considerations, and field descriptions for the OA - Patch System File Migration object type.

Description

The OA- Patch System File Migration object type allows users to automatically migrate files in patch file system between the two EBS systems.

Configuration consideration

- Enable both the **Server** and **Client** in the **Host** tab of the Environment window.
- In the **Application** tab, set **Server Base Path** to the *fs1* path location and **Client Base Path** to the *fs2* path location for each application.

Field descriptions

"[OA- Patch System File Migration object type](#)" above shows the default screen when adding a package line that uses the OA- Run System File Migration object type. "[OA- Patch System File Migration object type](#)" above provides field descriptions for the object type.

Figure A-19. OA:Patch System File Migration object type

The screenshot shows a dialog box titled "Add Line" with a close button (X) in the top right corner. The dialog is divided into several sections:

- Object Type Information:**
 - Object Type: OA - Patch System File Migration (with a list icon)
 - Sequence: 1 (text input)
 - Application Code: CLM (dropdown menu)
- Parameters:** (selected tab)
 - Source Env File Dir: /u01/install/APPS (text input)
 - File Location: Server (text input with list icon)
 - Sub-Path: log/ (text input with folder icon)
 - File Name: "patchetest.txt" (text input with document icon)
 - Dest Env File Dir: /u01/install/APPS (text input)
- User Data:** (unselected tab)
- Buttons:** Clear, OK, Add, Cancel
- Status Bar:** OA - Patch System File Migration' parameters loaded.

Table A-17. OA: Run System File Migration object type fields

Field Name (*Required)	Description
Source Env File Dir	The directory of file <code>EBSapps.env</code> in the source environment. Commands will be executed to source to this env file to get the patch file system location. If the value is empty, the env file is considered to be located in the Source environment's Server Base Path . If the env does not exist in this directory, retrieving the list of file location will get an error.

Field Name (*Required)	Description
*File Location	The location of the patch file system. The value is either Server or Client . The value will be automatically loaded after you click the Select button on the right side. If the patch file system of the source environment is in <i>fs1</i> , Server will be selected. If the patch file system of the source environment is in <i>fs2</i> , Client will be selected.
*Sub-Path	The sub path of the files that you are going to migrate.
*File Name	The file that you are going to migrate.
Dest Env File Dir	The directory of the file <code>EBSapps.env</code> in the destination environment. Commands will be executed to source to this env file to get the patch file system location. If the value is empty, the env file is considered to be located in the Destination environment's Server Base Path . If the env does not exist in this directory, migration will fail.

Special commands to get Run/Patch file system

This section introduces the special commands used to get the run/patch file systems.

ksc_oa_get_run_fs

Run this command to get the run file system. It sets the value of the run file system to the token `FS_EDITION`. The value of this token is *fs1* or *fs2*.

Usage

```
ksc_oa_get_run_fs DEST_ENV=[DEST_ENV] P_ENV_DIR=[P_ENV_DIR]
```

Parameters

`DEST_ENV`: The EBS environment from which you want to get the run file system. This parameter is mandatory.

`P_ENV_DIR`: The directory of the `EBSapps.env` locates. This parameter is optional. If this parameter is not set, the environment's Server Base Path will be used as the env file's directory.

ksc_oa_get_patch_fs

Use this command to get the patch file system. It sets the value of the patch file system to the token `FS_EDITION`. The value of this token is *fs1* or *fs2*.

Usage

```
ksc_oa_get_patch_fs DEST_ENV=[DEST_ENV] P_ENV_DIR=[P_ENV_DIR]
```

same as ksc_oa_get_run_fs

Validations to get Run /Patch file system location

This section describes the validations to get the run/patch file system location.

OA - Run File System location

Use this validation to get a list of the run file system locations. There is only one selection option in the list. The value of the selection option is either **Server** or **Client**. If the run file system is in f_{s1} , then the value is Server. If the run file system is in f_{s2} , then the value is Client.

OA - Patch File System location

Use this validation to get a list of the patch file system locations. There is only one selection option in the list. The value of the selection option is **Server** or **Client**. If the run file system is in f_{s1} , then the value is Server. If the run file system is in f_{s2} , then the value is Client.

Appendix B: Request Types

This section provides reference information about the Oracle-specific request types provided in the Extension. The request types are listed and defined in Table B-1.

Migration and compilation of request types can be driven by commands included within the request types. For more information about commands in the PPM Center environment, see the *Commands, Tokens, and Validations Guide and Reference*. For more information about using request types in packages, see the.

Subsequent figures of request type sample data in this appendix show windows you can use to create requests from existing request types. You can access a request type window as described in the following section.

All requests follow the same general process, as described in the following section. Each request type works in conjunction with a particular workflow that is also part of the Extension.

Overview of the Request Process

When functionality (for example, a gap analysis, report, setup change, or cloning request) is required for a certain business area, a business analyst is notified of this need by the group or individual with the requirement.

The business analyst gathers detailed requirements related to the request. These detailed requirements are checked for appropriateness and completeness, and are then forwarded to an approver (probably a business owner). The business owner looks at the requirements, asks for more information if necessary, and then accepts or rejects the requirements.

At this stage or at any of the later stages of the process, information such as screenshots, reports, and comments can be attached to the request. A PPM Center request type consolidates all relevant information for a request in one location so that requestors, analysts, and approvers can all access the information.

If the requirements are accepted, they are prioritized and matched with existing functionality (if such functionality exists). A solution review team, usually consisting of the business owner and the track owner, reviews the existing functionality and accepts or rejects it as a suitable solution to the request.

If the existing functionality matches the requirements, the business owner is notified and the request is closed.

If the existing functionality does not match the requirements, the track owner is notified to prioritize the request, which now becomes a request for development.

The track owner prioritizes the request and begins the detailed functional design phase. Functional designs are created and signed off, and the team lead is notified for approval.

The team lead might request more information about the functional designs, and this discussion might lead to design changes. Modified functional designs must go back through the sign-off process.

When the team lead accepts the design, a developer or team of developers is assigned. The developer creates technical designs, and these designs are reviewed against the requirements. During the technical approval phase, the technical designs might need to be modified.

When the technical design is approved, development begins. Until deployment is complete, the request waits at the package creation step. When the development is complete, the request closes and the original requestor is notified.

Each workflow step in the process has security groups assigned to it, so only authorized persons can act on those steps. At each approval step, package creation and completion notifications can be sent to appropriate users.

Subsequent figures in this appendix show windows you can use to create requests from existing request types. You can access these windows as follows:

1. Log on to PPM.
2. From the menu bar, select **Open > Administration > Open Workbench**.
The PPM Workbench opens.
3. From the shortcut bar, select **Demand Mgmt > Request Types**.
The Request Type Workbench opens.
4. Make sure that the desired request type is enabled. If necessary, open and enable it.
5. From the menu bar in the standard interface, select **Demand Management > Create a Request**.
The Create New Request page appears.

For more information see ["Configuring Request Types" on page 54](#).

Reference Request Types

Reference request types cannot be edited, but you can copy and rename them and edit the copies to meet your needs. You can also use existing non-reference request types as is or configure them further to meet your needs.

List of Request Types

["Table B-1. Request types included in the Extension"](#) lists and defines the request types included in the Extension. Each is described in subsequent sections.

Table B-1. Request types included in the Extension

Request Type	Workflow
OraApps Application Issue	OraApps Conversion Process

Table B-1. Request types included in the Extension, continued

Request Type	Workflow
OraApps Cloning Request	Streamlines the process of initiating and tracking a request to clone an Oracle E-Business Suite instan
OraAppsConversion Request	Initiates the process of converting data from legacy systems or third-party software for importing into Oracle systems
OraApps Design & Development	Consolidates information about Oracle configuration and customization requirements and the analysis needed to review and approve them
OraApps Enhancement Request	Initiates the process of requesting enhancements for an Oracle system
OraApps GAP Analysis Request	Initiates the process of requesting a gap analysis of the differences between existing company processes and those of Oracle E-Business Suite
OraApps Interface Request	Initiates the process of requesting a new interface to import or export data between Oracle tables and other software programs
OraApps Report Request	Initiates the process of developing a new or modified report for an Oracle system
OraApps Setup Change Request	Initiates the process of requesting a new setup for an Oracle system
OraApps Status Update Request	Initiates the process of gathering and reviewing status information
OraApps Interface Request	Initiates the process of requesting a new interface to import or export data between Oracle tables and other software programs

OraApps Application Issue Request Type

The following sections provide a description, configuration considerations, and field descriptions for the OraApps Application Issue request type.

Description

During an Oracle E-Business Suite implementation or upgrade, there is a constant influx of new problems, questions, and requests. There is usually no central repository of

information to store the issues, and they can originate from a number of different sources. Issues can “fall through the cracks” and end up being stored in testing scripts, design documents, or emails. Unless these issues are captured and addressed quickly, they often do not get resolved and end up creating a backlog that jeopardizes the success of the project.

The OraApps Application Issue request type is used to track issues at an Oracle implementation site. This request type provides a means for specifying information needed to resolve an issue. Requestors are prompted for such Oracle environment information as the application in which the issue is occurring, the browser being used, and the operating system. This request type is designed to work with the OraApps Application Issue workflow.

For a description of request processes, see .

Configuration Considerations

Most of the fields in the Details section of the OraApps Application Issue request type refer to a drop-down list of values. You can tailor each of these validation lists to give you the list of values appropriate to your business.

The default Details fields for an OraApps Application Issue request type are suggested information you might want to capture about Oracle Application issues. However, you can remove and add fields as required. If you remove fields, you must make corresponding changes to the reports associated with the Oracle issue, because these reports reference some of the Details fields.

You might decide to make some of the fields required or not required at different stages of issue resolution. You might also decide to place security on individual fields. You make these changes in the Request Type window.

You should also consider the following before you use or configure the OraApps Application Issue request type:

- The request type must be enabled before it can be used.
- By default, all users can create this type of request.
- The OraApps Application Issue workflow should be the default and the only allowed workflow assigned to this request type.
- Most of the Details fields in this request type refer to a drop-down list of values, which can be configured as required.
- You can configure security levels for this request type.

For more information about the workflow associated with the OraApps Application Issue request type, see OraApps Application Issue Workflow on page 306.

For information about the reports associated with this request type, see OraApps Apps Issues Summary Report on page 367 and OraApps Apps Issues Detail Report on page 362.

Field Descriptions

"Figure B-1. OraApps Application Issue request creation" shows the window for creating an OraApps Application Issue request based on the provided request type. "Table B-2. OraApps Application Issue request type fields" provides field descriptions for the request type.

Figure B-1. OraApps Application Issue request creation

Table B-2. OraApps Application Issue request type fields

Field Name (*Required)	Description
Summary section ^a	

Table B-2. OraApps Application Issue request type fields, continued

Field Name (*Required)	Description
*Workflow	OraApps Application Issue.
Problem/Resolution section	
*Problem	Summary of the specific problem, issue, or enhancement request.
*Business Area Affected	General business group or department affected by the issue.
*Source	Origin of the issue, such as GAP Analysis , CRP A , or Integration Test .
Resolution	Detailed resolution of the issue.
Solution	Detailed resolution of the issue.
Required Date	Required date by which the issue must be resolved.
Source Category	Nature of the issue. The field has Micro Focus-supplied values such as Form , Interface , and Report .
Project Team	Project team responsible for the functional area addressed in the issue.
Environment section	
Environment	Name of the environment with which there is an issue.
Apps Application	Oracle Application in the environment experiencing the issue.
Apps Responsibility	Oracle Responsibilities in the environment experiencing the issue.
Apps Username	Oracle Applications user in the environment experiencing the issue.
Apps Form	Name of the Oracle Applications Form with which there is an issue.
Browser Version	Browser version being used when the issue was found.

Table B-2. OraApps Application Issue request type fields, continued

Field Name (*Required)	Description
Desktop OS	Desktop operating system.
Jinitiator Version	Jinitiator version.
DB Version	DB version of the database.
Analysis section	
Business Benefit	Level of business benefit (High, Medium, or Low) estimated by the request creator.
Level of Effort	Level of effort (Very Easy, Easy, Medium, Hard, or Very Hard) estimated by the assigned developer.
Scope Change	Whether the problem is out of the original scope and needs a scope change to be resolved.
Internal/External	Whether the request needs internal or external resolution. External indicates that there is an Oracle patch required to solve the request.
TAR #	Unique identifier for the TAR. This information is provided by Oracle Support.
Technical Impact	Level of system impact (High, Medium, or Low) estimated by the analyst or assigned developer.
Est Completion Date	Estimated completion date for resolving the request.
Oracle Bug No.	Oracle bug number for the issue.
a. The Summary section is not expanded in "Figure B-1. OraApps Application Issue request creation" .	

OraApps Cloning Request Request Type

The following sections provide a description, configuration consideration, and field descriptions for the OraApps Cloning Request request type.

Description

The OraApps Cloning Request request type lets you streamline the process of initiating and tracking a request to have an Oracle instance cloned. This request type works with the OraApps 11i Cloning object type and the OraApps Cloning Process workflow.

For a description of request processes, see ["Overview of the Request Process" on page 207](#).

Configuration Considerations

The OraApps Cloning Process workflow should be the default and the only allowed workflow assigned to the OraApps Cloning Request request type.

For more information about the object type associated with the OraApps Cloning Request request type, see ["OraApps 11i Cloning Object Type" on page 192](#).

For more information about the workflow associated with this request type, see ["OraApps Cloning Process Workflow" on page 243](#).

Field Descriptions

["Figure B-2. OraApps Cloning Request creation"](#) shows the window for creating an OraApps Cloning Request based on the provided request type. ["Table B-3. OraApps Cloning Request request type fields"](#) provides field descriptions for the request type.

Figure B-2. OraApps Cloning Request creation

Create New OraApps Cloning Request

Jump To

- Summary
- Business Details
- Analysis
- Notes
- References

Expand All | Collapse All

+ Summary

- Business Details

Business Area:

Business Benefit:

* New Environment: Yes No

Application:

* Business Justification:

* Source Environment:

Dest Environment:

* Database Only ? Yes No

* Requested By:

- Analysis

Technical Impact:

Est. Effort:

Est. Start Date:

Est. Completion Date:

Analysis:

+ Notes

+ References

Table B-3. OraApps Cloning Request request type fields

Field Name (*Required)	Description
Summary section^a	
*Workflow	OraApps Application Issue.
Business Details section	
Business Area	General business group or department requesting the clone
Business Benefit	Level of business benefit (High, Medium, or Low) the cloning initiative is expected to have
*New Environment	Option to require a new destination environment
Application	List of applications under a business area that are requesting the cloning
*Business Justification	Justification for creating a clone of an instance

Table B-3. OraApps Cloning Request request type fields, continued

Field Name (*Required)	Description
*Source Environment	Source environment used for cloning
Dest Environment	Destination environment used for cloning
*Database Only	Option to refresh only the database
*Requested By	Requested completion date
Analysis section	
Technical Impact	Level of Technical Impact (High, Medium, or Low) the cloning initiative is expected to have
Est. Effort	Number of estimated hours required for the cloning effort
Est. Start Date	Estimated date the cloning will begin
Est. Completion Date	Estimated date the cloning will be complete and ready for use
Analysis	Analysis of the requested task
a. The Summary section is not expanded in "Figure B-2. OraApps Cloning Request creation" .	

OraApps Conversion Request Request Type

The following sections provide a description, configuration consideration, and field descriptions for the OraApps Conversion Request request type.

Description

Organizations using Oracle E-Business Suite might have used other legacy systems for similar functionality in the past. The data from these systems must be transformed and imported into Oracle using interfaces, and the correct tables must be populated with this data. For example, an organization that decides to use Oracle CRM must obtain all the contact information from legacy systems and put that information into Oracle before starting to use Oracle for CRM.

Organizations using Oracle E-Business Suite might also have used third-party software applications in the past. In order to be brought into Oracle, the data from these applications must go through conversion, as well. In both these examples, data conversion is a one-time-only occurrence.

The OraApps Conversion Request request type initiates the process of converting data from either legacy systems or third-party software into an Oracle system. This request type works in conjunction with the OraApps Conversion Process workflow.

For a description of request processes, see ["Overview of the Request Process" on page 207](#).

Configuration Considerations

The OraApps Conversion Process workflow should be the default and the only allowed workflow assigned to the OraApps Conversion Request request type.

For more information about this workflow, see ["OraApps Conversion Process Workflow" on page 244](#).

Field Descriptions

["Figure B-3. OraApps Conversion Request creation"](#) shows the window for creating an OraApps Conversion Request based on the provided request type. ["Table B-4. OraApps Conversion Request request type fields"](#) provides field descriptions for the request type.

Figure B-3. OraApps Conversion Request creation

The screenshot shows the 'Create New OraApps Conversion Request' form. The form is titled 'Create New OraApps Conversion Request' and has a 'Jump To' sidebar with links for Summary, Conversion Details, Analysis, Notes, and References. The main form is divided into sections: Summary, Conversion Details, Analysis, Notes, and References. The Conversion Details section includes fields for Conversion Name, Business Area, Application, Requested By, and Description Detail. The Analysis section includes fields for Technical Effort, Est. Effort (hrs), Est. Completion Date, and Project Team.

Table B-4. OraApps Conversion Request request type fields

Field Name (*Required)	Description
Summary section^a	
*Workflow	OraApps Application Issue.
Conversion Details section	
*Conversion Name	Name of the conversion
Business Area	Business area requesting the conversion
Application	Application under the business area requesting the conversion
*Requested By	Detailed description of the conversion
*Description Detail	Justification for creating a clone of an instance
External System Name	External system name
External System Owner	External system owner
Analysis section	
Technical Effort	Technical effort required for the conversion of data
Est. Effort (hrs)	Estimated effort (in hours) required for conversion
Est. Completion Date	Estimated completion date
Project Team	Project team assigned
Analysis	Details of the conversion analysis
a. The Summary section is not expanded in " Figure B-3. OraApps Conversion Request creation ".	

OraApps Design & Development Request Type

The following sections provide a description, configuration considerations, and field descriptions for the OraApps Design & Development request type.

Description

During a typical Oracle E-Business Suite implementation or upgrade, organizations devote a significant amount of time and effort to designing, developing, and rolling out new or updated configurations and changes. The high volume of configurations and customizations is difficult to track and control. A process for managing the configurations and customizations is necessary for a successful implementation.

Before any development work is done, the specific requirements driving the change must be identified and reviewed. It is also important that the requirements be approved and prioritized before significant design and development time is spent. Without this, resources, time, and money are spent in areas that do not significantly help the business while higher priority needs are left unattended or with inadequate resources.

The OraApps Design & Development request type gathers summary information about the requirement, including information such as a summary description, specific business areas being addressed, and the business benefit the request is expected to provide. This request type also gathers the major analysis information that is needed when reviewing and approving the requirement. Finally, through notes and attachments, the request type serves as the central repository for detailed requirements and justification documents, so that everyone in the design and development process has access to all the information surrounding the specific effort. The OraApps Design & Development request type is designed to work in conjunction with the OraApps Design & Development workflow.

For a description of request processes, see ["Overview of the Request Process" on page 207](#).

Configuration Considerations

The OraApps Design & Development workflow should be the default and the only allowed workflow assigned to the OraApps Design & Development request type.

Most of the Details fields in the OraApps Design & Development request type refer to a drop-down list of values. You can tailor each of these validation lists to give you the list of values appropriate to your business.

If you want to gather additional information during the design and development process, you can add fields to the OraApps Design & Development request type.

For more information about the workflow associated with this request type, see ["OraApps Design & Development Workflow" on page 251](#).

Field Descriptions

["Figure B-4. OraApps Design & Development request creation"](#) shows the window for creating an OraApps Design & Development request based on the provided request

type. "Table B-5. OraApps Design & Development request type fields"⁵ provides field descriptions for the request type.

Figure B-4. OraApps Design & Development request creation

Table B-5. OraApps Design & Development request type fields

Field Name (*Required)	Description
Summary section^a	
*Workflow	OraApps Design & Development.
Dev. Details section	
*Application	List of the applications by which design and development units should be categorized. This can be set to be general categories such as Manufacturing or Finance or more granular ones such as individual Oracle Application codes.
Unit Category	Technical nature of the unit. Micro Focus supplies such values as Form, Interface, and Report .
Business Area	General business group or department that will be affected by the unit category. While multiple groups might be affected, this field signifies the group that is most affected.

Table B-5. OraApps Design & Development request type fields, continued

Field Name (*Required)	Description
*Source	List of where the requests for this unit came from. This could include values such as GAP Analysis, CRP 1, or Conversion Test.
Business Benefit	Level of business benefit (High, Medium, or Low) the cloning initiative is expected to have.
Technical Impact	Level of Technical Impact (High, Medium, or Low) the cloning initiative is expected to have. Example: A new report might have a low impact, while an update to a major form might have a high impact.
Level of Effort	Length of time it will take to design and develop this new unit.
Est. Effort	Number of days required for the design and development effort.
Est. Completion Date	Estimated date the unit will be complete and ready for deployment.
*Summary	Short summary of the unit. This allows a user to quickly get to the overview of the unit without going through the notes or opening up any attachments.
a. The Summary section is not expanded in "Figure B-4. OraApps Design & Development request creation" .	
a. The Summary section is not expanded in "Field Descriptions" .	

OraApps Enhancement Request Request Type

The following sections provide a description, configuration considerations, and field descriptions for the OraApps Enhancement Request request type.

Description

As business requirements within an Oracle system change, the existing functionality must be modified or new functionality enhanced. Proposed enhancements to the system go through a process of business and technical review and approval.

The OraApps Enhancement request type initiates the process of requesting enhancements for an Oracle system. This request type works with the OraApps Enhancement Process workflow.

For a description of request processes, see ["Overview of the Request Process" on page 207](#).

Configuration Considerations

The OraApps Enhancement Process workflow should be the default and the only allowed workflow assigned to the OraApps Enhancement Request request type.

For more information about this workflow, see ["OraApps Enhancement Process Workflow" on page 254](#).

Field Descriptions

["Figure B-5. OraApps Enhancement Request creation"](#) shows the window for creating an OraApps Enhancement Request based on the provided request type. ["Table B-6. OraApps Enhancement Request request type fields"](#) provides field descriptions for the request type.

Figure B-5. OraApps Enhancement Request creation

The screenshot shows a web-based form titled "Create New OraApps Enhancement Request". On the left, a "Jump To" sidebar contains a list of sections: Summary, Enhancement Det..., Analysis, Notes, and References. The main form area is titled "Expand All | Collapse All" and contains several sections:

- Summary:** A section header with a plus sign.
- Enhancement Details:** A section header with a minus sign. It contains:
 - Enhancement Name:** A text input field.
 - Detailed Description:** A large text area with a scroll bar.
 - New Enhancement:** Radio buttons for "Yes" and "No", with "No" selected.
 - Requested By:** A text input field with a user selection icon.
 - Business Area:** A dropdown menu.
 - Application:** A dropdown menu.
- Analysis:** A section header with a minus sign. It contains:
 - Technical Effort:** A dropdown menu.
 - Est. Effort (hrs):** A text input field.
 - Est. Completion Date:** A text input field with a user selection icon.
 - Project Team:** A dropdown menu.
 - Analysis:** A large text area with a scroll bar.
- Notes:** A section header with a plus sign.
- References:** A section header with a plus sign.

Table B-6. OraApps Enhancement Request request type fields

Field Name (*Required)	Description
Summary section^a	
*Workflow	OraApps Enhancement Process
Enhancement Details section	
*Enhancement Name	Name of the requested enhancement
*Detailed Description	Detailed description of the enhancement
*New Enhancement	Option to require a new enhancement
Business Area	Business area requesting the enhancement
*Requested By	Date by which the enhancement is required to be completed
Application	Application under the business area requesting the enhancement
Analysis section	
Technical Effort	Technical effort required for the enhancement
Est. Effort (hrs)	Estimated effort (in hours) required for the enhancement
Est. Completion Date	Estimated completion date
Project Team	Project team assigned
Analysis	Details of the enhancement analysis
*Summary	Short summary of the unit. This allows a user to quickly get to the overview of the unit without going through the notes or opening up any attachments.
a. The Summary section is not expanded in "Figure B-5. OraApps Enhancement Request creation" .	

OraApps GAP Analysis Request Request Type

The following sections provide a description, configuration consideration, and field descriptions for the OraApps GAP Analysis Request request type.

Description

An implementation or upgrade in an Oracle system can mean significant changes to the system's business processes. The difference between the existing processes and those provided by the packaged application is referred to as a "gap." Stakeholders in an Oracle-related project must be able to request a "gap analysis" for specific Oracle functionality.

The OraApps GAP Analysis Request request type initiates the process of requesting a gap analysis for an Oracle system. Project managers, track owners, and project sponsors use this request type to create a gap analysis.

Once a request for a gap analysis has been created, the request should be attached to relevant project tasks.

The OraApps GAP Analysis Request request type works with the OraApps GAP Analysis Process workflow, which consists of approval, assignment, and analysis steps.

For a description of request processes, see ["Overview of the Request Process" on page 207](#).

Configuration Considerations

The OraApps GAP Analysis Process workflow should be the default and the only allowed workflow assigned to the OraApps GAP Analysis Request request type.

For more information about this workflow, see ["OraApps GAP Analysis Process Workflow" on page 256](#).

Field Descriptions

["Figure B-6. OraApps GAP Analysis Request creation"](#) shows the window for creating an OraApps GAP Analysis Request based on the provided request type. ["Table B-7. OraApps GAP Analysis Request request type fields"](#) provides field descriptions for the request type.

Figure B-6. OraApps GAP Analysis Request creation

Create New OraApps GAP Analysis Request

The screenshot shows a web-based form for creating a new OraApps GAP Analysis Request. The form has a sidebar on the left with a 'Jump To' menu containing links for Summary, GAP Details, Analysis, Notes, and References. The main content area is titled 'Expand All | Collapse All' and contains several sections:

- Summary:** A single-line text field.
- GAP Details:**
 - *GAP Name: A single-line text field.
 - Business Area: A dropdown menu.
 - Application: A dropdown menu.
 - *Requested By: A single-line text field with a user selection icon.
 - *Business Requirement: A multi-line text area.
 - *GAP Description: A multi-line text area.
- Analysis:**
 - Technical Effort: A dropdown menu.
 - Est. Effort (hrs): A single-line text field.
 - Est. Completion Date: A single-line text field with a date selection icon.
 - Project Team: A dropdown menu.
 - Analysis: A multi-line text area.
- Notes:** A section header for a notes field.
- References:** A section header for a references field.

Table B-7. OraApps GAP Analysis Request request type fields

Field Name (*Required)	Description
Summary section^a	
*Workflow	OraApps GAP Analysis Process
GAP Details section	
*GAP Name	Name of the gap
Business Area	Business area requesting the gap analysis
Application	Application under the business area requesting the gap analysis
*Requested By	Date by which the enhancement is required to be completed
*Business Requirement	Description of the business requirements

Table B-7. OraApps GAP Analysis Request request type fields, continued

Field Name (*Required)	Description
*GAP Description	Description of the gap
Analysis section	
Technical Effort	Technical effort required for the gap analysis
Est. Effort (hrs)	Estimated effort (in hours) required for the gap analysis
Est. Completion Date	Estimated completion date of the gap analysis
Project Team	Project team assigned
Analysis	Gap analysis solution details
a. The Summary section is not expanded in "Figure B-6. OraApps GAP Analysis Request creation" .	

OraApps Interface Request Request Type

The following sections provide a description, configuration consideration, and field descriptions for the OraApps Interface Request request type.

Description

An Oracle E-Business Suite instance might have to use data from other ERP systems as well as third-party software for different functionality. Interfaces must be created that allow data to be extracted from other software programs to populate Oracle tables, and export data from Oracle to other software programs.

The OraApps Interface Request request type initiates the process of requesting a new interface for an Oracle system. This request type works with the OraApps Interfaces Process workflow.

For a description of request processes, see ["Overview of the Request Process" on page 207](#).

Configuration Considerations

The OraApps Interfaces Process workflow should be the default workflow and the only allowed workflow assigned to the OraApps Interface Request request type.

For more information about this workflow, see ["OraApps Interfaces Process Workflow" on page 259](#).

Field Descriptions

" [Figure B-7. OraApps Interface Request creation](#) " shows the window for creating an OraApps Interface Request based on the provided request type. "[Table B-8. OraApps Interface Request request type fields](#)" provides field descriptions for the request type.

Figure B-7. OraApps Interface Request creation

Table B-8. OraApps Interface Request request type fields

Field Name (*Required)	Description
Summary section^a	
*Workflow	OraApps Interfaces Process
Interfaces Details section	
*Interfaces Name	Name of the requested interface
*Requested By	Date by which the interface must be completed

Table B-8. OraApps Interface Request request type fields, continued

Field Name (*Required)	Description
Business Area	Business area requesting the interface
Application	Application under the business area requesting the interface
*Description Detail	Detailed description of the interface
External System Name	Name of the external system
External System Owner	Owner of the external system
Update Frequency	How frequently the interface is updated or run
Interface Type	Type of interface
Analysis section	
Technical Effort	Technical effort required for the interface analysis
Est. Effort (hrs)	Estimated effort (in hours) required for the interface analysis
Est. Completion Date	Estimated completion date of the interface generation
Project Team	Project team assigned
Analysis	Analysis for the interface generation
a. The Summary section is not expanded in " Figure B-7. OraApps Interface Request creation ".	
a. The Summary section is not expanded in " Field Descriptions ".	

OraApps Report Request Request Type

The following sections provide a description, configuration consideration, and field descriptions for the OraApps Report Request request type.

Description

Although Oracle E-Business Suite provides many reports, these standard reports are often insufficient to meet a company's business needs. For example, it could be important to include information captured in a Descriptive Flexfield or to report on transactional data based on company-specific criteria. When a report does not currently exist that can provide the information that is required, a new report must be created.

The OraApps Report Request request type initiates the process of developing a new or modified report for an Oracle system. This request type works with the OraApps Reports Process workflow.

For a description of request processes, see ["Overview of the Request Process" on page 207](#).

Configuration Considerations

The OraApps Reports Process workflow should be the default and the only allowed workflow assigned to the OraApps Report Request request type.

For more information about this workflow, see ["OraApps Reports Process Workflow" on page 267](#).

Field Descriptions

["Figure B-8. OraApps Report Request creation"](#) shows the window for creating an OraApps Enhancement Request based on the provided request type. ["Table B-9. OraApps Report Request request type fields" on the next page](#) provides field descriptions for the request type.

Figure B-8. OraApps Report Request creation

The screenshot shows a web-based form titled "Create New OraApps Report Request". On the left, there is a "Jump To" sidebar with a list of sections: Summary, Report Details, Analysis, Notes, and References. The main content area is divided into several sections. At the top, there is a "Summary" section with a plus sign. Below it is the "Report Details" section, which contains the following fields: "Report Name" (text input), "Requested By" (text input), "New Report" (radio buttons for Yes and No, with No selected), "Business Area" (dropdown menu), "Application" (dropdown menu), and "Description Detail" (text area). Below the "Report Details" section is the "Analysis" section, which contains: "Technical Effort" (dropdown menu), "Est. Effort (hrs)" (text input), "Est. Completion Date" (text input), "Project Team" (dropdown menu), and "Analysis" (text area). At the bottom of the form, there are sections for "Notes" and "References", both with plus signs.

Table B-9. OraApps Report Request request type fields

Field Name (*Required)	Description
Summary section^a	
*Workflow	OraApps Reports Process
Report Details section	
*Report Name	Name of the requested report
*Requested By	Date by which the report is required to be completed
*Detailed Description	Detailed description of the report
*New Report	Whether the report requested is new (Yes) or a modification (No)
Business Area	Business area requesting the report
Application	Application under the business area requesting the report
*Description Detail	Detailed description of the report
Analysis section	
Technical Effort	Technical effort required for the enhancement
Est. Effort (hrs)	Estimated effort (in hours) required for the enhancement
Est. Completion Date	Estimated completion date
Project Team	Project team assigned
Analysis	Details of the enhancement analysis
a. The Summary section is not expanded in "Figure B-8. OraApps Report Request creation" .	

OraApps Setup Change Request Request Type

The following sections provide a description, configuration consideration, and field descriptions for the OraApps Setup Change Request request type.

Description

Oracle systems use a number of different application setups to satisfy their business requirements. These application setups (referred to as “setup changes”) include modules

such as Accounting Periods, Asset Categories, and Inventory Orgs. Each environment and operating unit has its own set of modules, and needs its own setup changes.

The OraApps Setup Change Request request type initiates the process of requesting a new setup for an Oracle system. This request type works with the OraApps Setup Change Process workflow.

For a description of request processes, see ["Overview of the Request Process" on page 207](#).

Configuration Considerations

The OraApps Setup Change Process workflow should be the default and the only allowed workflow assigned to the OraApps Setup Change Request request type.

For more information about this workflow, see ["OraApps Setup Change Process Workflow" on page 270](#).

Field Descriptions

["Figure B-9. OraApps Setup Change Request creation"](#) shows the window for creating an OraApps Setup Change Request based on the provided request type. ["Table B-10. OraApps Setup Change Request request type fields"](#) provides field descriptions for the request type.

Figure B-9. OraApps Setup Change Request creation

Create New OraApps Setup Change Request

Jump To
Expand All | Collapse All

- Summary
- Setup Details
- Analysis
- Notes
- References

+ Summary

- Setup Details

* Setup Name:

Business Area: Application:

* Requested By:

* Setup Description:

* Environment Name:

Operating Unit:

- Analysis

Technical Effort: Est. Effort (hrs):

Est. Completion Date: Project Team:

Dependencies:

+ Notes

+ References

Table B-10. OraApps Setup Change Request request type fields

Field Name (*Required)	Description
Summary section^a	
*Workflow	OraApps setup change Process
Setup Change Details section	
*Setup Name	Name of the requested setup
Business Area	Business area requesting the setup change
Application	Application under the business area requesting the setup change
*Requested By	Required date of completion for the setup change
*Setup Description	Detailed description of the setup change
*Environment Name	Environment to which setup changes are made
Operating Unit	Operating unit to which setup changes are made

Table B-10. OraApps Setup Change Request request type fields, continued

Field Name (*Required)	Description
Analysis section	
Technical Effort	Technical effort required for the setup change
Est. Effort (hrs)	Estimated effort in hours required for the setup change
Est. Completion Date	Estimated completion date
Project Team	Project team assigned
Dependencies	Dependencies for the setup changes
a. The Summary section is not expanded in "Figure B-9. OraApps Setup Change Request creation" .	

OraApps Status Update Request

The following sections provide a description, configuration consideration, and field descriptions for the OraApps Status Update Request request type.

Description

Many Oracle organizations use manual status reports to gather information on project status, exceptions, and risks. However, the process can be inefficient due to the time and resources spent on updating and consolidating information, and the need for managers to follow up with group members regarding task completion. Reviewers might not know when status reports are ready for review, who to contact, or how to access the most current versions of the reports.

The OraApps Status Update Request request type automates the process of gathering and reviewing status information. This request type allows managers to request information, automatically notifies group members of requirements for information, and monitors task progress to support timely completion. When tasks for completing a report are done, all reviewers are automatically notified. All involved parties have access to the latest version of the report.

This request type works with the OraApps Status Update Request workflow.

For a description of request processes, see ["Overview of the Request Process" on page 207](#).

Configuration Considerations

The OraApps Status Update Request workflow should be the default and the only allowed workflow assigned to the OraApps Status Update Request request type.

Decide which subteams must produce status reports, and then do the following:

- Create new **Status** and **Team Lead** fields for the new teams that are not predefined.
- Delete the **Status** and **Team Lead** fields that are not relevant to your project.

Decide whether the default for each Status field should be **Requested** or **Not Requested**.

If you have already identified the team leads for the project, make them the default values in the **Team Lead** fields.

For more information about the workflow associated with the OraApps Status Update Request request type, see "[OraApps Status Update Request Workflow](#)" on page 271.

Field Descriptions

"[Figure B-10. OraApps Status Update Request creation](#)" shows the window for creating an OraApps Status Update Request based on the provided request type "[Table B-11. OraApps Status Update Request request type fields](#)" provides field descriptions for the request type.

Figure B-10. OraApps Status Update Request creation

Table B-11. OraApps Status Update Request request type fields

Field Name (*Required)	Description
Summary section ^a	

Table B-11. OraApps Status Update Request request type fields, continued

Field Name (*Required)	Description
*Workflow	OraApps setup change Process
Status Request section	
Distribution Team Lead	User name of the person who will provide the distribution team report. Required if Distribution Status is set to Requested .
*Finance Status	Option to ask for a status report from the finance team (Requested or Not Requested). Requested activates the finance team branch in the business process workflow, prompting for and collecting the finance team status report.
Finance Team Lead	User name of the person who will provide the finance team report. Required if Finance Status is set to Requested .
*Mfg Status	Option to ask for a status report from the manufacturing team (Requested or Not Requested). Requested activates the manufacturing team branch in the business process workflow, prompting for and collecting the manufacturing team status report.
Mfg Team Lead	User name of the person who will provide the manufacturing team report. Required if Manufacturing Status is set to Requested .
*CRM Status	Option to ask for a status report from the CRM team (Requested or Not Requested). Requested activates the CRM team branch in the business process workflow, prompting for and collecting the CRM team status report.
CRM Team Lead	User name of the person who will provide the CRM team report. Required if CRM Status is set to Requested .
*Conversions Status	Option to ask for a status report from the conversion team (Requested or Not Requested). Requested activates the conversion team branch in the business process workflow, prompting for and collecting the conversion team status report.

Table B-11. OraApps Status Update Request request type fields, continued

Field Name (*Required)	Description
Conversions Team Lead	User name of the person who will provide the conversions team report. Required if Conversions Status is set to Requested .
*Interfaces Status	Option to ask for a status report from the interfaces team (Requested or Not Requested). Requested activates the interfaces team branch in the business process workflow, prompting for and collecting the interfaces team status report.
Interfaces Team Lead	User name of the person who will provide the interfaces team report. Required if Interfaces Status is set to Requested .
*Reports Status	User name of the person who will provide the reports team report. Required if Reports Status is set to Requested .
*Testing Status	Option to ask for a status report from the testing team (Requested or Not Requested). Requested activates the testing team branch in the business process workflow, prompting for and collecting the testing team status report.
Testing Team Lead	User name of the person who will provide the testing team report. Required if Testing Status is set to Requested .
a. The Summary section is not expanded in "Figure B-10. OraApps Status Update Request creation" .	

Appendix C: Workflows

This section provides reference information about the Oracle-specific workflows provided in the Extension. These workflows are listed and defined in Table C-1.

Some workflows are associated with object types and Deployment Management, and include package execution. Other workflows are associated with request types and Demand Management, and describe decision-making processes. For more information about Deployment Management and Demand Management, see the *Deployment Management Configuration Guide* and the *Demand Management Configuration Guide*.

In the description of each workflow, the associated object type or request type, if any, is identified. For more information about how each workflow and the associated object type or request type work together, see the referenced section at the end of the description.

Migration and compilation of entities using workflows are driven by commands included within the entities or workflow steps. For more information about commands in the PPM Center environment, see the *Commands, Tokens, and Validations Guide and Reference*.

Reference Request Types

Reference workflows cannot be edited, but you can copy and rename them and edit the copies to meet your needs. You can also use existing non-reference workflows as is or configure them further to meet your needs.

List of Request Types

"[Table C-1. Workflows included in the Extension](#)" lists the workflows included in the Extension. Each is described in subsequent sections.

Table C-1. Workflows included in the Extension

Workflow Name	Product Scope	Workflow
OraApps Application Issue	Demand Management	Evaluates requests and routes them according to priority, keeping requestors informed of status and prompting for action as needed.
OraApps Cloning Process	Demand Management	Initiates and tracks a request to clone an Oracle E-Business Suite instance.
OraAppsConversion Process	Demand Management	Helps to automates conversion and importing of data from legacy systems or third-party software into Oracle systems.

Table C-1. Workflows included in the Extension, continued

Workflow Name	Product Scope	Workflow
OraApps Customization/ Configuration Deployment	Deployment Management	Helps to automate deployment and review of customizations and configurations from development to test to production.
OraApps Design & Development	Demand Management	Manages functional analysis, technical design, and reviews, prompting for action as needed.
OraApps GAP Analysis Process	Demand Management	Provides a process to analyze and address gaps between existing company processes and those of Oracle E-Business Suite.
OraApps Interface Process	Demand Management	Provides a process to design and develop interfaces to import and export data between Oracle tables and other software programs.
OraApps Patch Deployment	Deployment Management	Provides a process to review and apply Oracle patches in “VANILLA,” development, test, and production environments. Calls the OraApps Patch Deployment Subworkflow and the OraApps Patch Data Capture Subworkflow.
OraApps Report Process	Deployment Management	Provides a process for designing and developing new or modified reports for an Oracle system.
OraApps Setup Change Process	Demand Management	Provides a process for requesting a new setup for an Oracle system.
OraApps Status Update Request	Demand Management	Provides a process for gathering, coordinating, and reviewing status reports.
OraApps11i Cloning	Deployment Management	Helps to clone an instance of Oracle E-Business Suite by automating major steps in the process.

General Configuration Considerations

Consider the following before you use or configure the workflows provided in the Extension:

- You must enable workflows before they can be used.
- Review the following areas to determine whether configurations are required to meet the needs of your organization:
 - The list of notification recipients and text.
 - The list of security groups. (You should probably configure security groups for each workflow step.)
 - The granularity level of steps. (Consider whether additional steps are needed or whether a subworkflow is needed.)

For information about configuring workflows, see ["Configuring Workflows" on page 55](#).

OraApps Application Issue Workflow

During an Oracle implementation or upgrade, issues can arise quickly. Unless these issues are resolved efficiently, they can grow into a backlog that might stall the project. Issues can come from a number of different sources and can be recorded in various locations. A process must exist to consolidate Oracle-related issues so that project members can create and review issues systematically.

The OraApps Application Issue workflow initially evaluates requests according to their priority level. Requests designated as critical are routed to project managers. All other requests are routed to the initial review step. During these initial reviews and analyses, requestors are kept informed of status through notifications. Anyone needing to take action is prompted to do so.

If detailed analysis determines that there is a system issue, the issue is automatically triaged to either an external Oracle bug resolution process, or an internal justification, design, and development process.

The external process includes actions such as contacting Oracle, opening a TAR, and getting an Oracle patch. Visibility is maintained through each of these steps, with clear responsibility as to who must perform each action.

The internal process can include justification and sign-off steps if the issue involves a change in scope, or merely following the normal design and development process for changes with limited impact. Both the internal and external processes eventually converge into the automated deployment process that uses Deployment Management, leveraging additional content included in the Extension.

This workflow works in conjunction with the OraApps Application Issue request type.

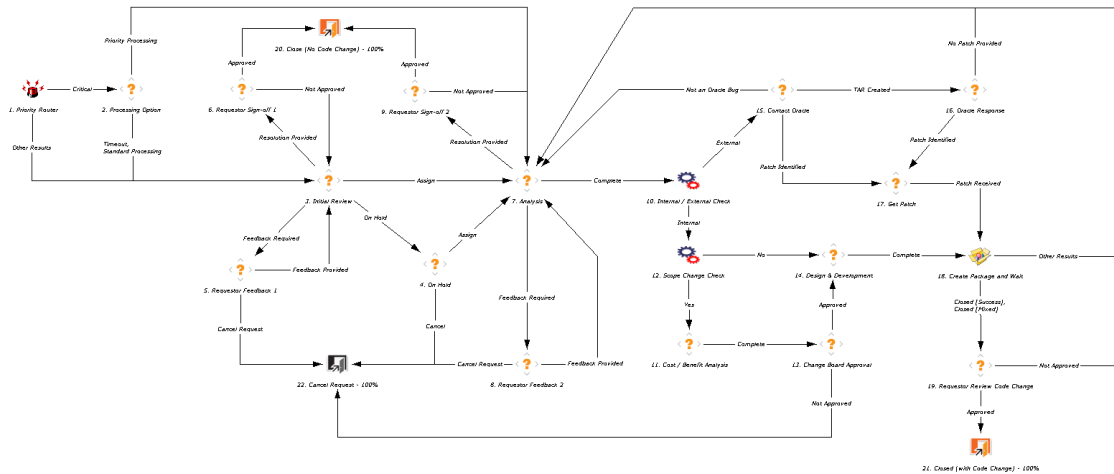
For a description of request processes, see ["Overview of the Request Process" on page 207](#).

For general configuration considerations for workflows, see ["General Configuration Considerations"](#) on the previous page.

Workflow Diagram and Step Descriptions

"[Figure C-1. OraApps Application Issue workflow](#)" shows the workflow.

Figure C-1. OraApps Application Issue workflow



The workflow steps are described as follows:

1. **Priority Router.** Once an issue is submitted, the priority router automatically routes the issue based on its priority. Critical issues are routed so that the project managers can decide on the processing option. All other issues are routed to the Initial Review step.
2. **Processing Option.** Project managers decide the processing option for critical issues. They can either decide on priority processing, in which case they assign a resource for analysis, or they can decide on the standard processing, in which case the track owners make a decision on the issue at the Initial Review step.
3. **Initial Review.** At the Initial Review step, the track owners review the issue. They might ask for more information from the requester or provide a resolution to the requester. Using their judgment and available resource capacity, the track owners can also keep non-critical issues on hold until the next development cycle.
4. **On Hold.** Project managers and track owners can act on the issue and either cancel it or assign it for analysis.
5. **Requestor Feedback 1.** The requester can provide more information on the issue when track owners request it.
6. **Requestor Sign-off 1.** The requester can accept or disapprove the resolution provided.

7. **Analysis.** The business analyst performs a detailed review of the request and might make one more attempt at a resolution without a system change. If a resolution cannot be achieved, then it is assumed that a code or configuration change is required. The analyst determines whether this change is within the current project scope and whether the change is external (a change to the standard Oracle Application code) or internal (a change to a customization or configuration), and moves the issue forward. When analysis is complete, the workflow automatically routes the request to the Internal / External Check step. There, if the issue requires an external change, the external vendor is contacted, otherwise the issue must go through internal approval and development.
8. **Requestor Feedback 2.** The requester can provide feedback to the business analyst assigned to the issue.
9. **Requestor Sign-off 2.** The requester can accept or disapprove the resolution provided by the business analyst analyzing the issue.
10. **Internal / External Check.** At this automatic step after analysis is complete, the request is routed according to whether the issue is external or internal. An internal issue is routed according to whether or not it is a scope change.
11. **Cost / Benefit Analysis.** A change that is out of the current project scope is analyzed for estimated effort as well the quantified business benefit. Once the justification information is gathered, the issue is passed on for approval. Usually, there is a specific group (here called the Change Board) that approves scope changes. This group can look at all issues currently pending review, look at the details on the request, and make a decision on the change.
12. **Scope Change Check.** This step determines whether the change is out of the project scope. If the change is out of scope, the workflow proceeds to the Cost / Benefit Analysis step. If the change is in scope, the workflow proceeds to the Design & Development step.
13. **Change Board Approval.** The Change Board can approve the change or reject it. Rejected changes are cancelled. Approved changes proceed to design and development.
14. **Design & Development.** This is a single step to indicate the full design and development process for the given change. At this step, a blocking request reference can be added to the issue, linking the issue to a request that models the full development process.
15. **Contact Oracle.** If the issue is external, then Oracle Support must be contacted. A new TAR might be created, or an existing patch might be located, or Oracle might indicate that the problem is not a bug. If a TAR is created, the TAR number is stored in the issue (in the TAR # field) and the issue waits for Oracle's response to the TAR. If an existing patch is found, the issue can go directly to the Get Patch step. If the problem is deemed not to be a bug, the issue returns to the analyst for further review.
16. **Oracle Response.** After the TAR has been created, Oracle provides a patch, commits to resolving the TAR in a future release, or indicates that the TAR will not be resolved. In the last two cases, no patch is provided, and the issue must go back to

the business analyst. At that point, the analyst might determine that an internal workaround or change must be made or that the issue cannot be resolved.

17. **Get Patch.** If a patch is provided, it must be downloaded from Oracle Support and placed in the appropriate staging area.
18. **Create Package and Wait.** Once the development is done or once the Oracle patch is received, the change must go through the standard deployment cycle. For this, a new package is generated with either the Oracle Patch Deployment or the Customization/Configuration Deployment workflow. The developer then adds the appropriate package lines and kicks off the deployment process.
19. **Requestor Review Code Change.** Once the package has completed the entire deployment process, the original requestor is notified and can approve the result (closing the issue) or set the result to not approved (sending the issue back to the business analyst).
20. **Close (No Code Change).** If the Requestor Sign-off 1 step or the Requestor Sign-off 2 step is an approval, the issue closes with no code change.
21. **Closed (with Code Change).** An approved issue closes with success.
22. **Cancel Request.** If the Change Board does not approve the request in the Change Board Approval step, the request gets cancelled.

Workflow Steps with Predefined Security

"[Table C-2. OraApps Application Issue workflow predefined security](#)" shows the steps of the workflow that have predefined security.

Table C-2. OraApps Application Issue workflow predefined security

Step	Step Name	Security Setting
2	Processing Option	OA - Management
3	Initial Review	OA - Track Owner
4	On Hold	OA - Management OA - Track Owner
7	Analysis	OA - Developer
11	Cost / Benefit Analysis	OA - Developer
13	Change Board Approval	OA - Management
14	Design & Development	OA - Developer
15	Contact Oracle	OA - Developer OA - Oracle Patch Admin

Table C-2. OraApps Application Issue workflow predefined security, continued

Step	Step Name	Security Setting
16	Oracle Response	OA - Developer OA - Oracle Patch Admin
17	Get Patch	OA - Developer OA - Oracle Patch Admin
18	Create Package and Wait	OA - Developer

OraApps Cloning Process Workflow

The OraApps Cloning Process workflow streamlines the process of initiating and tracking a request to clone an Oracle E-Business Suite instance.

This workflow works in conjunction with the OraApps Cloning Request request type.

For a description of request processes, see ["Overview of the Request Process" on page 207](#).

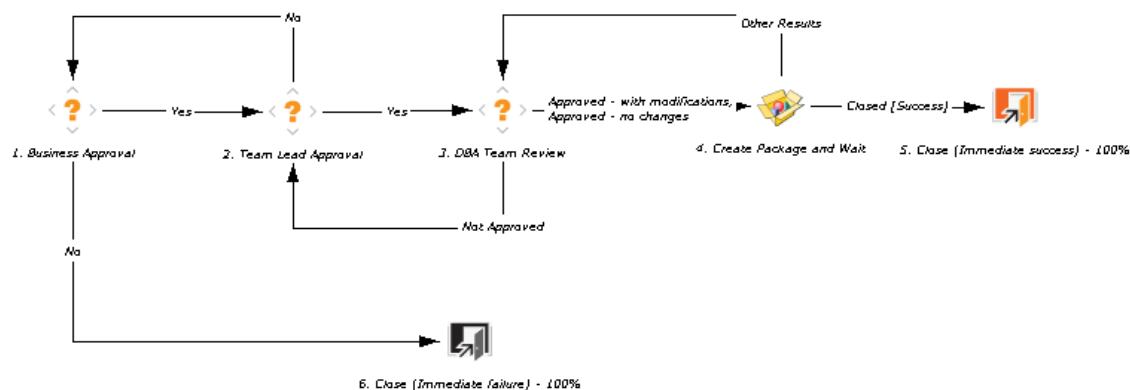
For general configuration considerations for workflows, see ["General Configuration Considerations" on page 239](#).

For more information about the request type associated with the OraApps Cloning Process workflow, see ["OraApps Cloning Request Request Type" on page 213](#).

Workflow Diagram and Step Descriptions

Figure C-2 shows the workflow.

Figure C-2. OraApps Cloning Process workflow



The workflow steps are described as follows:

1. **Business Approval.** Once submitted, the cloning request is reviewed by the business group.

2. **Team Lead Approval.** The team lead reviews the request. If approved, the request goes to the DBA team for review.
3. **DBA Team Review.** At this step, the DBA team analyzes the request for technical feasibility. If the request is approved, a package is created for the environment cloning of the Oracle application.
4. **Create Package and Wait.** The request waits at this step until the package completes the deployment.
5. **Close (Immediate success).** If the environment cloning is successful, the request closes out.
6. **Close (Immediate failure).** If the request does not obtain business approval in the Business Approval step, the request is closed with a status of failure.

OraApps Conversion Process Workflow

Organizations using Oracle E-Business Suite might have used other legacy systems for similar functionality in the past. The data from other systems must be transformed and imported into Oracle using interfaces, and the correct tables should be populated with this data. For example, an organization that decides to use Oracle CRM must obtain all the contact information from legacy systems and put the information into Oracle before starting to use Oracle for CRM.

Organizations using Oracle systems might have also used third-party software applications in the past. In order to be brought into Oracle, this data must go through conversion, as well. In both of these examples, data conversion is a one-time-only occurrence.

The OraApps Conversion Process workflow automates the conversion process.

This workflow works in conjunction with the OraApps Conversion Request request type.

For a description of request processes, see ["Overview of the Request Process" on page 207](#).

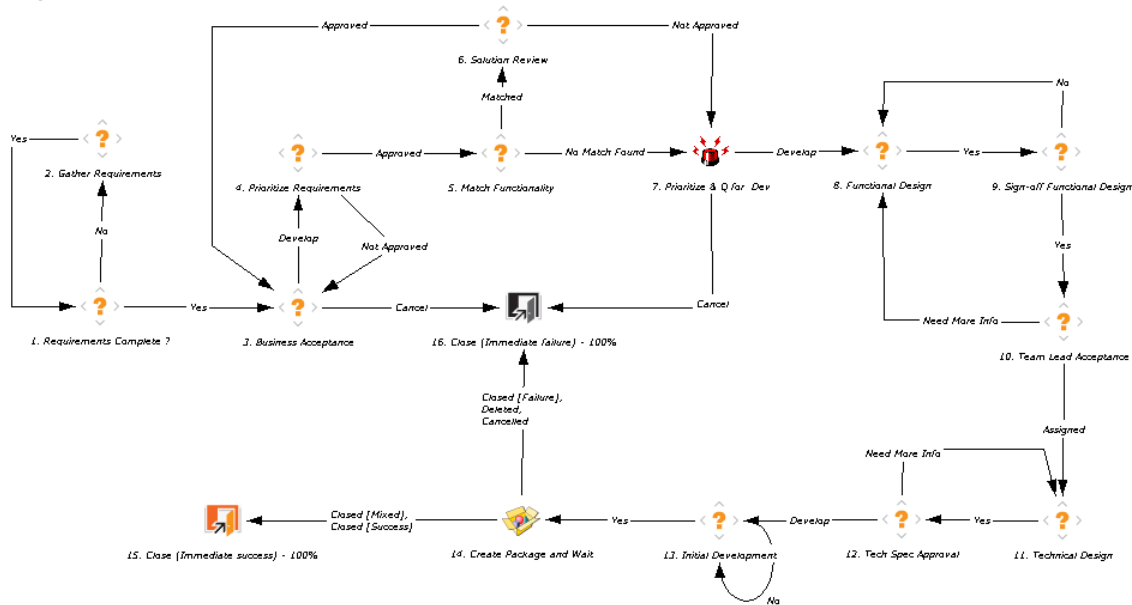
For general configuration considerations for workflows, see ["General Configuration Considerations" on page 239](#).

For more information about the request type associated with the OraApps Conversion Process workflow, see ["OraApps Conversion Request Request Type" on page 216](#).

Workflow Diagram and Step Descriptions

["Figure C-3. OraApps Conversion Process workflow"](#) shows the workflow.

Figure C-3. OraApps Conversion Process workflow



The workflow steps are described as follows:

1. **Requirements Complete?** The OraApps Conversion Request is submitted. Once submitted, the requirements are checked for completeness.
2. **Gather Requirements.** If requirements are unclear or more requirements are needed, the requestor is notified for more information. After the requirements are signed off, the request goes to the Business Acceptance step.
3. **Business Acceptance.** At this step, the business lead makes a decision to accept the request or ask for more information on the requirements.
4. **Prioritize Requirements.** If the requirements are accepted, they are prioritized based on business need and resource availability.
5. **Match Functionality.** The prioritized requirements are matched with existing programs. If a match is found, the request goes to the Solution Review step. If no match is found, the request goes to the Prioritize & Q for Dev step.
6. **Solution Review.** The solution review team analyzes and justifies the match. If a match is found, the solution review team approves the request and notifies the business lead. If the suggested program does not match the requirements, the team disapproves the request and it goes to the Prioritize & Q for Dev step for prioritization.
7. **Prioritize & Q for Dev.** The track owner prioritizes the request for development work.
8. **Functional Design.** The functional designs are created.
9. **Sign-off Functional Design.** The functional designs are reviewed and might be sent back if the reviewer needs more information about them. If approved, the request goes to the team lead for acceptance.
10. **Team Lead Acceptance.** The team lead might request more information about the functional designs, which might lead to changes to them. If so, the designs being

modified must be signed off again. When the team lead accepts the functional designs, a developer or team of developers is assigned.

11. **Technical Design.** The developer or team of developers creates technical designs and these designs are reviewed with the requirements.
12. **Tech Spec Approval.** During the technical approval, more information might be requested from the developer or the developer team, which might require them to modify the technical designs.
13. **Initial Development.** Upon technical approval, the initial development begins.
14. **Create Package and Wait.** Until the deployment is complete, the request waits at this step.
15. **Close (Immediate success).** When the conversion program is complete, the request closes out and the requestor is notified.
16. **Close (Immediate failure).** If the request is cancelled in the Business Acceptance step or the Prioritize & Q for Dev step, the request is closed with a status of failure.

Steps with Predefined Security

"[Table C-3. OraApps Conversion Process workflow predefined security](#)" shows the steps of the workflow that have predefined security.

Table C-3. OraApps Conversion Process workflow predefined security

Step	Step Name	Security Setting
3	Business Acceptance	OA - Business Owner
7	Prioritize & Q for Dev	OA - Track Owner
8	Functional Design	OA - Functional Designer
9	Sign-off Functional Design	OA - Functional Gap Reviewer
10	Team Lead Acceptance	OA - Track Owner
11	Technical Design	OA - Developer
12	Tech Spec Approval	OA - Technical Gap Reviewer

OraApps Customization/Configuration Deployment Workflow

It is crucial to the success of an Oracle E-Business Suite implementation, an upgrade, or ongoing production support that a process exist for managing configurations and customizations. After initial development, configurations and customizations must be efficiently deployed through the various testing environments and finally to a production instance. This process can involve multiple testers and environment gatekeepers, so

notifications for actions and for exceptions (such as failed tests and technical reviews) are important.

A key proven practice is to redeploy customizations first to the development environment before the formal unit test and before deployment to the QA environment. This accomplishes the following:

- It verifies that scripts can be rerun as they should, since they are being applied to an environment that should already have the changes.
- The developer should be the one initiating the automated deployment in the development environment. This individual can test the package for completeness and accuracy before sending it to the more controlled QA environment where someone else (a gatekeeper or operations person) initiates the deployment.
- There is better control of the changes being made to the development environment, giving visibility to the customizations and configurations in progress. This is very important when tracking down environment issues or when performing environment refreshes.

The OraApps Configuration/Customization Development workflow helps to automate this process.

The review steps incorporated near the end of the deployment cycle are a key component of this workflow. For Oracle E-Business Suite, the smallest configuration or customization can have enormous performance impact on the production system. This is especially true with custom reports. All configurations and customizations should go through a performance review before being deployed into the production environment. Because of the higher impact of database changes, changes at the database level should go through additional DBA review.

The “horseshoe” shape of this workflow is based on the concept that if a non-recoverable failure ever occurs in a deployment step or review, the package should go back to the developer for rework. If the rework is successful, the package must go through the entire deployment process again. The package should not go back to the step where the original failure occurred. In this way, all environments get a consistent picture of the customization or configuration.

Configuration Considerations

If you want additional object types to be reviewed by your DBA team, add them as transition values from the Check for DB Objects step to the DBA Review & Approval step.

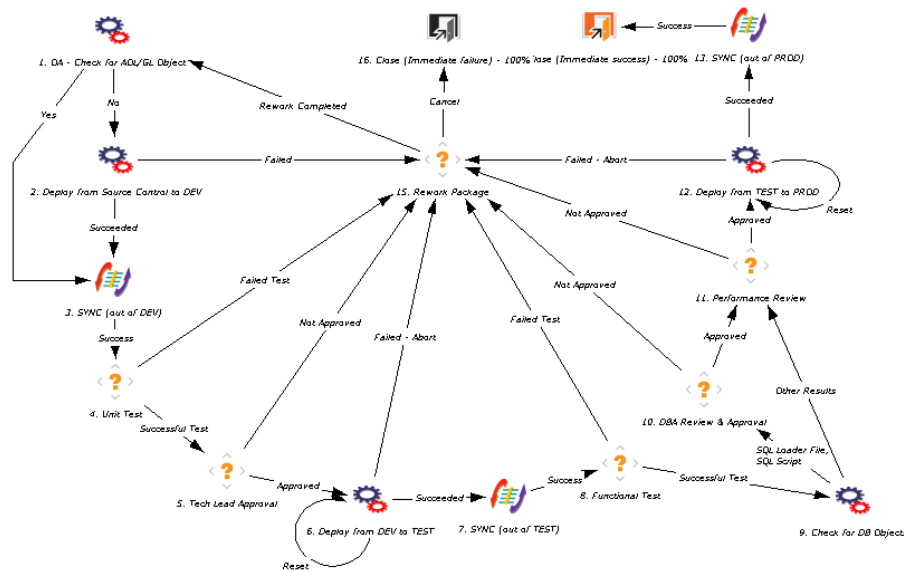
You might want to add more SYNC steps so that the entire package moves in unison for every step in the process. The workflow supplied by Micro Focus limits this synchronization to the deployment steps.

If you have additional environments in your deployment cycle, add steps similar to the Deploy from DEV to TEST step and the SYNC (out of TEST) step.

Workflow Diagram and Step Descriptions

"Figure C-4. OraApps Customization/Configuration Deployment workflow" shows the workflow.

Figure C-4. OraApps Customization/Configuration Deployment workflow



The workflow steps are described as follows:

1. **OA - Check for AOL/GL Object.** This is a automated token resolution step that checks to determine whether the object type on the package line is an entity that gets migrated by Object Migrator or GL Migrator. If the object type is an AOL or GL object, the Deploy from Source Control to DEV step is bypassed, since migrating the configuration from DEV back to DEV does not add any value to the process. For other object types, this step is performed, not bypassed, to make sure the scripts can be rerun and have been checked into version control.
2. **Deploy from Source Control to DEV.** This is an immediate execution step that deploys the package line into the DEV environment. The source environment should be set to the source control environment (or possibly the DEV environment), and the destination environment should be set to the development environment.
3. **SYNC (out of DEV).** This step causes all package lines to wait until all lines have been successfully deployed to DEV. The step makes sure that the package moves through the various environments and deployment stages as a single entity and not as individual lines.
4. **Unit Test.** Once formally deployed to DEV, the Customization/ Configuration is ready for unit testing. The assigned-to user for the package is notified of the deployment and prompted to perform the testing. Once the testing is done, the user provides the result for this step. If testing is successful, the package line goes forward. Otherwise, the line goes to the Rework Package step.

5. **Tech Lead Approval.** Once the package has passed unit test, it must be reviewed for adherence to technical standards and technical correctness. This is usually done by an architecture group or a technical team lead. The appropriate group is notified and should review the package. This might involve examining the actual code and scripts being deployed. In this step, reviewers and approvers provide any comments as notes on the package and provide the result of the review/approval. If approved, the line goes forward. Otherwise, the line goes to the Rework Package step.
6. **Deploy from DEV to TEST.** This is an execution step that deploys the line from the DEV environment to the TEST environment. The developer or operations user must tell the system to perform the deployment. The line does not automatically deploy as soon as the step becomes eligible. The source environment should be set to the development environment, and the destination environment should be set to the testing environment. If the deployment fails, the execution logs should be investigated. If the problem can be resolved without changing the original code (for example, by adding a tablespace to the database), the step should be reset and redeployed. If changes to the code must be made, the line goes to the Rework Package step.
7. **SYNC (out of TEST).** This step causes package lines to wait until all lines have been successfully deployed to TEST.
8. **Functional Test.** Once formally deployed to TEST, the Customization/ Configuration is ready for QA testing. The QA team is notified of the deployment and prompted to perform the testing. Once the testing is done, the testers provide the result for this step. If testing is successful, the package line goes forward. Otherwise, the line goes to the Rework Package step.
9. **Check for DB Objects.** This is an automatic token resolution that checks the object type for the package line. If the object type is related to the database, the line should be reviewed by a DBA, since there is a database impact. If the object type is not related to the database, the line skips the DBA Review & Approval step and goes to the Performance Review step.
10. **DBA Review & Approval.** For database-related object types, the DBA team is notified to review the appropriate package line. This can include reviewing the actual code being deployed. In this step, reviewers and approvers provide any comments as notes on the package and provide the result of the review/approval. If approved, the line goes forward. Otherwise, the line goes to the Rework Package step.
11. **Performance Review.** Because of the significant performance impact that changes can have to a production environment, every package should go through a review by the designated performance architecture team. In this step, reviewers and approvers provide any comments as notes on the package and provide the result of the review/approval. If approved, the line goes forward. Otherwise, the line goes to the Rework Package step.
12. **Deploy from TEST to PROD.** This is an execution step that deploys the line from the TEST environment to the PROD environment. The operations user must tell the system to perform the deployment. The line does not automatically deploy as soon

as the step becomes eligible. The source environment should be set to the testing environment, and the destination environment should be set to the production environment (or the “golden instance” if the system has not gone live yet.) If the deployment fails, the execution logs should be investigated. If the problem can be resolved without changing the original code (for example, by adding a tablespace to the database), the step should be reset and redeployed. If changes to the code must be made, the line goes to the Rework Package step.

13. **SYNC (out of PROD).** This step causes all package lines to wait until all lines have been successfully deployed to PROD.
14. **Close (Immediate success).** Once all the lines have been successfully deployed to the PROD environment, the package is closed.
15. **Rework Package.** If there is any failure in deployment or review, the specific package line is sent back for rework. The original developer is notified of this action and should review the line to identify the problem. Once rework fixes the problem, the package line starts at the beginning of the deployment cycle, since the fix must be deployed through all the environments and reviewed again. If a fix cannot be devised, the line can be cancelled.
16. **Close (Immediate failure).** If a package rework is cancelled at the Rework Package step, the package is closed with a status of failure.

Steps with Predefined Security

"Table C-4. OraApps Customization/Configuration workflow predefined security" shows the steps of the workflow that have predefined security.

Table C-4. OraApps Customization/Configuration workflow predefined security

Step	Step Name	Security Setting
1	OA - Check for AOL/GL Object	Assigned To Group Assigned To User
2	Deploy from Source Control to DEV	Assigned To Group Created By User Assigned To User
4	Unit Test	Assigned To Group Assigned To User Created By User
5	Tech Lead Approval	Assigned To Group Assigned To User
6	Deploy from DEV to TEST	Assigned To Group Assigned To User

Table C-4. OraApps Customization/Configuration workflow predefined security, continued

Step	Step Name	Security Setting
8	Functional Test	Assigned To Group Assigned To User
5	Tech Lead Approval	Assigned To Group Assigned To User
9	Check for DB Objects	Assigned To Group Assigned To User
11	Performance Review	Assigned To Group Assigned To User
12	Deploy from TEST to PROD	Assigned To Group Assigned To User
14	Close (Immediate success)	Assigned To Group Assigned To User
15	Rework Package	Assigned To Group Assigned To User Created By User
16	Close (Immediate failure)	Assigned To Group Assigned To User

OraApps Design & Development Workflow

The design and development cycle requires a functional analysis and review followed by a technical design and review. After the functional and technical sign-off, the core development is done. Then the configuration or customization is deployed through the standard deployment cycle, during which there is testing by other parties and possible rework.

During the process, it is important that individuals be proactively prompted to action and reminded if the actions are not performed in the specified period of time. The inefficiencies in the design and development process often occur in the approval and review process, where it might be unclear who must perform each action and when the actions are performed. The OraApps Design & Development workflow has multiple notifications at each step to do the following:

- Inform the appropriate individuals when they must perform an action, such as reviewing a technical design.
- Remind the individuals of the pending actions until they actually perform the action.
- Let the people involved in the process know the outcomes of each action, especially if the outcome was not expected (a disapproval or a cancellation).

The workflow also integrates with the deployment process by spawning the deployment package, maintaining visibility to the progress of the package through the deployment cycle, and waiting for the package to be fully deployed.

This workflow works with the OraApps Design & Development request type.

For a description of request processes, see ["Overview of the Request Process" on page 207](#).

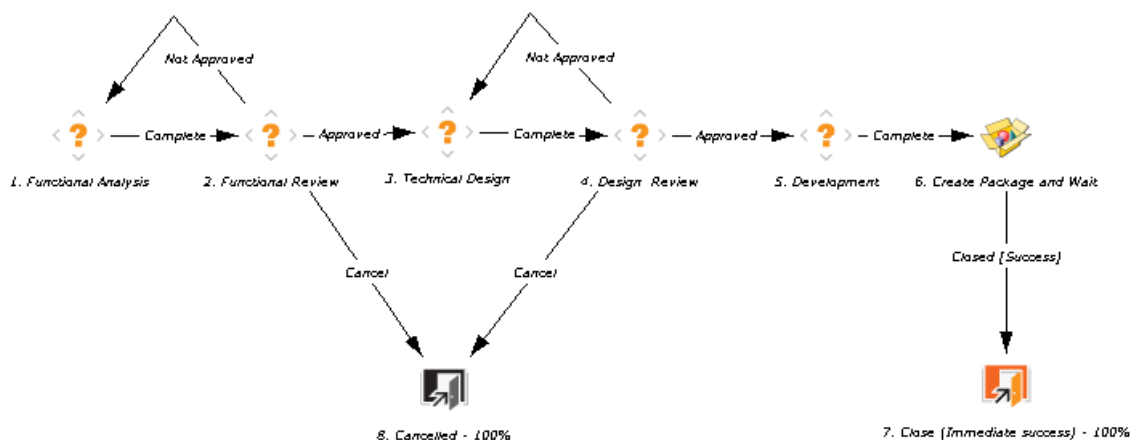
For general configuration considerations for workflows, see ["General Configuration Considerations" on page 239](#).

For more information about the request type associated with the OraApps Design & Development workflow, see ["OraApps Design & Development Request Type" on page 218](#).

Workflow Diagram and Step Descriptions

"[Figure C-5. OraApps Design & Development workflow](#)" shows the workflow.

Figure C-5. OraApps Design & Development workflow



The workflow steps are described as follows:

- 1. Functional Analysis.** At this step, a business analyst takes the requirement and creates a functional design to meet the requirement. This usually involves the creation of a functional design or high level design document, which should be attached to the request. The workflow supplied by Micro Focus assumes that the initial assigned-to user is the analyst and sends the notification for this step to that user.
- 2. Functional Review.** Once the functional design document is complete, it must be reviewed. Usually, a group of individuals review the document. This step notifies all the members of this group, signified in the system as the OA -Functional Gap Reviewer Security Group. If the request does not pass the review, this step notifies the creator of the request as well as the assigned-to user and sends the request back to the Functional Analysis step.

3. **Technical Design** and 4. **Design Review**. These steps are similar to the Functional Analysis and Functional Review steps but involve the detailed technical design and the developer and architect group instead of the analysts and functional review group.
4. **Development**. Once the design documents are complete and have been reviewed, the request goes into development. The assigned-to user now becomes the developer. Once initial development is done, this step is set to Complete to move the request into the deployment process.
5. **Create Package and Wait**. At this step, a package is automatically created and referenced to the request. The developer should then update the package with the appropriate package lines for this unit. Then the package should be released and should use the appropriate deployment workflow. Once the package is complete, the request is automatically closed.
6. **Close (Immediate success)**. Once the package is complete, the request is automatically closed with a status of success.
7. **Cancelled**. If the Functional Review step or the Design Review step results in disapproval, the request is cancelled.

Steps with Predefined Security

"[Table C-5. OraApps Design & Development workflow predefined security](#)" shows the steps of the workflow that have predefined security.

Table C-5. OraApps Design & Development workflow predefined security

Step	Step Name	Security Setting
1	Functional Analysis	OA - Functional Designers Assigned To Group Assigned To User
2	Functional Review	OA - Functional Gap Reviewer
3	Technical Design	OA - Developer
4	Design Review	OA - Technical Gap Reviewer
5	Development	OA - Developer
6	Create Package and Wait	OA - Developer

Table C-5. OraApps Design & Development workflow predefined security, continued

Step	Step Name	Security Setting
7	Close (Immediate success)	OA - Developer OA - Technical Gap Reviewer OA - Functional Gap Reviewer OA - Functional Designers
8	Cancelled	OA - Functional Gap Reviewer OA - Technical Gap Reviewer

OraApps Enhancement Process Workflow

When using Oracle E-Business Suite, users might find that the system does not meet all of their business needs. This might be due to shortcomings within Oracle E-Business Suite, or to unusual circumstances or needs within the company. Modifications to existing functionality, or even entirely new functionality, might be required in order to meet the company's business needs.

Proposed enhancements must go through a process of business and technical approval to verify that they are truly necessary, that they will function well in conjunction with Oracle E-Business Suite, and that they will meet the business requirements. The OraApps Enhancement Process workflow implements the enhancement process, working in conjunction with the OraApps Enhancement Request request type.

For a description of request processes, see ["Overview of the Request Process" on page 207](#).

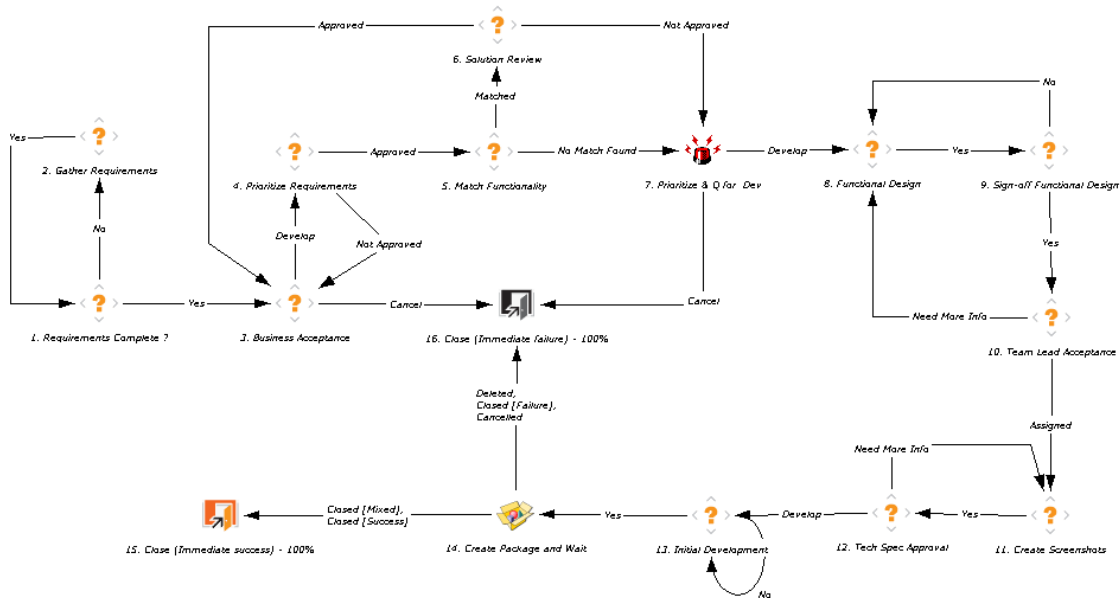
For general configuration considerations for workflows, see ["General Configuration Considerations" on page 239](#).

For more information about the request type associated with the OraApps Enhancement Process workflow, see ["OraApps Enhancement Request Request Type" on page 221](#).

Workflow Diagram and Step Descriptions

["Figure C-6. OraApps Enhancement Process workflow"](#) shows the workflow.

Figure C-6. OraApps Enhancement Process workflow



The workflow steps are described as follows:

1. **Requirements Complete?** The OraApps Conversion Request is submitted. Once submitted, the requirements are checked for completeness.
2. **Gather Requirements.** If requirements are unclear or more requirements are needed, the requestor is notified for more information. After the requirements are signed off, the request goes to the Business Acceptance step.
3. **Business Acceptance.** At this step, the business lead makes a decision to accept the request or ask for more information on the requirements.
4. **Prioritize Requirements.** If the requirements are accepted, they are prioritized based on business need and resource availability.
5. **Match Functionality.** The prioritized requirements are matched with existing programs. If a match is found, the request goes to the Solution Review step. If no match is found, the request goes to the Prioritize & Q for Dev step.
6. **Solution Review.** The solution review team analyzes and justifies the match. If a match is found, the solution review team approves the request and notifies the business lead. If the suggested program does not match the requirements, the team disapproves the request and it goes to the Prioritize & Q for Dev step for prioritization.
7. **Prioritize & Q for Dev.** The track owner prioritizes the request for development work.
8. **Functional Design.** The functional designs are created.
9. **Sign-off Functional Design.** The functional designs are reviewed and might be sent back if the reviewer needs more information about them. If approved, the request goes to the team lead for acceptance.
10. **Team Lead Acceptance.** The team lead might request more information about the functional designs, which might lead to changes to them. If so, the designs being modified must be signed off again.

11. **Create Screenshots.** When the team lead accepts the functional designs, a developer or team of developers is assigned. The developer or team of developers creates technical designs, including screenshots, mock-ups, and so on.
12. **Tech Spec Approval.** The designs are reviewed against the requirements, design standards, and so on. During the technical approval, more information might be requested from the developer or the developer team, which might require them to modify the technical designs.
13. **Initial Development.** Upon technical approval, the initial development begins.
14. **Create Package and Wait.** Until the deployment is complete, the request waits at this step.
15. **Close (Immediate success).** When the enhancement is complete, the request closes out and the requestor is notified.
16. **Close (Immediate failure).** If the request does not obtain business acceptance in the Business Acceptance step, if the track owner disapproves the request in the Prioritize & Q for Dev step, or if the request is cancelled at the Create Package and Wait step, the request is closed with a status of failure.

Steps with Predefined Security

"[Table C-6. OraApps Enhancement Process workflow predefined security](#)" shows the steps of the workflow that have predefined security.

Table C-6. OraApps Enhancement Process workflow predefined security

Step	Step Name	Security Setting
3	Business Acceptance	OA - Business Owner
7	Prioritize & Q for Dev	OA - Track Owner
8	Functional Design	OA - Functional Designers
9	Sign-off Functional Design	OA - Functional Gap Reviewer
10	Team Lead Acceptance	OA - Track Owner
11	Technical Design	OA - Developer
12	Tech Spec Approval	OA - Technical Gap Reviewer

OraApps GAP Analysis Process Workflow

An implementation or upgrade of Oracle E-Business Suite can mean significant changes to the company's business processes. The difference between the existing processes and those provided by the packaged application is referred to as a "gap." Stakeholders in an Oracle-related project must be able to request a "gap analysis" for specific Oracle functionality.

The OraApps GAP Analysis Process workflow implements the gap analysis process. This workflow operates in conjunction with the OraApps GAP Analysis request type.

For a description of request processes, see ["Overview of the Request Process" on page 207](#).

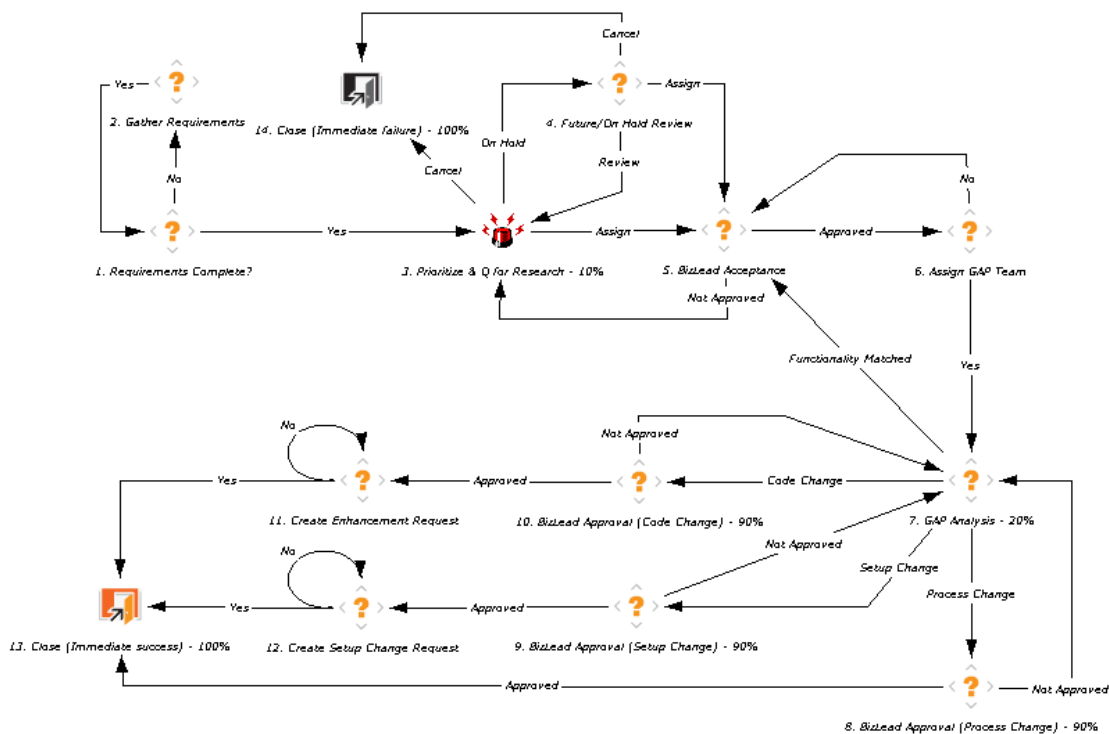
For general configuration considerations for workflows, see ["General Configuration Considerations" on page 239](#).

For more information about the request type associated with the OraApps GAP Analysis Process workflow, see ["OraApps GAP Analysis Request Request Type" on page 223](#).

Workflow Diagram and Step Descriptions

Figure C-7 shows the workflow.

Figure C-7. OraApps GAP Analysis Process workflow



The workflow steps are described as follows:

- 1. Requirements Complete?** The OraApps GAP Analysis Request is submitted. Once submitted, the requirements are checked for completeness.
- 2. Gather Requirements.** If requirements are unclear or more requirements are needed, the requestor is notified for more information. After the requirements are signed off, the request goes to the Business Acceptance step.
- 3. Prioritize & Q for Dev.** The manager prioritizes the request and assigns a business team lead to it. The request goes to the BizLead Acceptance step, or the manager

- puts the request on hold and it goes to the Future/On Hold Review step, or the manager cancels the request and it goes to the Close (Immediate failure) step.
4. **Future/On Hold Review.** The request is on hold, to be assigned or cancelled later.
 5. **BizLead Acceptance.** The business lead can approve the request and then assign a team for gap analysis. If the business lead rejects the request, it goes back to the manager.
 6. **Assign GAP Team.** The business lead assigns a team for gap analysis.
 7. **GAP Analysis.** If the gap analysis team can find matching functionality in Oracle, they can send this information back to the business lead. The analysis team could also determine that a code change, a setup change, or a process change is required to bridge the gap. The gap analysis team specifies all the details about the analysis and the time estimates for any changes that must be made.
 8. **BizLead Approval (Process Change).** The business lead approves the creation of a process change if the gap analysis team determines that a process change is required. The gap analysis request closes as soon as the business lead approves the process change.
 9. **BizLead Approval (Setup Change).** The business lead approves the creation of a setup change request if the gap analysis team determines that a setup change is required.
 10. **BizLead Approval (Code Change).** The business lead approves the creation of a code change request if the gap analysis team determines that a code change is required.
 11. **Create Enhancement Request.** An enhancement request is created. The gap analysis request closes after the enhancement request is created.
 12. **Create Setup Change Request.** A setup change request is created. The gap analysis request closes after the setup change request is created.
 13. **Close (Immediate success).** The gap analysis request closes after the enhancement request or setup change request is created.
 14. **Close (Immediate failure).** The manager can cancel a request during prioritization or after it is kept on hold for later review. Then the request is closed with failure.

Steps with Predefined Security

"[Table C-7. OraApps GAP Analysis Process workflow predefined security](#)" shows the steps of the workflow that have predefined security.

Table C-7. OraApps GAP Analysis Process workflow predefined security

Step	Step Name	Security Setting
3	Prioritize & Q for Research	OA - Management
5	BizLead Acceptance	OA - Business Owner

Table C-7. OraApps GAP Analysis Process workflow predefined security, continued

Step	Step Name	Security Setting
6	Assign GAP Team	OA - Business Owner
8	BizLead Approval (Process Change)	OA - Business Owner
9	BizLead Approval (Setup Change)	OA - Business Owner
10	BizLead Approval (Code Change)	OA - Business Owner

OraApps Interfaces Process Workflow

An Oracle system may have to use data from other ERP systems as well as third-party software for different functionality. Interfaces must be created that allow extraction of data from other software programs to populate Oracle tables, and exporting of data from Oracle to other software programs.

The OraApps Interfaces Process workflow implements the interfaces process. This workflow works with the OraApps Interface Request request type.

For a description of request processes, see ["Overview of the Request Process" on page 207](#).

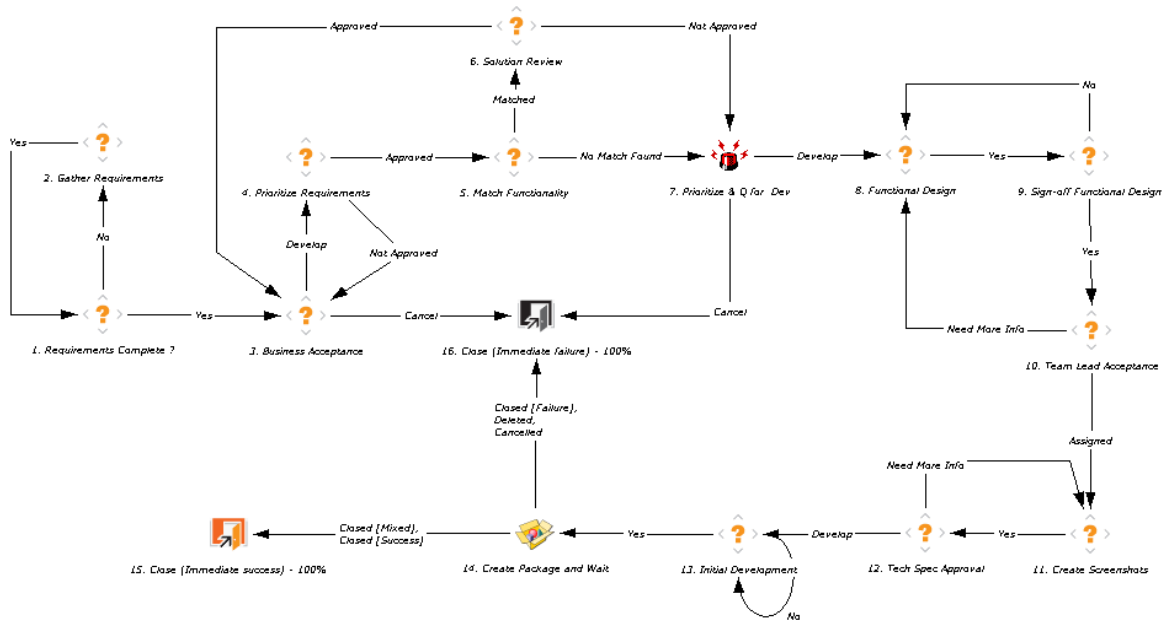
For general configuration considerations for workflows, see ["General Configuration Considerations" on page 239](#).

For more information about the request type associated with the OraApps Interfaces Process workflow, see ["OraApps Interface Request Request Type" on page 226](#).

Workflow Diagram and Step Descriptions

["Figure C-8. OraApps Interfaces Process workflow"](#) shows the workflow.

Figure C-8. OraApps Interfaces Process workflow



The workflow steps are described as follows:

1. **Requirements Complete?** The OraApps Interface Request is submitted. Once submitted, the requirements are checked for completeness.
2. **Gather Requirements.** If requirements are unclear or more requirements are needed, the requestor is notified for more information. After the requirements are signed off, the request goes to the Business Acceptance step.
3. **Business Acceptance.** At this step, the business lead makes a decision to accept the request or ask for more information on the requirements.
4. **Prioritize Requirements.** If the requirements are accepted, they are prioritized based on business need and resource availability.
5. **Match Functionality.** The prioritized requirements are compared with the existing functionality.
6. **Solution Review.** The solution review team analyzes cases where existing functionality was found to meet the business requirements. If the analysis shows that the functionality completely meets requirements, the solution review team approves the request and notifies the business lead. When the functionality does not match the requirements, the team disapproves the request and it goes to the Prioritize and Q for Dev step for prioritization.
7. **Prioritize & Q for Dev.** The track owner prioritizes the request for development work.
8. **Functional Design.** The functional design phase begins.
9. **Sign-off Functional Design.** The functional designs are reviewed and might be sent back if the reviewer needs more information about them. If approved, the request goes to the team lead for acceptance.
10. **Team Lead Acceptance.** The team lead might request more information about the functional designs, which might lead to changes to them. If so, the designs being

modified must be signed off again.

11. **Create Screenshots.** When the team lead accepts the functional designs, a developer or team of developers is assigned. The developer or team of developers creates technical designs, including screenshots, mock-ups, and so on.
12. **Tech Spec Approval.** The designs are reviewed against the requirements, design standards, and so on. During the technical approval, more information might be requested from the developer or the developer team, which might require them to modify the technical designs.
13. **Initial Development.** Upon technical approval, the initial development begins.
14. **Create Package and Wait.** Until the deployment is complete, the request waits at this step.
15. **Close (Immediate success).** When the interface is complete, the request closes out and the requestor is notified.
16. **Close (Immediate failure).** If the request does not obtain business acceptance in the Business Acceptance step, if the track owner disapproves the request in the Prioritize & Q for Dev step, or if the request is cancelled at the Create Package and Wait step, the request is closed with a status of failure.

Steps with Predefined Security

"[Table C-8. OraApps Interfaces Process workflow predefined security](#)" shows the steps of the workflow that have predefined security.

Table C-8. OraApps Interfaces Process workflow predefined security

Step	Step Name	Security Setting
3	Business Acceptance	OA - Business Owner
4	Prioritize Requirements	OA - Business Owner
7	Prioritize & Q for Dev	OA - Track Owner
9	Sign-off Functional Design	OA - Functional Gap Reviewer
10	Team Lead Acceptance	OA - Track Owner
11	Create Screenshots	OA - Developer
12	Tech Spec Approval	OA - Technical Gap Reviewer

OraApps Patch Deployment Workflow and Its Subworkflows

A significant number of patches are distributed for each release of Oracle E-Business Suite. The Extension provides automation to streamline and standardize the application

of Oracle patches to all your environments. Because of the volume and impact of the patches, they must be monitored and managed. The environments to which they are applied must be monitored and managed as well.

The OraApps Patch Deployment workflow provides a process for viewing and controlling your Oracle patch activities.

Once an Oracle patch has been identified and received from Oracle, it should go through a structured process of review and deployment to maintain visibility to the patch and reduce the risk of applying a bad patch to a critical environment.

Because of the complex nature of Oracle patches, it is important that each patch be reviewed for impact, pre- and post-patch application steps, and general relevancy to your specific environment. After the review process, the patch must be applied first to a “VANILLA” Oracle instance. This is an environment that does not include any in-house customization. This is a standard practice to isolate the Oracle patch in a clean environment and verify its correctness. If the patch works in this environment and not in the other instances, then the problem might be due to a customization or configuration. After the successful deployment to the VANILLA instance, the Oracle patch proceeds through the development, testing, and production environments, getting verified in each instance.

The layout of this workflow is based on the concept that if a non-recoverable failure ever occurs in a deployment or verification step, the Oracle patch should go back for review. This re-review determines the problem with the patch and, if the problem can be corrected, it moves the patch through the full deployment cycle. If the patch cannot be fixed, then the deployment is cancelled and Oracle Support should be contacted.

The OraApps Patch Deployment workflow calls the OraApps Patch Data Capture Subworkflow and the OraApps Patch Deployment Subworkflow at several steps.

Configuration Considerations

Oracle Release 11i and Release 12 provide the capability to combine multiple patches into a single patch that includes all the individual changes. This lets you reduce the number of post-application steps such as recompiling database objects or Oracle forms. If you plan to allow merging of individual Oracle patches, you must modify the workflow supplied by Micro Focus. This can involve significant changes in the process and the automation to account for the fact that a single deployment includes multiple individual patches. Contact Micro Focus Software Support. As necessary, see the Micro Focus Software Support Web site:

<https://softwaresupport.hpe.com>

A proven practice for the Oracle Patch Application process is to put all patches in a central staging area and deploy patches to the different environments from this staging area. This establishes a single repository for patch files and drivers.

Therefore, the source environment for each deployment step in the workflow should be set to the staging environment.

The Apply to <environment> steps and the Capture Patch (<environment>) steps should be set up in pairs, with the same destination environments specified.

For more information about configuring this workflow, see ["OraApps Patch Deployment Workflow and Its Subworkflows"](#) on page 153.

Workflow Diagram and Step Descriptions

["Figure C-9. OraApps Patch Deployment workflow"](#) shows the OraApps Patch Deployment workflow.

["Figure C-10. OraApps Patch Deployment Subworkflow"](#) shows the OraApps Patch Deployment Subworkflow.

["Figure C-11. OraApps Patch Data Capture Subworkflow"](#) shows the OraApps Patch Data Capture Subworkflow.

Figure C-9. OraApps Patch Deployment workflow

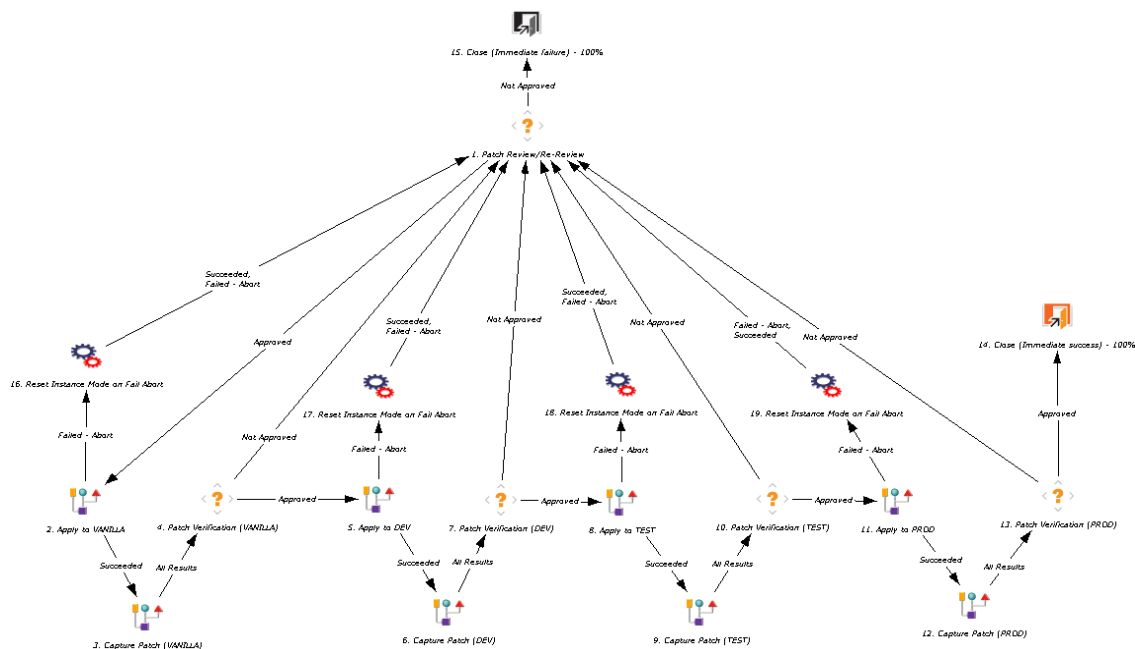


Figure C-10. OraApps Patch Deployment Subworkflow

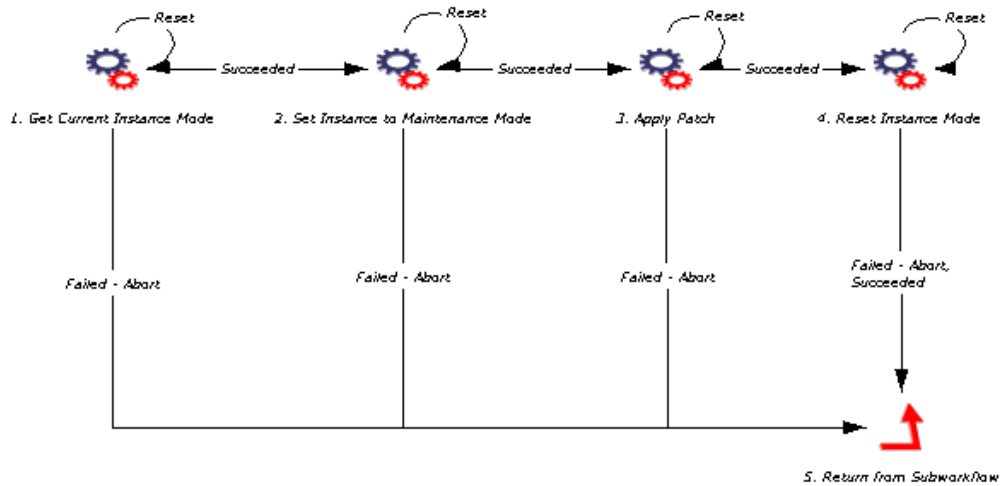
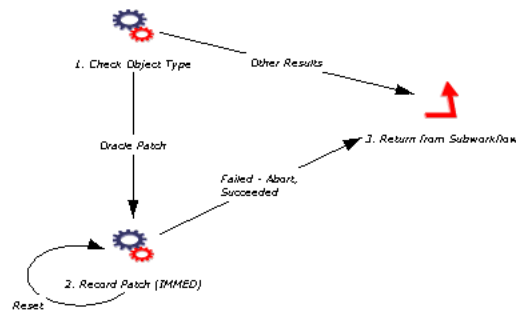


Figure C-11. OraApps Patch Data Capture Subworkflow



The OraApps Patch Deployment workflow steps are described as follows:

1. **Patch Review/Re-Review.** Once the package is initially submitted, the Oracle patch administration group is notified to review the patch. This typically includes reviewing the README file along with the driver files. The patch review looks at the impact of the patch, the pre- and post-patch application steps, and general relevancy to the specific environment or problem. If the patch is approved, it goes to the Apply to VANILLA step. This step also serves as the re-review step, in case the patch fails in deployment or verification during the deployment cycle. In this case, patch administrators should reexamine the patch to determine whether the problem can be fixed. For example, a patch application might fail because a prerequisite patch was not applied to the environment. If the problem can be fixed, the patch goes to the start of the deployment cycle (where the patch is applied to the VANILLA instance). Otherwise, the package line containing this patch is cancelled.
2. **Apply to VANILLA.** This is an execution step that applies the patch into the VANILLA environment. The developer or operations user must tell the system to perform the deployment once any required pre-application steps are performed. The execution step does not automatically deploy the patch as soon as the step becomes eligible. The source environment should be set to the staging environment and the destination environment should be set to the VANILLA environment. This step calls the OraApps Patch Deployment Subworkflow, which applies the patch to the

destination instance and returns to the workflow. If the Oracle E-Business Suite instance is at Release 11.5.10 or later, or Release 12, it must be in maintenance mode to apply the patch, so the subworkflow determines the mode of the destination instance, sets it to maintenance mode, applies the patch, resets the instance to its original mode, and returns to the workflow. If deployment fails, the execution logs should be investigated. These logs include links to the ADPATCH and ADADMIN logs, as well. If the problem can be resolved, the step should be reset and redeployed. If not, the patch should be sent back for re-review.

3. **Capture Patch (VANILLA).** Once the patch has been applied to the VANILLA instance, details of the contents of the patch are captured to allow analysis and detail reporting if the patch is eligible for this capture. This step calls the OraApps Patch Data Capture Subworkflow, which verifies that the object type is Oracle Patch, captures the patch details into the PPM Center patch repository, and returns to the workflow.
4. **Patch Verification (VANILLA).** Once formally applied to VANILLA, the patch is ready for testing. The assigned-to user for the package is notified of the deployment and prompted to perform the testing. Once the testing is done, the user provides the result for this step. If testing was successful, the package line goes forward. Otherwise, the line goes to the Patch Review/Re-Review step.
5. **Apply to DEV, 6. Capture Patch (DEV), and 7. Patch Verification (DEV).** These steps follow the same logic as the Apply to VANILLA, Capture Patch (VANILLA), and Patch Verification (VANILLA) steps 2-4, including calling the OraApps Patch Deployment Subworkflow and the OraApps Patch Data Capture Subworkflow, but the steps apply to the development environment. This is the first environment that includes customizations.
8. **Apply to TEST, 9. Capture Patch (TEST), and 10. Patch Verification (TEST).** These steps follow the same logic as the Apply to DEV, Capture Patch (DEV), and Patch Verification (DEV) steps 5-7, including calling the OraApps Patch Deployment Subworkflow and the OraApps Patch Data Capture Subworkflow, but the steps apply to the test environment.
11. **Apply to PROD, 12. Capture Patch (PROD), and 13. Patch Verification (PROD).** These steps follow the same logic as the Apply to TEST, Capture Patch (TEST), and Patch Verification (TEST) steps 8-10, including calling the OraApps Patch Deployment Subworkflow and the OraApps Patch Data Capture Subworkflow, but the steps apply to the production environment. This is the “live” production instance or the “golden” instance, if Oracle E-Business Suite has not been rolled out to production yet.
14. **Close (Immediate success).** If the Patch Verification (PROD) step results in approval, the deployment is closed with a status of success.
15. **Close (Immediate failure).** If the Patch Review/Re-Review step results in disapproval, the patch deployment is closed with status of failure.
16. **17, 18, and 19. Reset Instance Mode on Fail Abort.** If patch application failed at a preceding step, here you can abort the package line and reset the mode of the instance to its original state.

Steps with Predefined Security

"[Table C-9. OraApps Patch Deployment workflow predefined security](#)" shows the steps of the OraApps Patch Deployment workflow that have predefined security.

"[Table C-10. OraApps Patch Deployment Subworkflow predefined security](#)" shows the steps of the OraApps Patch Deployment Subworkflow that have predefined security.

"[Table C-11. OraApps Patch Data Capture Subworkflow predefined security](#)" shows the steps of the OraApps Patch Data Capture Subworkflow that have predefined security.

Table C-9. OraApps Patch Deployment workflow predefined security

Step	Step Name	Security Setting
1	Patch Review/Re-Review	OA - Oracle Patch Admin Assigned To Group Assigned To User Created By User
4	Patch Verification (VANILLA)	OA - Oracle Patch Admin Assigned To Group Assigned To User Created By User
7	Patch Verification (DEV)	OA - Oracle Patch Admin Assigned To Group Assigned To User Created By User
10	Patch Verification (TEST)	OA - Oracle Patch Admin Assigned To Group Assigned To User Created By User
13	Patch Verification (PROD)	OA - Oracle Patch Admin Assigned To Group Assigned To User Created By User

Table C-10. OraApps Patch Deployment Subworkflow predefined security

Step	Step Name	Security Setting
1	Get Current Instance Mode	OA - Oracle Patch Admin Assigned To Group Assigned To User Created By User
2	Set Instance to Maintenance Mode	OA - Oracle Patch Admin Assigned To Group Assigned To User Created By User
3	Apply Patch	OA - Oracle Patch Admin Assigned To Group Assigned To User Created By User
4	Reset Instance Mode	OA - Oracle Patch Admin Assigned To Group Assigned To User Created By User

Table C-11. OraApps Patch Data Capture Subworkflow predefined security

Step	Step Name	Security Setting
2	Record Patch (IMMED)	OA - Oracle Patch Admin Assigned To Group Assigned To User

OraApps Reports Process Workflow

Although Oracle E-Business Suite provides many reports, these standard reports are often insufficient to meet a company's business needs. For example, it might be important to include information captured in a Descriptive Flexfield, or to report on transactional data based on company-specific criteria. When a report does not currently exist that can provide the required information, a new report must be created. The OraApps Reports Process workflow implements the process of developing a new or

modified report for an Oracle system. This workflow works with the OraApps Report Request request type.

For a description of request processes, see ["Overview of the Request Process" on page 207](#).

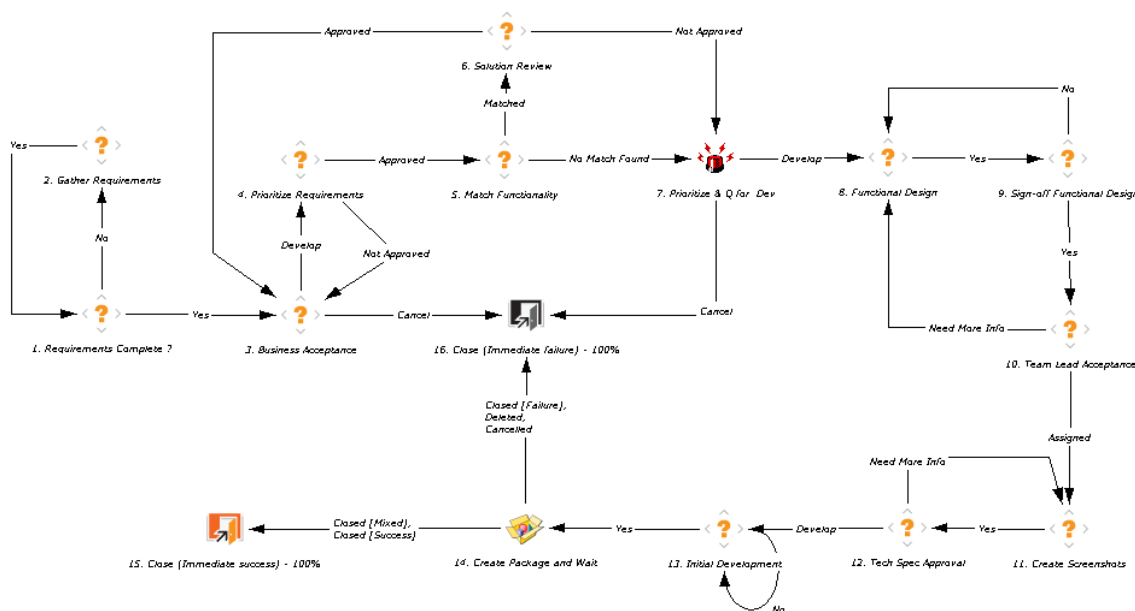
For general configuration considerations for workflows, see ["General Configuration Considerations" on page 239](#).

For more information about the request type associated with the OraApps Reports Process workflow, see ["OraApps Report Request Request Type" on page 228](#).

Workflow Diagram and Step Descriptions

["Workflow Diagram and Step Descriptions"](#) shows the workflow.

Figure C-12. OraApps Reports Process workflow



The workflow steps are described as follows:

- 1. Requirements Complete?** The OraApps Report Request is submitted. Once submitted, the requirements are checked for completeness.
- 2. Gather Requirements.** If requirements are unclear or more requirements are needed, the requestor is notified for more information. After the requirements are signed off, the request goes to the Business Acceptance step.
- 3. Business Acceptance.** At this step, the business lead makes a decision to accept the request or ask for more information on the requirements.
- 4. Prioritize Requirements.** If the requirements are accepted, they are prioritized based on business need and resource availability.

5. **Match Functionality.** The prioritized requirements are compared with the existing functionality.
6. **Solution Review.** The solution review team analyzes cases where existing functionality was found to meet the business requirements. If the analysis shows that the functionality completely meets requirements, the solution review team approves the request and notifies the business lead. When the functionality does not match the requirements, the team disapproves the request and it goes to the Prioritize and Q for Dev step for prioritization.
7. **Prioritize & Q for Dev.** The track owner prioritizes the request for development work.
8. **Functional Design.** The functional design phase begins.
9. **Sign-off Functional Design.** The functional designs are reviewed and might be sent back if the reviewer needs more information about them. If approved, the request goes to the team lead for acceptance.
10. **Team Lead Acceptance.** The team lead might request more information about the functional designs, which might lead to changes to them. If so, the designs being modified must be signed off again.
11. **Create Screenshots.** When the team lead accepts the functional designs, a developer or team of developers is assigned. The developer or team of developers creates technical designs, including screenshots, mock-ups, and so on.
12. **Tech Spec Approval.** The designs are reviewed against the requirements, design standards, and so on. During the technical approval, more information might be requested from the developer or the developer team, which might require them to modify the technical designs.
13. **Initial Development.** Upon technical approval, the initial development begins.
14. **Create Package and Wait.** Until the deployment is complete, the request waits at this step.
15. **Close (Immediate success).** When the enhancement is complete, the request closes out and the requestor is notified.
16. **Close (Immediate failure).** If the request does not obtain business acceptance in the Business Acceptance step, if the track owner disapproves the request in the Prioritize & Q for Dev step, or if the request is cancelled at the Create Package and Wait step, the request is closed with a status of failure.

Steps with Predefined Security

"[Table C-12. OraApps Reports Process workflow predefined security](#)" shows the steps of the workflow that have predefined security.

Table C-12. OraApps Reports Process workflow predefined security

Step	Step Name	Security Setting
3	Business Acceptance	OA - Business Owner

Table C-12. OraApps Reports Process workflow predefined security, continued

Step	Step Name	Security Setting
4	Prioritize Requirements	OA - Business Owner
7	Prioritize & Q for Dev	OA - Track Owner
9	Sign-off Functional Design	OA - Functional Gap Reviewer
10	Team Lead Acceptance	OA - Track Owner
11	Create Screenshots	OA - Developer
12	Tech Spec Approval	OA - Technical Gap Reviewer

OraApps Setup Change Process Workflow

Oracle systems use a number of different application setups to satisfy their business requirements. These application setups (referred to as “setup changes”) include modules such as Accounting Periods, Asset Categories, and Inventory Orgs. Each environment and operating unit has its own set of modules and needs its own setup changes.

The OraApps Setup Change Process workflow implements the process of requesting a new setup for an Oracle system. This workflow works with the OraApps Setup Change Request request type.

For a description of request processes, see ["Overview of the Request Process" on page 207](#).

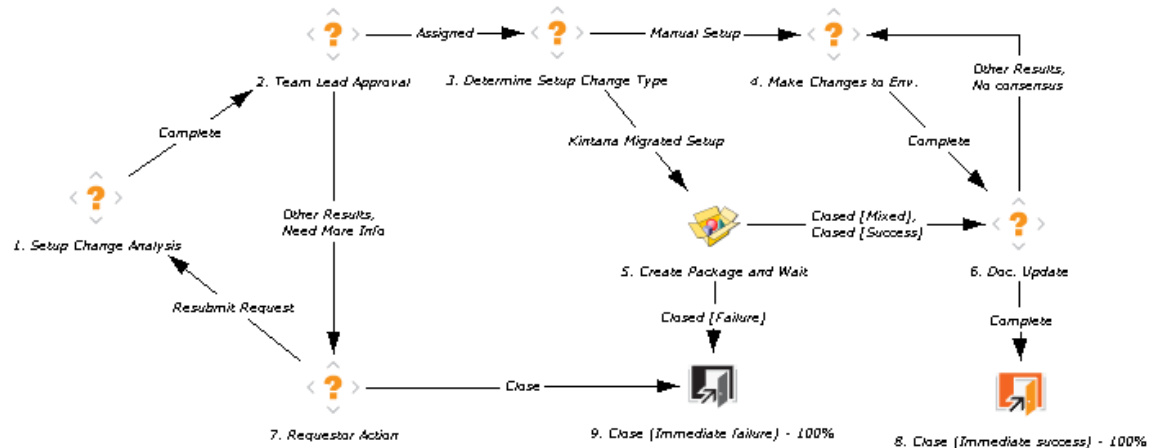
For general configuration considerations for workflows, see ["General Configuration Considerations" on page 239](#).

For more information about the request type associated with the OraApps Setup Change Process workflow, see ["OraApps Setup Change Request Request Type" on page 230](#).

Workflow Diagram and Step Descriptions

["Figure C-13. OraApps Setup Change Process workflow"](#) shows the workflow.

Figure C-13. OraApps Setup Change Process workflow



The workflow steps are described as follows:

1. **Setup Change Analysis.** The functional lead analyzes the proposed change, determines its estimated effort and dependencies, if any, and determines the estimated completion date for the change.
2. **Team Lead Approval.** In this step, the technical lead can approve the setup change for the environment and assign a resource. If the team lead disapproves the setup change or needs more information about it, the change request goes to the Requestor Action step.
3. **Determine Setup Change Type.** The assigned developer determines whether the request requires a manual setup change or a setup change migrated by PPM Center. If the request requires a manual setup change, the request goes to the Make Changes to Env step. If the request requires a setup change migrated by PPM Center, the request goes to the Create Package and Wait step.
4. **Make Changes to Env.** The setup changes are implemented by the assigned developer.
5. **Create Package and Wait.** For setup changes migrated by PPM Center, a package is created by the developer, and the package waits here until implemented.
6. **Doc. Update.** The setup changes are documented by the developer.
7. **Requestor Action.** The requestor can resubmit the request or close it.
8. **Close (Immediate success).** After the setup changes are documented, the request closes with status of success.
9. **Close (Immediate failure).** If the requestor closes the request in the Requestor Action step or the package is closed with failure at the Create Package and Wait step, the setup change is closed with a status of failure.

OraApps Status Update Request Workflow

Many organizations use manual status reports for gathering information on project status, exceptions, and risks. However, the process can be inefficient due to the time and resources spent on updating and consolidating information, and on follow-up by managers to make sure group members are completing their tasks. Reviewers might not

know when status reports are ready for review, whom to contact, or how to access the most current versions of the manual reports.

The OraApps Status Update Request workflow implements the process of gathering and reviewing status reports. This workflow allows managers to request information, automatically notifies group members of requirements for information, and monitors task progress to support timely completion. When tasks for completing a report have been done, all reviewers are automatically notified. All involved parties have access to the latest version of the report.

This workflow works with the OraApps Status Update Request request type.

For a description of request processes, see ["Overview of the Request Process" on page 207](#).

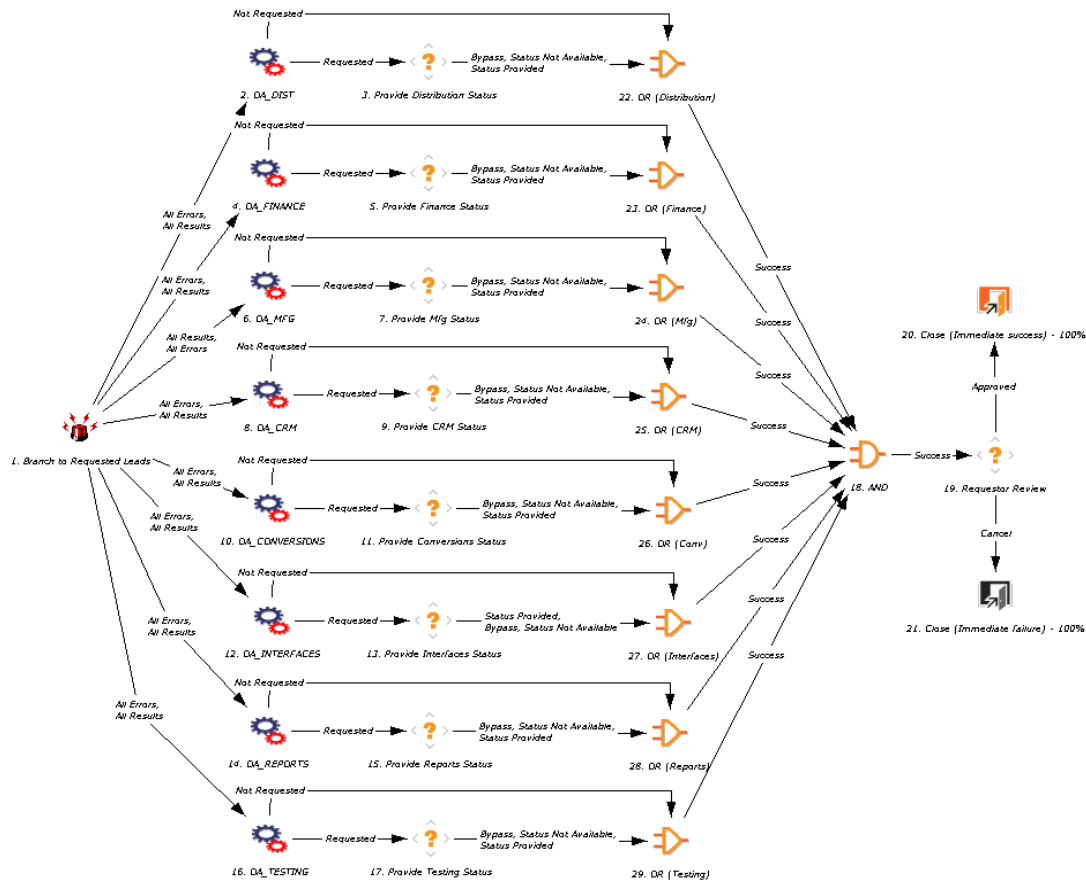
For general configuration considerations for workflows, see ["General Configuration Considerations" on page 239](#).

For more information about the request type associated with the OraApps Status Update Request workflow, see ["OraApps Status Update Request" on page 233](#).

Workflow Diagram and Step Descriptions

["Figure C-14. OraApps Status Update Request workflow"](#) shows the workflow.

Figure C-14. OraApps Status Update Request workflow



The workflow steps are described as follows:

1. **Branch to Requested Leads.** This is an automatic execution step that starts each of the team branches in the workflow.
2. **OA_DIST.** This is an automatic token resolution step that looks at the value of the Distribution Status field (field token = OA_DIST). If the value is Requested, the workflow goes to the Provide Distribution Status step. If the value is **Not Requested**, the workflow skips over that step. The key to this token resolution is that the step name must exactly match the request field token name.
3. **Provide Distribution Status.** This step notifies the user who was specified in the Distribution Team Lead field to complete a status report and attach it to the request. By default, the reminder notification is sent every two days until the task is done (the amount of time is configurable). The user attaches the report, when completed, and sets this step to Status Provided. The user or requestor can also set this step to Bypass, Status Not Available if no report is going to be provided and the workflow should move forward.
4. **-17. Other branches.** All the other branches in the workflow (such as OA_FINANCE and Provide Finance Status, OA_MFG and Provide Mfg Status, and so on) follow the same logic as the OA_DIST and Provide Distribution Status steps.

19. **Requestor Review.** Once every branch is complete (either by the completion of its Provide Status step or by skipping the branch because its report was not requested), the original creator of the request is notified that the reports are ready. The requestor should review the reports and either set this step to Approved or Cancel. Either result closes the request.

Steps with Predefined Security

"[Table C-13. OraApps Status Update Request workflow predefined security](#)" shows the steps of the workflow that have predefined security.

Table C-13. OraApps Status Update Request workflow predefined security

Step	Step Name	Security Setting
2	Business Acceptance	Assigned To Group Assigned To User Created By User
3	Provide Distribution Status	Assigned To Group Assigned To User Created By User
4	OA_FINANCE	Assigned To Group Assigned To User Created By User
5	Provide Finance Status	Assigned To Group Assigned To User Created By User
6	OA_MFG	Assigned To Group Assigned To User Created By User
7	Provide Mfg Status	Assigned To Group Assigned To User Created By User

Table C-13. OraApps Status Update Request workflow predefined security, continued

Step	Step Name	Security Setting
8	OA_CRM	Assigned To Group Assigned To User Created By User
9	Provide CRM Status	Assigned To Group Assigned To User Created By User
10	OA_CONVERSIONS	Assigned To Group Assigned To User Created By User
11	Provide Conversions Status	Assigned To Group Assigned To User Created By User
12	OA_INTERFACES	Assigned To Group Assigned To User Created By User
13	Provide Interfaces Status	Assigned To Group Assigned To User Created By User
14	OA_REPORTS	Assigned To Group Assigned To User Created By User
15	Provide Reports Status	Assigned To Group Assigned To User Created By User

Table C-13. OraApps Status Update Request workflow predefined security, continued

Step	Step Name	Security Setting
16	OA_TESTING	Assigned To Group Assigned To User Created By User
17	Provide Testing Status	Assigned To Group Assigned To User Created By User
18	Requestor Review	Assigned To Group Assigned To User Created By User

OraApps 11i Cloning Workflow

Using Oracle E-Business Suite requires management of a number of ongoing operating instances and temporary implementation or upgrade instances. These instances must be efficiently created, maintained, and periodically cloned without risking the stability of the system and disruption of day-to-day activities.

The OraApps 11i Cloning workflow works in conjunction with the OraApps 11i Cloning object type to clone an instance of Oracle E-Business Suite by automating the major steps in the process. The initial request for a clone usually comes from the OraApps Cloning Request request type and its associated workflow, the OraApps Cloning Process.

Information pertinent to the environments to be refreshed is captured in the object type. This information is to be used during execution of commands that clone the instance. Only the databases and directories that are supposed to be cloned are affected.

When the OraApps Cloning Process workflow reaches the Create Package and Wait step, the user creates a package that uses the OraApps 11i Cloning workflow and the OraApps 11i Cloning object type in its package lines.

The process implemented by this workflow and its associated object type follow the guidelines in the *Cloning Oracle Applications Release 11i* white paper published by Oracle.

The OraApps 11i Cloning workflow does the following general steps:

- Verifies the cloning requirements
- Copies the cloning scripts

- Runs the scripts in pre-clone mode to preserve the instance's configuration information
- Shuts the database down, deletes any database files that must be deleted, and copies database files from the source backup to their destination
- Creates the control files for the destination database
- Starts the destination database
- Cleans up any concurrent requests copied from the source
- Copies various Oracle code files from the source backup directory to the destination
- Executes the cloning scripts in post-cloning mode

If the package was invoked from a request, the outcome status of the package is returned to the request.

Caution:

Cloning your Oracle instance is a complex process, and misconfiguration can cause serious consequences. This object type and its associated workflow provide a template for cloning your instances, but require tailoring for your environment. Contact Micro Focus Software Support to facilitate a successful implementation and see the Micro Focus Software Support Web site:

<https://softwaresupport.hpe.com>

For a description of request processes, see "[Overview of the Request Process](#)" on page 207.

For general configuration considerations for workflows, see General Configuration Considerations on page 305.

For more information about the request type associated with the OraApps 11i Cloning workflow, see "[OraApps Cloning Request Request Type](#)" on page 213.

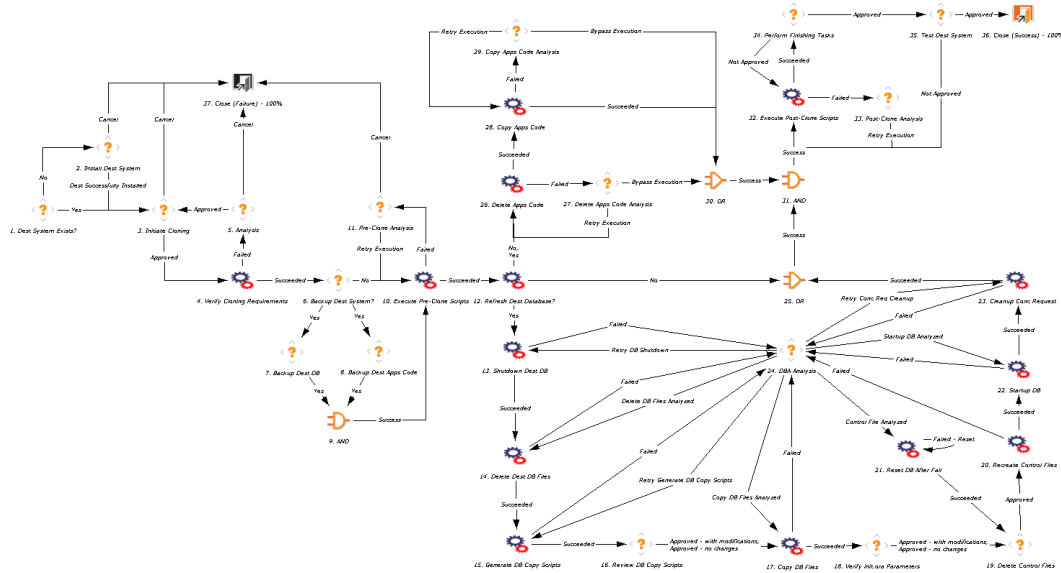
For more information about the object type associated with the OraApps 11i Cloning workflow, see "[OraApps 11i Cloning Object Type](#)" on page 192.

For more information about the OraApps Cloning Process workflow, see "[OraApps Cloning Process Workflow](#)" on page 243.

Workflow Diagram and Step Descriptions

"[Figure C-15. OraApps 11i Cloning workflow](#)" shows the workflow.

Figure C-15. OraApps 11i Cloning workflow



The workflow steps are described as follows:

1. **Dest System Exists?** To clone an instance, the destination system must be installed. The user is asked to verify the existence of the destination system.
2. **Install Dest System.** If the destination system does not exist, the user must install it. After installation, the workflow goes to the Initiate Cloning step.
3. **Initiate Cloning.** If the destination system exists, the cloning process is initiated.
4. **Verify Cloning Requirements.** The requirements for cloning (such as having sufficient disk space, the existence of Perl, and the application of the cloning patch) are verified.
5. **Analysis.** If any of the cloning requirements are not met, an analysis of the problem is conducted. The problem must be resolved before the workflow can proceed to the Backup Dest System? step.
6. **Backup Dest System?** If all the cloning requirements are met, the user is prompted to back up the destination system.
7. **Backup Dest DB and 8. Backup Dest Apps Code.** The user can choose whether to perform these backup steps. If the user chooses not to perform these steps, if the backup has already been performed, or after the backup completes, the workflow proceeds to the Execute Pre-Clone Scripts step.
10. **Execute Pre-Clone Scripts.** The pre-cloning scripts are executed.
11. **Pre-Clone Analysis.** If the Execute Pre-Clone Scripts step fails, a pre-clone analysis is performed. The cloning can be cancelled or a fix can be implemented for retry of the Execute Pre-Clone Scripts step.
12. **Refresh Dest Database?** The user can choose to refresh the database and application code simultaneously, or refresh only the application code. During the database refresh, the following steps are executed:

- **Shutdown Dest DB.** The database is shut down.
 - **Delete Dest DB Files.** The destination database files are deleted.
 - **Generate DB Copy Scripts.** Copy scripts are generated to copy the database files from source to destination.
16. **Review DB Copy Scripts.** The user is prompted to review the generated copy scripts and make changes, if necessary.
 17. **Copy DB Files.** The copy scripts are executed and the database files are copied to the destination in conjunction with the following associated steps:
 18. **Verify init.ora Parameters.** Verify that the init.ora fields are correct.
 19. **Delete Control Files.** Decide whether to delete any control files indicated in the init.ora file.
 20. **Recreate Control Files.** The control files are re-created.
 21. **Reset DB After Fail.** If the Recreate Control Files step, the Startup DB step, or the Cleanup Conc Request step fails, the DBA Analysis step is called. After the control file is analyzed, the database is reset in this step.
 22. **Startup DB.** The database is opened.
 23. **Cleanup Conc Request.** Any concurrent requests are cleaned up.
 24. **DBA Analysis.** Any errors that require analysis and resolution by the DBA are identified.
 26. **Delete Apps Code.** During the refresh of the application code, the destination files are deleted.
 27. **Copy Apps Code Analysis.** If any errors were found during the Copy Apps Code step, an analysis is performed and the step is rerun.
 28. **Execute Post-Clone Scripts.** Once the refresh is completed, the post-clone scripts are executed.
 29. **Post-Clone Analysis.** If any errors are found after the scripts are executed, an analysis is performed and the step is rerun.
 32. **Execute Post-Clone Scripts.** Once the refresh is completed, the post-clone scripts are executed.
 33. **Post-Clone Analysis.** If any errors are found after the scripts are executed, an analysis is performed and the step is rerun.
 34. **Perform Finishing Tasks.** Any required finishing tasks are performed.
 35. **Test Dest System.** The destination system is tested.
 36. **Close (Success).** The workflow is closed with a status of success.
 37. **Close (Failure).** The workflow is closed with a status of failure.

Appendix D: Report Types

This section provides reference information about the Oracle-specific report types provided in the Extension. The report types are described in alphabetical order, except that patch-related reports are consolidated at the end of this appendix.

The content or output of a report is controlled by what you specify in the fields for the report type, as described in this appendix. Some fields allow multiple entries. Some fields are hidden by default, but can be enabled to allow further control of the report output.

Execution of reports is driven by commands included within the report types. For more information about commands in the PPM Center environment, see the *Commands, Tokens, and Validations Guide and Reference*.

Note:

In addition to the report types provided by the Extension and described here, PPM Center includes the Compare Oracle Environments Report, which you can use to compare the data model of two Oracle schemas. For more information, see the *Reports Guide and Reference*.

To configure report types:

1. Log on to PPM.
2. From the menu bar, select **Open > Administration > Open Workbench**.
The PPM Workbench opens.
3. From the shortcut bar, select **Demand Mgmt > Request Types**.
The Request Type Workbench opens.
4. (Optional) To limit the list of reports to those for the installed Extensions, select **Extension** in the **Report Category** field on the **Query** tab.
5. Click **List**.
6. Select the report type of interest and click **Open**.
7. Edit the report type. (On the **Fields** tab, the list of fields in the **Prompts** column is alphabetized.)

To run (submit) a report:

1. Log on to PPM.
2. From the menu bar, select **Reports > Create a Report**.
3. Select a report from the **Recently Submitted Reports** section, or select a category from the **Report Category** option and select a particular report.
4. Complete the fields on the Submit Report window and click **Submit**. Refer to this appendix for details.

For information about running reports, see the *Reports Guide and Reference*.

Reference Report Types

Reference report types cannot be edited, but you can copy and rename them and edit the copies to meet your needs. You can also use existing non-reference report types as is or configure them further to meet your needs.

List of Request Types

"[Table D-1. Report types included in the Extension](#)" lists and defines the report types included in the Extension. Each is described in subsequent sections.

Table D-1. Report types included in the Extension

Request Type	Workflow
OraApps Apps Issues Detail Report	For the OraApps Application Issue request type, provides information similar to the Request Detail (Filter by Custom Fields) Report in Demand Management.
OraApps Apps Issues Summary Report	For the OraApps Application Issue request type, provides information similar to the Request Summary (Filter by Custom Fields) Report in Demand Management.
OraApps Critical Requests Summary	Lists a summary of Extension-related requests with Critical priority, similar to the Request Quick View Report in Demand Management.
OraApps IT Demand Summary	Lists a summary of selected Extension-related requests, similar to the Request Summary Report in Demand Management.
Patch-Related Reports	
OraApps Patch Analysis Report	Lists details of selected patches. Similar to the OraApps Patch Detail Report but does not allow loading of patch data.
OraApps Patch Detail Report	Lists details of selected patches or loads patch data for unapplied patches.
OraApps Patch Matrix	Generates an annotated matrix of patches and environments, identifying the environments that have had patches applied.
OraApps Patches Applied for Specific Bug	Lists the patches that have been applied to resolve the bug that a user specifies.

Table D-1. Report types included in the Extension, continued

Request Type	Workflow
Patch Application Comparison Report	Lists Oracle patches migrated to or applied to up to six different environments, similar to the Environment Comparison by Objects Migrated Report in Deployment Management.
Patches Applied to an Environment Report	Lists Oracle patches migrated to or applied to a specific environment, similar to the Environment/Objects Detail Report in Deployment Management.
Pending Patches Report	Lists package lines for Oracle patches that are waiting at a migration step. This functionality is an Oracle-specific subset of the Packages Pending Report in Deployment Management.

General Configuration Considerations

Consider the following for all report types:

- Report types must be enabled before they can be used.
- Any security restrictions required for the reports must be configured. By default, the reports are available to all users.

OraApps Apps Issues Detail Report

The following sections provide a description, configuration considerations, and field descriptions for the OraApps Apps Issues Detail Report.

Description

The OraApps Apps Issues Detail Report is similar to the Request Detail (Filter by Custom Fields) report in Demand Management, but the OraApps Apps Issues Detail Report is specialized for the OraApps Application Issue request type. You can filter by the standard header fields as well as some of the specific detail fields, for example, **TAR #**, **Business Benefit**, and **Environment**.

Configuration Considerations

Consider the following in configuring this report:

- Users can further configure the report.
- Additional hidden fields can be configured to restrict the set of data in the report.

Field Descriptions

"[Figure D-1. OraApps Apps Issues Detail Report](#)" shows the Submit Report window you use to specify which requests you want to appear in the OraApps Apps Issues Detail Report and which information you want to appear for each request. "[Table D-2. OraApps Apps Issues Detail Report fields](#)" provides descriptions of the fields in the report that are displayed by default.

Figure D-1. OraApps Apps Issues Detail Report

Hewlett Packard Enterprise
HPE PROJECT AND PORTFOLIO MANAGEMENT CENTER

Submit Report: OraApps Apps Issues Detail Report

Submit
Cancel

Report Parameters
Restore Default

Request Numbers:	<input type="text"/>	*Include Closed Requests:	<input checked="" type="radio"/> Yes <input type="radio"/> No
*Request Type:	<input type="text" value="OraApps Application Issue"/>	Business Priority:	<input type="text"/>
Status:	<input type="text"/>	Assigned To Group:	<input type="text"/>
Assigned To:	<input type="text"/>	Request Sub Type:	<input type="text"/>
Created By:	<input type="text"/>	Request Group:	<input type="text"/>
Department:	<input type="text"/>	Company Name:	<input type="text"/>
Workflow:	<input type="text"/>	Creation Date From:	<input type="text"/>
Contact:	<input type="text"/>	Creation Date To:	<input type="text"/>
Creation Date From:	<input type="text"/>	Last Update Date From:	<input type="text"/>
Last Update Date From:	<input type="text"/>	Last Update Date To:	<input type="text"/>
Problem Contains:	<input type="text"/>		
Internal/External:	<input type="text" value="Internal"/>	Business Area Affected:	<input type="text"/>
TAR #	<input type="text"/>	Source:	<input type="text"/>
Environment:	<input type="text"/>	Business Benefit:	<input type="text"/>
Required within next X days:	<input type="text"/>	Est. Comp. within next X days:	<input type="text"/>
Field Prompt1:	<input type="text"/>	Field Value1:	<input type="text"/>
Field Prompt2:	<input type="text"/>	Field Value2:	<input type="text"/>
Report Title:	<input type="text" value="OraApps Apps Issues Detail Report"/>	*Order By:	<input type="text" value="Request Number"/>
*Show Header Fields:	<input checked="" type="radio"/> Yes <input type="radio"/> No	*Show Detail Fields:	<input checked="" type="radio"/> Yes <input type="radio"/> No
*Hide Prompts for Empty Fields:	<input checked="" type="radio"/> Yes <input type="radio"/> No	*Show Contents of Table Fields:	<input type="radio"/> Yes <input checked="" type="radio"/> No
*Show Field Audit History:	<input type="radio"/> Yes <input checked="" type="radio"/> No	*Filter Notes:	<input type="text" value="Show all notes"/>
*Show Notes:	<input checked="" type="radio"/> Yes <input type="radio"/> No	*Show References:	<input checked="" type="radio"/> Yes <input type="radio"/> No
*Show Status:	<input checked="" type="radio"/> Yes <input type="radio"/> No		

Scheduling

Run Report Immediately

Run Report On:

Repeat Every Hours Until

Send email to: when report is finished

Advanced Notifications

Submit
Cancel

Close Window X

Table D-2. OraApps Apps Issues Detail Report fields

Field Name (*Required)	Description
Request Numbers	Request numbers to include in the report
*Include Closed Requests	Option to include requests that have been closed or cancelled
*Requested Type	Requests having this request type
Status	Requests having this status
Business Priority	Requests having this business priority
Assigned To	Requests assigned to this user
Assigned To Group	Requests assigned to this group
Created By	Requests created by this user
Request Sub Type	Requests having this sub-type
Department	Requests logged against this department
Workflow	Requests that are associated with this workflow
Request Group	Requests using request group(s)– Customization, Upgrade, or Setup
Contact	Requests having this contact name
Company Name	Requests having this company name
Creation Date From	Requests created on this date or later
Creation Date To	Requests created on this date or earlier
Last Update Date From	Requests last updated on this date or later
Last Update Date To	Requests last updated on this date or earlier
Problem Contains	Requests having this text string
Internal/External	Requests that are Internal or External
Business Area Affected	Requests for this business area
TAR #	Requests with this TAR number
Source	Requests with this source

Table D-2. OraApps Apps Issues Detail Report fields, continued

Field Name (*Required)	Description
Environment	Requests associated with this Oracle environment
Business Benefit	Requests with this business benefit from resolving a particular issue
Required within next X days	Requests required within a particular number of days
Est. Comp. within next X days	Requests with an estimated completion date within a particular number of days
Field Prompt1	Custom field to use as a filter
Field Value1	Value for Field Prompt1
Field Prompt2	Custom field to use as a filter
Field Value2	Value for Field Prompt
Report Title	Title of the report
*Order By	<p>Criterion to use to sort the requests in the report:</p> <ul style="list-style-type: none"> • Application • Assigned To • Created By • Creation Date • Department • Last Update Date • Priority • Request Group • Request Number • Request Sub Type • Request Type • Status
*Show Header Fields	Option for the report to show the full header fields of each request
*Show Detail Fields	Option for the report to show the detail fields of each request

Table D-2. OraApps Apps Issues Detail Report fields, continued

Field Name (*Required)	Description
*Hide Prompts for Empty Fields	Option for the report to hide prompts that have empty fields
*Show Contents of Table Fields	Option for the report to show table fields for requests that have them
*Show Field Audit History	Option for the report to show the transaction history of each request
*Show Notes	Option for the report to show the notes attached to each request
*Filter Notes	Option for the report to show all notes or only user notes
*Show Status	Option for the report to show the workflow steps and current step status for each request
*Show References	Option for the report to show the references associated with each request

OraApps Apps Issues Summary Report

The following sections provide a description, configuration considerations, and field descriptions for the OraApps Apps Issues Summary Report.

Description

The OraApps Apps Issues Summary Report is similar to the Request Summary (Filter by Custom Fields) Report in Demand Management, but the OraApps Apps Issues Summary Report is specialized for the OraApps Application Issue request type. You can filter by both the standard header fields and some of the specific detail fields, for example, **TAR #**, **Business Benefit**, and Environment. You can also summarize by many of the detailed data, using the dynamic **Group By** field.

Configuration Considerations

Consider the following in configuring this report:

- If you have removed detailed fields in the OraApps Application Issue request type, disable any fields referring to the deleted fields.
- If you change the name of the request type to something other than OraApps Application Issue, you must also do the following:

- Change the default for the **Request Type** field.
- Create a copy of the OA - Apps Issue Reporting Request Types validation and make the appropriate changes to the values within the copy.
- Create a copy of the OA - Apps Issue Request Types validation and modify it to point to the lookup type used by the new version of the OA - Apps Issue Reporting Request Types validation you just created in the previous bullet.
- Change the **Request Type** field to use the new version of the OA - Apps Issue Request Types validation you just created in the previous bullet.
- You can add grouping columns to OraApps Apps Issues Summary Report, but if detail fields are used, they must be part of the first field batch.

Field Descriptions

"[Figure D-2. OraApps Apps Issues Summary Report](#)" shows the Submit Report window you use to specify which requests you want to appear in the OraApps Apps Issues Summary Report and which information you want to appear for each request. "[Table D-3. OraApps Apps Issues Summary Report fields](#)" provides descriptions of the fields in the report that are displayed by default.

Figure D-2. OraApps Apps Issues Summary Report

Hewlett Packard Enterprise
HPE PROJECT AND PORTFOLIO MANAGEMENT CENTER

Submit Report: OraApps Apps Issues Summary Report

Submit
Cancel

Report Parameters
Restore Default

<p>Request Numbers: <input type="text"/></p> <p>*Request Type: OraApps Application Issue</p> <p>Status: <input type="text"/></p> <p>Assigned To: <input type="text"/></p> <p>Created By: <input type="text"/></p> <p>Department: <input type="text"/></p> <p>Workflow: <input type="text"/></p> <p>Contact: <input type="text"/></p> <p>Creation Date From: <input type="text"/></p> <p>Last Update Date From: <input type="text"/></p> <p>Description Contains: <input type="text"/></p> <p>Internal/External: Internal</p> <p>TAR # <input type="text"/></p> <p>Environment: <input type="text"/></p> <p>Required within next X days: <input type="text"/></p> <p>Field Prompt 1: <input type="text"/></p> <p>Field Prompt 2: <input type="text"/></p> <p>Report Title: OraApps Apps Issues Summary Rep</p> <p>*Group By: <input type="text"/></p> <p>*Include Subtotals for First Group Column: <input checked="" type="radio"/> Yes <input type="radio"/> No</p>	<p>*Include Closed Requests: <input checked="" type="radio"/> Yes <input type="radio"/> No</p> <p>Priority: <input type="text"/></p> <p>Assigned To Group: <input type="text"/></p> <p>Request Sub Type: <input type="text"/></p> <p>Request Group: <input type="text"/></p> <p>Creation Date To: <input type="text"/></p> <p>Last Update Date To: <input type="text"/></p> <p>Business Area Affected: <input type="text"/></p> <p>Source: <input type="text"/></p> <p>Business Benefit: <input type="text"/></p> <p>Est. Comp. within next X days: <input type="text"/></p> <p>Field Value 1: <input type="text"/></p> <p>Field Value 2: <input type="text"/></p>
---	---

Scheduling

Run Report Immediately

Run Report On:

Repeat Every Hours v Until

Send email to: Admin User when report is finished

Advanced Notifications

Submit
Cancel

Close Window

Table D-3. OraApps Apps Issues Summary Report fields

Field Name (*Required)	Description
Request Numbers	Request numbers to include in the report
*Include Closed Requests	Option to include requests that have been closed or cancelled
*Requested Type	Requests having this request type
Status	Requests having this status
Business Priority	Requests having this business priority
Assigned To	Requests assigned to this user
Assigned To Group	Requests assigned to this group
Created By	Requests created by this user
Request Sub Type	Requests having this sub-type
Department	Requests logged against this department
Workflow	Requests that are associated with this workflow
Request Group	Requests using request group(s)– Customization, Upgrade, or Setup
Contact	Requests having this contact name
Creation Date From	Requests created on this date or later
Creation Date To	Requests created on this date or earlier
Last Update Date From	Requests last updated on this date or later
Last Update Date To	Requests last updated on this date or earlier
Description Contains	Requests having this description text string (not a case-sensitive search)
Internal/External	Requests that are Internal or External
Business Area Affected	Requests for this business area
TAR #	Requests with this TAR number
Source	Requests with this source

Table D-3. OraApps Apps Issues Summary Report fields , continued

Field Name (*Required)	Description
Environment	Requests associated with this Oracle environment
Business Benefit	Requests with this business benefit from resolving a particular issue
Required within next X days	Requests required within a particular number of days
Est. Comp. within next X days	Requests with an estimated completion date within a particular number of days
Field Prompt1	Custom field to use as a filter
Field Value1	Value for Field Prompt1
Field Prompt2	Custom field to use as a filter
Field Value2	Value for Field Prompt
Report Title	Title of the report
*Group By	How to group the report results by a particular field or fields from the request
*Show Detail Fields	Option for the report to show the detail fields of each request
*Include Subtotals for First Group Column	Option for the report to show subtotals for the first column specified in the Group By field

OraApps Critical Requests Summary Report

The following sections provide a description, configuration considerations, and field descriptions for the OraApps Critical Requests Summary Report.

Description

The OraApps Critical Requests Summary Report is similar to the Request Quick View Report in Demand Management. The OraApps Critical Requests Summary Report lists a summary of Extension-related requests with Critical priority. The report displays summary information about the critical requests opened and closed in the current week as well as the numbers that are open by request type. In addition, the report lists detailed information for each request.

Configuration Considerations

Consider the following in configuring this report:

- Users can further configure the report.
- Additional hidden fields can be configured to restrict the set of data in the report.

Field Descriptions

"[Figure D-3. OraApps Critical Requests Summary Report](#)" shows the Submit Report window you use to specify which requests you want to appear in the OraApps Critical Requests Summary Report and which information you want to appear for each request. "[Table D-4. OraApps Critical Requests Summary Report fields](#)" provides descriptions of the report's displayed fields and some of the fields that are hidden by default.

Figure D-3. OraApps Critical Requests Summary Report

Hewlett Packard Enterprise | HPE PROJECT AND PORTFOLIO MANAGEMENT CENTER

Submit Report: OraApps Critical Requests Summary

Submit Cancel

Report Parameters Restore Default

Report Title: OraApps Critical Requests Summary

Show Request Details: Yes No

Scheduling

Run Report Immediately

Run Report On: [] []

Repeat Every [] Hours [] Until [] []

Send email to: Admin User [] when report is finished

Advanced Notifications

Submit Cancel

Close Window X

Table D-4. OraApps Critical Requests Summary Report fields

Field Name (*Required)	Description	Hidden by Default?
Report Title	Title of the report	No
*Show Request Details	Option for the report to show the Request Details table	No
Requested Type	Requests having this request type	Yes
Status	Requests having this status	Yes
Priority	Critical requests, in keeping with this report's purpose	Yes
Assigned To	Requests assigned to this user	Yes
Assigned To Group	Requests assigned to this group	Yes
Department	Requests logged against this department	Yes
Application	Requests that are associated with this application	Yes
Workflow	Requests that are associated with this workflow	Yes
Request Group	Requests using this request group	Yes
Create Date From	Requests created on this date or later	Yes
Creation Date To	Requests created on this date or earlier	Yes
Order By	Criterion to use to sort the requests in the report	Yes

OraApps IT Demand Summary Report

The following sections provide a description, configuration considerations, and field descriptions for the OraApps IT Demand Summary Report.

Description

The OraApps IT Demand Summary Report lists a summary of selected Extension-related requests. The report is similar to the Request Summary Report in Demand Management, and it displays the following columns:

- **Request Type**
- **Priority**
- **Assigned Group**
- **Count**

Optionally, this report provides subtotals. This report is available to all PPM users by default.

Configuration Considerations

Consider the following in configuring this report:

- Users can further configure the report.
- Additional hidden fields can be configured to restrict the set of data in the report.

Field Descriptions

"[Figure D-4. OraApps IT Demand Summary Report](#)" shows the Submit Report window you use to specify which information you want to appear for each request in the OraApps IT Demand Summary Report. Hidden fields, when enabled, allow you to limit which requests are included in the report. "[Table D-5. OraApps IT Demand Summary Report fields](#)" provides descriptions of the report's displayed fields and some of the fields that are hidden by default.

Figure D-4. OraApps IT Demand Summary Report

The screenshot shows the 'Submit Report: OraApps IT Demand Summary' window in the HPE Project and Portfolio Management Center. The window is divided into three main sections: 'Report Parameters', 'Scheduling', and 'Advanced Notifications'.
 - **Report Parameters:** Includes a 'Restore Default' button and a radio button for 'Include Subtotals for First Group Column' set to 'Yes'.
 - **Scheduling:** Features radio buttons for 'Run Report Immediately' (selected) and 'Run Report On:'. The 'Run Report On' section includes a date picker, a 'Repeat Every' checkbox with a 'Hours' dropdown, and an 'Until' date picker. There is also a 'Send email to:' field with 'Admin User' and a checkbox for 'when report is finished'.
 - **Advanced Notifications:** A section with a 'Submit' button.
 The window has 'Submit' and 'Cancel' buttons at the top right and bottom right, and a 'Close Window' button at the bottom right.

Table D-5. OraApps IT Demand Summary Report fields

Field Name (*Required)	Description	Hidden by Default?
*Include Subtotals for First Group Column	Option for the report to show subtotals for the first column specified in the Group By field	No
Request Numbers	Request numbers to include in the report	Yes
Include Closed Requests	Option to include requests that have been closed or cancelled	Yes
Request Type	Requests having this request type	Yes
Status	Requests having this status	Yes
Priority	Requests having this priority	Yes
Assigned To	Requests assigned to this user	Yes

Table D-5. OraApps IT Demand Summary Report fields, continued

Field Name (*Required)	Description	Hidden by Default?
Assigned To Group	Requests assigned to this group	Yes
Created By	Requests created by this user	Yes
Request Sub Type	Requests having this sub-type	Yes
Department	Requests logged against this department	Yes
Application	Requests that are associated with this application	Yes
Workflow	Requests that are associated with this workflow	Yes
Request Group	Requests using this request group	Yes
Contact	Requests having this contact name	Yes
Company Name	Requests having this company name	Yes
Create Date From	Requests created on this date or later	Yes
Creation Date To	Requests created on this date or earlier	Yes
Last Update Date From	Requests last updated on this date or later	Yes
Last Update Date To	Requests last updated on this date or earlier	Yes
Description Contains	Requests having this description text string (not a case-sensitive search)	Yes
Group By	How to group the report results by a particular field or fields from the request	Yes

Patch-Related Reports

In addition to the Deployment Management reports common for all package processing, there are reports specific to the Patch Management functionality of the Extension. These reports provide detailed information about patches that have been applied or are in progress.

OraApps Patch Analysis Report

The following sections provide a description, configuration considerations, and field descriptions for the OraApps Patch Analysis Report.

Description

The OraApps Patch Analysis Report is similar to the OraApps Patch Detail Report, except that the OraApps Patch Analysis Report does not allow loading of patch data. Also, rather than giving information in a master-detail format, the OraApps Patch Analysis Report consolidates all the information into a single table, which allows you to bring the information into spreadsheet programs such as Microsoft Excel for additional reporting. In addition to having a variety of filter criteria, you can also configure which information to show in the table and in what order. For example, the user might want to view the patch header and included bug information, or the bug and file information. You can use the OraApps Patch Analysis Report to show, for example, which patches have been applied to a given environment or which patches contain a specific file. Only unique value combinations are displayed, eliminating the need to filter through multiple records.

Field Descriptions

"[Figure D-5. OraApps Patch Analysis Report](#)" shows the Submit Report window you use to specify which patches you want the OraApps Patch Analysis Report to analyze and which information you want to appear for each patch. Table D-6 provides descriptions of the fields in the report that are displayed by default.

Figure D-5. OraApps Patch Analysis Report

HPE PROJECT AND PORTFOLIO MANAGEMENT CENTER

Submit Report: OraApps Patch Analysis Report

Report Parameters Restore Default

*Report Title: OraApps Patch Analysis Report

Patch:

Bug:

Patch Application:

Environment:

Applied Since:

File Type:

File Application:

File Name:

File Version:

*OraApps Release: 11i

*Include All Data of Selected Patches? Yes No

*Include Non-Applied Patches? Yes No

*Patch Info Sequence: 1 - Show and Order First

*SubPatch Info Sequence: 2 - Show and Order Second

*Env Info Sequence: 3 - Show and Order Third

*File Info Sequence: Dont Show

Scheduling

Run Report Immediately

Run Report On:

Repeat Every Hours Until

Send email to: Admin User when report is finished

Advanced Notifications

Submit Cancel

Close Window X

Table D-6. OraApps Patch Analysis Report fields

Field Name (*Required)	Description
*Report Title	Title of the report
Patch	Patch number to analyze.

Table D-6. OraApps Patch Analysis Report fields, continued

Field Name (*Required)	Description
Bug	Bug (subpatch) number to analyze. If the Include All Data of Selected Patches field is set to Yes , the report lists all the information for patches that contain this bug. If that field is set to No , the report lists only information relevant to this bug.
Patch Application	Application to which the patch, subpatch, or bug belongs.
Environment	Environment for which applied patches will be reported.
Applied Since	Patches applied on this date or later.
File Type	Patches that include files of this file type. If the Include All Data of Selected Patches field is set to Yes , the report lists information for patches that contain files of this file type. If that field is set to No , the report lists only information relevant to the files of this file type.
File Name	Patches that include this file name. See the File Type field for analogous behavior. This field is case-sensitive.
File Version	Patches that include files of this file version. See the File Type field for analogous behavior.
*OraApps Release	Oracle release to report on. The default is 11i.
*Patch Info Sequence	Option to show the patch header (patch number, patch application) in the report, and if shown, the order to place the patch header in the output table. If multiple fields have the same sequence number, the report orders the information based on the field ordering in the report type. Also, if a sequence is missed (for example, the sequences are set to 1, 3, 4, and Don't Show), the report ignores the gap. Finally, if all the columns are set to Don't Show, the report lists only the patch header information.
*SubPatch Info Sequence	Option to show the subpatch header (for example, bug number, bug app) in the report, and if shown, the order to place the subpatch header in the output table. See the Patch Info Sequence field for additional information.

Table D-6. OraApps Patch Analysis Report fields, continued

Field Name (*Required)	Description
*Env Info Sequence	Option to show environment information (environment name, last applied) in the report, and if shown, the order to place the environment information in the output table. See the Patch Info Sequence field for additional information.
*File Info Sequence	Option to show file information (for example, file name, file version) in the report, and if shown, the order to place the file information in the output table. See the Patch Info Sequence field for additional information.

OraApps Patch Detail Report

The following sections provide a description, configuration considerations, and field descriptions for the OraApps Patch Detail Report.

Description

The OraApps The OraApps Patch Detail Report lists details of selected patches. This report is also used to load patch data for unapplied patches. Users have control over the types of information displayed and the patches selected. If fields are selected to load patch data, the report type calls a special command to load the data before executing the report. If required, the report calls the same command to remove the patch data after report completion.

Configuration Considerations

You should carefully control who is allowed to load and purge patch data. If multiple groups of users need access to OraApps Patch Detail Report, but not all of them should be able to load and purge patch data, make additional copies of this report for these users, hiding the load and purge fields.

Field Descriptions

"[Figure D-6. OraApps Patch Detail Report](#)" shows the Submit Report window you use to specify which patches you want the OraApps Patch Detail Report to report on or load, and which information you want to appear for each patch. "[Table D-7. OraApps Patch Detail Report fields](#)" provides descriptions of the report's displayed fields and some of the fields that are hidden by default.

Figure D-6. OraApps Patch Detail Report

Table D-7. OraApps Patch Detail Report fields

Field Name (*Required)	Description	Hidden by Default?
Patch	Patch number to report on or load. Required to load data.	No
Includes Bug	Patches containing this bug or subpatch. Can be specified only when the Load Data field is set to No.	No
*OraApps Release	Oracle release to report on. The default is 11i.	No

Table D-7. OraApps Patch Detail Report fields, continued

Field Name (*Required)	Description	Hidden by Default?
Application	Application to which the patch, subpatch, or bug belongs. Can be configured only when the Load Data field is set to No .	No
Environment	Environment to which the patch, subpatch, or bug belongs. Can be configured only when the Load Data field is set to No .	No
Applied Since	Patches applied on this date or later. Can be configured only when the Load Data field is set to No .	No
Includes File	Patches that include this file. Can be configured only when the Load Data field is set to No .	No
Includes File Version	Patches that include this file version. Can be configured only when the Load Data field is set to No .	No
*Order Files By	If files are listed, controls the sort order of the file records.	No
*Order Bugs By	If bugs are listed, controls the sort order of the bug records.	No
*Show SubPatch Info	Option for the report to show subpatch details.	No
*Show Env Deploy Info	Option for the report to show environment patch application details.	No
*File Details Display	Whether and how file details are included in the output. Options are None and Unique Files , that is, files associated with the patch.	No

Table D-7. OraApps Patch Detail Report fields, continued

Field Name (*Required)	Description	Hidden by Default?
*Load Data	Option to parse and load the C- or U-driver for the patch before running the report. Used to load or report on patches before applying them to any environment.	No
Patch Archive	If patch data is being loaded, the name of the .zip patch archive file where the C- or U-driver can be found. This archive should be in the patch repository (Patch Stage environment).	No
Remove Data after Run	Option to remove patch data after running the report if the patch data was loaded beforehand. Should be set to Yes if data is not required for additional reporting before application in OraApps environments. Can be configured only when the Load Data field is set to Yes .	No
C or U Driver File	If patch data is being loaded, name of the C- or U-driver file to extract from the Patch Archive. This driver file is used to load information regarding the patch.	No
Append Data	Option to evaluate the entire C- or U-driver if the patch already exists in the detail tables. The default value of No should not be changed under normal circumstances.	Yes

OraApps Patch Matrix Report

The following sections provide a description, configuration considerations, and field descriptions for the OraApps Patch Matrix Report.

Description

The OraApps Patch Matrix Report generates an annotated matrix of patches and environments, identifying the environments that have had patches applied. The patch levels of up to ten Oracle E-Business Suite environments can be compared. The report

fields control the appearance of the report and which details about each patch are presented.

If a patch has been applied to an instance, the report provides a link to the package that last applied the patch. If a patch has not been applied to an instance, the report provides links to all packages that could apply the patch.

Configuration Considerations

By default, the columns of the report alternate between red and green. To customize the colors, change the following statements in the `kacrpt_ora_patch_matrix.pls` file:

```
x_red := 'bgcolor=red';  
x_green := 'bgcolor=green';
```

For example, to change the red column to pink:

1. Change the first line to the following:

```
x_red := 'bgcolor=pink';
```


2. Recompile the procedure against the PPM Center schema by running the following:

```
sqlplus <ppm_user>/<ppm_pwd>@<ppm_database> @kacrpt_ora_patch_matrix.pls
```

Field Descriptions

"[Figure D-7. OraApps Patch Matrix Report](#)" shows the Submit Report window you use to specify the criteria for displaying patch details for environments. "[Table D-8. OraApps Patch Matrix Report fields](#)" provides descriptions of the report's displayed fields.

Figure D-7. OraApps Patch Matrix Report

 | HPE PROJECT AND PORTFOLIO MANAGEMENT CENTER

Submit Report: OraApps Patch Matrix

***Patch Type:**

***Environment 1:**

Environment 2:

Environment 3:

Environment 4:

Environment 5:

Environment 6:

Environment 7:

Environment 8:

Environment 9:

Environment 10:

Transaction Date From:

Transaction Date To:

App Code:

***Show Colors:** Yes No

***Show Performed by:** Yes No

***Show Date Applied:** Yes No

***Show Package Info:** Yes No

***Show Log File:** Yes No

***Show ReadMe File:** Yes No

***Show Heading:** Yes No

Run Report Immediately

Run Report On:

Repeat Every Hours Until

Send email to: when report is finished

Table D-8. OraApps Patch Matrix Report fields

Field Name (*Required)	Description
*Patch Type	Title of the report
*Environment 1	Environment to be reported on
Environment 2- Environment 10	Other environments to report on
Transaction Date From	Transactions that occurred on this date or later
Transaction Date To	Transactions that occurred on this date or earlier
App Code	Transactions performed on package lines using this App Code
*Show Colors	Option for the report to show colors to indicate whether the patch has been applied to the environment
*Show Performed by	Option for the report to show who applied the patch
*Show Date Applied	Option for the report to show information about the date the patch was last applied
*Show Package Info	Option for the report to show the package number and package line sequence number used to apply the patch
*Show Log File	Option for the report to show a link to the ADPATCH log file
*Show ReadMe File	Option for the report to show a link to the patch README file
*Show Heading	Option for the report to show headings for rows of data

OraApps Patches Applied for Specific Bug Report

The following sections provide a description, configuration considerations, and field descriptions for the OraApps Patches Applied for Specific Bug Report.

Description

The OraApps Patches Applied for Specific Bug Report lists the patches that have been applied to resolve the bug that a user specifies.

Configuration Considerations

If you want this report to display patches that have not been applied, you must do the following:

1. Run the OraApps Patch Detail Report with the Load Data field set to **Yes** and the **Remove Data after Run** field set to **No** (see "[OraApps Patch Detail Report](#)" on page 300).
2. Run the OraApps Patches Applied for Specific Bug Report with the **Include Non-Applied Patches** field set to **Yes**.

Field Descriptions

"[Figure D-8. OraApps Patches Applied for Specific Bug Report](#)" shows the Submit Report window you use to obtain details about patches applied (and, optionally, not applied) that include the bug you specify. "[Table D-4. OraApps Critical Requests Summary Report fields](#)" provides descriptions of the report's displayed fields.

Figure D-8. OraApps Patches Applied for Specific Bug Report

Hewlett Packard Enterprise | HPE PROJECT AND PORTFOLIO MANAGEMENT CENTER

Submit Report: OraApps Patches Applied for Specific Bug

Submit Cancel

Report Parameters Restore Default

*Report Title: OraApps Patches Applied for Specific Bug

*Bug:

Environment:

Applied Since:

*OraApps Release: 11i

*Include Non-Applied Patches? Yes No

Scheduling

Run Report Immediately

Run Report On: Repeat Every Hours Until

Send email to: Admin User when report is finished

Advanced Notifications

Submit Cancel

Close Window

Table D-4. OraApps Critical Requests Summary Report fields

Field Name (*Required)	Description	Hidden by Default?
*Report Title	Title of the report	No
*Bug	Report on only the patches that include this SubPatch/Bug.	No
Environment	Environment to be reported on.	No
Applied Since	Include in the report only the patches that have been applied to the environment on or after this date.	No
*OraApps Release	Oracle release to report on. The default is 11i.	No
*Include Non-Applied Patches	Option to include in the report patches for this bug that have not been applied to the specified Environment. See "Configuration Considerations" on the previous page.	No

Patch Application Comparison Report

The following sections provide a description and field descriptions for the Patch Application Comparison Report.

Description

The Patch Application Comparison Report is similar to the Environment Comparison by Objects Migrated Report in Deployment Management, but the Patch Application Comparison Report is specialized for Oracle patches. The Patch Application Comparison Report can be used to obtain a list of patches migrated to or applied to up to six different environments. The report lists each patch number and the last migration date for the patch into each environment, allowing you to determine whether a patch was applied out of order or was missed in one environment.

The Patch Type field for this report represents the object types used for patching. By default, the only available value is **Oracle Patch**. The user can extend the list of values available to the report by adding values to the OA -Patching Object Type Names validation, for example by doing the following:

- Modifying the name of an object type
- Adding special-purpose object types

The **Meaning** field of the validation entry must exactly match the name of the object type to be reported on.

Field Descriptions

"[Figure D-9. Patch Application Comparison Report](#)" shows the Submit Report window you use to specify which patches and environments you want the Patch Application Comparison Report to report on. "[Table D-10. Patch Application Comparison Report fields fields](#)" provides descriptions of the fields in the report that are displayed by default.

Figure D-9. Patch Application Comparison Report

Submit Report: Patch Application Comparison Report

Report Parameters

Patch Type:

Environment 1:

Environment 2:

Environment 3:

Environment 4:

Environment 5:

Environment 6:

Transaction Date From:

Transaction Date To:

App Code:

Performed by:

Order by:

Scheduling

Run Report Immediately

Run Report On:

Repeat Every Hours Until

Send email to: when report is finished

Advanced Notifications

Submit Cancel

Close Window

Table D-10. Patch Application Comparison Report fields fields

Field Name (*Required)	Description
*Patch Type	Patch type for which to obtain migration information (by default, the only available value is Oracle Patch)
*Environment 1	Environment to be reported on
Environment 2- Environment 6	Other environments to report on
Transaction Date From	Transactions that occurred on this date or later
Transaction Date To	Transactions that occurred on this date or earlier
App Code	Transactions performed on package lines using this App Code
Performed by	Transactions performed by this user
*Order by	Criterion to use to sort the report— Patch Number or Last Update Date

Patches Applied to an Environment Report

The following sections provide a description and field descriptions for the Patches Applied to an Environment Report.

Description

Similar to the Environment/Objects Detail Report in Deployment Management, the Patches Applied to an Environment Report lists Oracle patches migrated to or applied to a specific environment. You can use this report to view the history of Deployment Management executions for these patches. For example, you can determine whether an Oracle Application patch has been applied multiple times to the same environment. You can also use this report to obtain a list of all the patches applied in a specific date range or all the patches run after a specific patch was applied. This report also has Web links to execution logs and README files for each patch.

The **Patch Type** field for this report represents the object types used for patching. By default, the only available value is **Oracle Patch**. The user can extend the list of values available to the report by adding values to the OA -Patching Object Type Names validation, for example by doing the following:

- Modifying the name of an object type
- Adding special-purpose object types

The **Meaning** field of the validation entry must exactly match the name of the object type to be reported on.

Field Descriptions

"[Figure D-10. Patches Applied to an Environment Report](#)" shows the Submit Report window you use to specify which patches and environments you want the Patches Applied to an Environment Report to report on. "[Table D-11. Patches Applied to an Environment Report fields](#)" provides descriptions of the fields in the report that are displayed by default.

Figure D-10. Patches Applied to an Environment Report

Table D-11. Patches Applied to an Environment Report fields

Field Name (*Required)	Description
*Patch Type	Patch type for which to obtain migration information (by default, the only available value is Oracle Patch)
*Environment	Environment to be reported on

Table D-11. Patches Applied to an Environment Report fields, continued

Field Name (*Required)	Description
Transaction Date From	Transactions that occurred on this date or later
Transaction Date To	Transactions that occurred on this date or earlier
App Code	Transactions performed on package lines using this App Code
Performed by	Transactions performed by this user
*Order by	Criterion to use to sort the report— Patch Number or Last Update Date
Since Patch	Only migrations that have occurred after this patch number was applied to the specified environment

Pending Patches Report

The following sections provide a description and field descriptions for the Pending Patches Report.

Description

This functionality of the Pending Patches Report is a subset of the functionality of the Packages Pending Report in Deployment Management, specific to Oracle patches. The Pending Patches Report lists package lines for Oracle patches that are waiting at a migration step (that is, lines that have been approved for a given environment to which the patch must be applied). This report also has Web links to execution logs and README files if the patch has been applied.

The **Patch Type** field for this report represents the object types used for patching. By default, the only available value is **Oracle Patch**. The user can extend the list of values available to the report by adding values to the OA -Patching Object Type Names validation, for example by doing the following:

- Modifying the name of an object type
- Adding special-purpose object types

The **Meaning** field of the validation entry must exactly match the name of the object type to be reported on.

Field Descriptions

"[Figure D-11. Pending Patches Report](#)" shows the Submit Report window you use to specify which patches and environment you want the Pending Patches Report to report

on. "Table D-12. Pending Patches Report fields" provides descriptions of the fields in the report that are displayed by default.

Figure D-11. Pending Patches Report

Table D-12. Pending Patches Report fields

Field Name (*Required)	Description
*Patch Type	Patch type for which to obtain migration information (by default, the only available value is Oracle Patch)
Environment	Environment that has package lines for Oracle patches that are waiting at a migration step
App Code	Transactions performed on package lines using this App Code

Appendix E: Tokens

While configuring certain features in PPM Center, you often need to refer to information that is undefined until it is used in a specific context. Instead of generating entities that are valid only in specific contexts, you can use variables to create general entities that can be applied to a variety of contexts. These variables are called tokens.

Tokens can be used in (but are not limited to) the following PPM Center entities:

- Object types
- Request types
- Validations and SQL statements
- Report types
- Workflow executions and notifications
- Workflow steps

For more information about tokens, see ["List of Tokens" below](#) and the Commands, Tokens, and Validations Guide and Reference.

List of Tokens

["Table E-1. Tokens included in the Extension"](#) describes some of the standard tokens included in the Extension.

Table E-1. Tokens included in the Extension

Prefix	Token	Description
ENV.AC	OA_APPLMGR_PASSWORD	Application Manager password for the Oracle installation on the server host. This value is encrypted.
ENV.AC	OA_APPLMGR_USERNAME	Application Manager user name for the Oracle installation on the server host.
ENV.AC	OA_APPS_DB_LINK	Password for the APPLSYS schema of the environment's database. This value is encrypted.
ENV.AC	OA_APPS_USERNAME	User or schema name of the APPLSYS account of the environment's database.
ENV.AC	OA_APPS_VERSION	Version of the Oracle E-Business Suite installation (11 for Release 11, Release 11i, or Release 12).

Table E-1. Tokens included in the Extension, continued

Prefix	Token	Description
ENV.AC	OA_CONC_REQUEST_PASSWORD	Password used by PPM Center to retrieve Oracle Concurrent Request logs from the Oracle E-Business Suite instance. This value is encrypted. Required only if this instance will execute Object Migrator requests.
ENV.AC	OA_CONC_REQUEST_USERNAME	User name used by PPM Center to retrieve Oracle Concurrent Request logs from the Oracle Server. Required only if this instance will execute Object Migrator requests.
ENV.AC	OA_MT_APPSFORMS	Flag indicating whether forms are installed in the APPL_TOP of the Oracle installation on the middle-tier (client) host.
ENV.AC	OA_MT_CONC_PROGRAMS	Flag indicating whether concurrent programs are installed in the APPL_TOP of the Oracle installation on the middle-tier (client) host.
ENV.AC	OA_MT_DATABASE_FILES	Flag indicating whether database files are installed in the APPL_TOP of the Oracle installation on the middle-tier (client) host.
ENV.AC	OA_MT_SELFSESV_APPS	Flag indicating whether self-service applications are installed in the APPL_TOP of the Oracle installation on the middle-tier (client) host.
ENV.AC	OA_NCA_PASSWORD	Application Manager password for the Oracle installation on the middle-tier (client) host. This value is encrypted.
ENV.AC	OA_NCA_PATCH_TOP	Directory path of the NCA Patch Top on the middle-tier (client) host.
ENV.AC	OA_NCA_USERNAME	Application Manager user name for the Oracle installation on the middle-tier (client) host.
ENV.AC	OA_OM_DB_VALUE	Value identifying this Oracle instance to Object Migrator as recorded in the Oracle value set CLM_DATABASES.

Table E-1. Tokens included in the Extension, continued

Prefix	Token	Description
ENV.AC	OA_OM_INSTALLED	Flag indicating whether Object Migrator is installed in this environment's database.
ENV.AC	OA_REMOTE_OM_ENV_ID	ID of the environment containing the Object Migrator instance to use for AOL Object Migrations to this environment.
ENV.AC	OA_SERVER_APPSFORMS	Flag indicating whether forms are installed in the APPL_TOP of the Oracle installation on the server host.
ENV.AC	OA_SERVER_CONC_PROGRAMS	Flag indicating whether concurrent programs are installed in the APPL_TOP of the Oracle installation on the server host.
ENV.AC	OA_SERVER_DATABASE_FILES	Flag indicating whether database files are installed in the APPL_TOP of the Oracle installation on the server host.
ENV.AC	OA_SERVER_PATCH_TOP	Directory path of the NCA Patch Top for server files in the Oracle installation on the server host.
ENV.AC	OA_SERVER_SELFSEV_APPS	Flag indicating whether self-service applications are installed in the APPL_TOP of the Oracle installation on the server host.
ENV.AC	OA_SERVER_UTL_DIR_PATH	UTL directory path.
ENV.AC	OA_SYSTEM_PASSWORD	Password for the SYSTEM schema of the environment's database. This value is encrypted.
ENV.AC	OA_APPS_USERNAME	The Oracle Applications user name used by this PPM user.

Send Us Feedback



Let us know how we can improve your experience with the Deployment Management Extension for Oracle E- Business Suite Guide.
Send your email to: docteam@microfocus.com