

K I N T A N A™

Configuring a Deployment System in Kintana

Version 5.0.0

Publication Number: DeployConfig-0603A

Kintana, Inc. and all its licensors retain all ownership rights to the software programs and related documentation offered by Kintana. Use of Kintana's software is governed by the license agreement accompanying such Kintana software. The Kintana software code is a confidential trade secret of Kintana and you may not attempt to decipher or decompile Kintana software or knowingly allow others to do so. Information necessary to achieve the interoperability of the Kintana software with other programs may be obtained from Kintana upon request. The Kintana software and its documentation may not be sublicensed and may not be transferred without the prior written consent of Kintana.

Your right to copy Kintana software and this documentation is limited by copyright law. Making unauthorized copies, adaptations, or compilation works (except for archival purposes or as an essential step in the utilization of the program in conjunction with certain equipment) is prohibited and constitutes a punishable violation of the law.

THIS DOCUMENTATION IS PROVIDED "AS IS" WITHOUT WARRANTY OF ANY KIND. IN NO EVENT SHALL KINTANA BE LIABLE FOR ANY LOSS OF PROFITS, LOSS OF BUSINESS, LOSS OF USE OR DATA, INTERRUPTION OF BUSINESS, OR FOR INDIRECT, SPECIAL, INCIDENTAL, OR CONSEQUENTIAL DAMAGES OF ANY KIND, ARISING FROM ANY ERROR IN THIS DOCUMENTATION.

Kintana may revise this documentation from time to time without notice.

Copyright © 1997, 1998, 1999, 2000, 2001, 2002, 2003 Kintana, Incorporated. All rights reserved.

Kintana, Kintana Deliver, Kintana Create, Kintana Drive, Kintana Dashboard, Kintana Accelerator, Kintana Demand Management (DM), Kintana Portfolio Management (PFM), Kintana Program Management Office (PMO), Kintana Enterprise Change Management (ECM), Object*Migrator, GL*Migrator and the Kintana logo are trademarks of Kintana, Incorporated. All other products or brand names mentioned in this document are the property of their respective owners.

Kintana Version 5.0.0

© Kintana, Incorporated 1997 - 2003
All rights reserved.
Printed in USA

Kintana, Inc.
1314 Chesapeake Terrace, Sunnyvale, California 94089
Telephone: (408) 543-4400
Fax: (408) 752-8460
<http://www.kintana.com>

Contents

Chapter 1	
Introduction	11
Who should read this guide	12
How to use this guide	12
What this guide is NOT	13
Additional Resources	13
Kintana Documentation	13
<i>Kintana Business Application Guides</i>	14
<i>User Guides</i>	14
<i>Kintana Application Reference Guides</i>	14
<i>Kintana Instance Administration Guides</i>	15
<i>External System Integration Guides</i>	15
<i>Kintana Solution Guides</i>	15
<i>Kintana Accelerator Guides</i>	16
Kintana Services	16
Kintana Education	16
Kintana Support	17
.....	17
Chapter 2	
Key Concepts	19
Kintana Interface and the Workbench	19
Deployment	20
Object Types	20
Package	22
Package Lines	23
Workflow	24
Object Type - Workflow Integration	25
Environments	27
Environment Groups	27
Commands	28
Special Commands	28

Validations	29
Tokens	29
Security Groups	30
Participants	31
Integration with Kintana Products and Solutions	32
Kintana Solutions	32
Kintana Dashboard	33
Kintana Drive.....	33
Kintana Create.....	34
Kintana Accelerators	34
Reports	35
Chapter 3	
Developing Your Kintana Configurations (Using Migrators)	37
Using Multiple Kintana Instances - Overview	38
Single PRODUCTION instance is currently in use.	39
New Kintana Implementation	39
Migrating your Kintana Configurations	40
How Kintana Migrators Work	40
Using the Kintana Migrators - Overview	41
Instance Requirements for Using Kintana Migrators.....	42
Kintana Requirements for PROD Instance.....	42
Archiving your Kintana Configurations.....	43
Chapter 4	
Configuring your Deployment System - Process Overview.....	45
Example: Configuring a Deployment System.....	46
Chapter 5	
Gathering Process Requirements and Specifications.....	49
Define the Deployment Process	50
Process (Workflow) Considerations.....	50
Decision versus Execution steps.....	51
<i>Decision Steps</i>	<i>51</i>
<i>Execution Steps.....</i>	<i>51</i>
Immediate Versus Manual Executions	51
Timeouts	52
Define the Business Flow	52
Example: Defining the Business Flow	52
<i>Example: ACME defines a high-level process flow.....</i>	<i>53</i>
Defining the Technical Flow.....	54
Example: Defining the Technical Flow.....	55

<i>Example: Detailed Process</i>	55
Gather Information on Each Step in the Process	59
<i>Example: Gathering Workflow Step Information</i>	60
Consider Using Subworkflows	61
<i>Example: Using a Subworkflow</i>	62
Consider Using Kintana Release Management.....	62
<i>Example: ACME Uses Release Management in their Deployment Process.</i>	63
Determine Information to Describe Objects	64
<i>Example: ACME collects information on their objects</i>	65
Determine Commands Needed for Objects	68
<i>Example: High level Command design</i>	68
Gather Information on Environments	69
<i>Example: ACME specifies the Environments</i>	70
Identify Participants and Security	71
<i>Example: ACME determines participants and security</i>	73
Establish Communication Points and Visibility	76
<i>Example: ACME configures notifications</i>	77

Chapter 6

Mapping your Process into a Kintana Workflow	79
Building the Workflow Skeleton - Overview	79
Required Workflow Settings for Deployment Process	80
Create the Required Step Source	81
Creating a Workflow Step Source - Overview	82
Workflow Step Source Configuration and Usage Restrictions	85
Creating a Decision Type Step.....	85
Enter the general information on the Decision step source	86
Select a Validation.....	88
Specify the voting requirements on the step.....	88
Specify the default timeout value	89
Create an Execution Type Step	90
Enter the general information on the Execution step source	91
Define the Executions	94
<i>Execute the Object Type Commands</i>	95
<i>Close the Package Line and mark it as a Success</i>	97
<i>Close the Package and mark it as Failed</i>	99
<i>Transition (jump) to a Workflow that is Processing a Request</i>	101
<i>Receive control from a Workflow that is Processing a Request</i>	101
<i>Set a Package "Ready for Release" for use in Kintana's Release Management</i>	101
<i>Return from a Subworkflow to the Parent Workflow</i>	102
<i>Execute a PL/SQL function and then transition based on the result</i>	103
<i>Execute a SQL statement and then transition based on the result</i>	104
<i>Evaluate a Token and then transition based on the result</i>	104
<i>Execute a number of system level commands and then transition based on the success</i>	

<i>or failure of those commands.</i>	106
Select a Validation.....	109
Specify the default timeout value.....	109
Configure the Step's Transition Values (Validation)	109
Validations and Execution Type Relationships.....	111
Add Steps and Transitions to the Workflow Layout	112
Adding Decision Steps.....	113
Enter the general information on the Decision step	113
Specify the Security.....	115
Configure Notifications for the Workflow Step	116
Adding Execution Steps	117
Enter the general information on the Execution step.....	118
Specify the Security.....	119
Configure Notifications for the Workflow Step	120
Adding a Subworkflow	121
Adding Transitions Between Steps	122
Transition based on a specific result	123
Transition based on a value in a field.....	123
Transition based on data in a table.....	125
Transition based on all but one specific value	125
Transition based on all results.....	126
Transition based on error	127
Transition back to the same step	130
Transition based on a previous workflow step result (parameters)	131
<i>Example: Using a Workflow Parameter to Transition.....</i>	<i>131</i>
Transition to and from Subworkflows.....	134
Transition to and from a Request Workflow	134

Chapter 7

Constructing the Object Type	135
Creating an Object Type - Overview	136
Creating Object Type Fields	137
Determining the Field Type (Selecting a Validation).....	139
Available Field Types.....	139
Selecting the Validation	141
Building a Validation	142
<i>Auto-Complete Versus Drop Down List.....</i>	<i>143</i>
<i>Tips for Configuring Validations.....</i>	<i>144</i>
Configuring Field Behavior	144
Configuring Field Dependencies	148
Using Commands to Change Field Values	150
Modifying the Object Type Layout	150
Changing a Column Width	150
Moving Fields.....	151
Setting the Object Name.....	152

Setting the Object Revision.....	153
Copying Object Type Fields	154
Editing Object Type Fields.....	155
Removing Fields	155
Creating Object Type Commands	156
Object Type Commands Overview.....	156
Commands Interface.....	156
Object Type Commands and Workflow	159
Kintana Special Commands	159
Command Steps.....	160
Command Conditions.....	161
Example Object Type Command Uses.....	162
Chapter 8	
Defining your Environments.....	163
Environment Requirements.....	163
Defining Environments in Kintana	164
Copying Environments	167
Selecting the Environment's Connection Protocol	167
Selecting the Environment's Transfer Protocol.....	168
Using App Codes with Your Environment	170
Copying App Codes from Other Environments.....	173
Setting the Access for Environments	174
Creating Environment Groups	176
When to Use Environment Groups	176
Defining an Environment Group	176
Setting Ownership for Environment Groups	180
Setting the Access for Environments.....	181
Copying an Environment Group	183
Adding Environments to an Environment Group.....	184
Removing Environments from an Environment Group	186
Setting the Environment Execution Order.....	187
Linking Environments and Environment Groups to Workflows	188
Choosing the Source Environment Based on Selected App Code	189
Environment Maintenance and Utilities	190
Testing the Environment Setup	190
Mass Update of Base Paths	191
Environment Password Management Utility	192
Updating Passwords Using the Kintana Workbench Interface	192
Updating Passwords Using the Command Prompt	193
Deleting Environments	194
Chapter 9	
Integrating Participants into Your Deployment System	195

User Security and Participation - Overview	195
Mapping the Worksheet to Participant Security	197
Establishing Security Groups	197
Creating a Security Group by Specifying a List of Users	198
Using Kintana's Resource Management to Control User Security	201
Setting Package Creation Security	202
Enabling Users to Create Packages.....	203
Restricting Users from Selecting a Specific Workflow.....	204
Restricting Users from Selecting a Specific Object Type	205
Setting Package Processing Security	206
Providing Users with General Access to Update Packages	206
Enabling Users to Act on a Specific Workflow Step.....	207
Restricting Package Processing to Participants.....	209
Setting Configuration Security	210
Setting Ownership for Kintana Configuration Entities	211
Removing Access Grants.....	213

Chapter 10

Setting Up Communication Paths	215
Adding Notifications to Workflow Steps	216
Adding a Notification to a Workflow step - Overview	216
Configuring When to Send a Notification	218
Sending a notification when a step becomes eligible	218
Sending a notification when a step has a specific result	219
Sending a notification when the step has a specific error	221
<i>Specific Errors for Workflow Steps</i>	222
Configuring multiple notifications for a single step.....	223
Specifying the Time the Notification is Sent.....	224
<i>Configuring the Notification Intervals</i>	225
Sending a follow up notification (reminder)	227
Configuration Tip: Sending a Notification Based on a Field Value.	228
Configuring the Notification Recipients	229
Recipient Configuration Tips	231
Configuring the Notification Message	231
Using Tokens in the Message Body.....	233
<i>Special Case: Tokens in HTML Message</i>	233
Including URLs to Open the Package (Smart URLs)	234
<i>Smart URLs in an HTML Formatted Messages</i>	235
Configuring Your Dashboard	236
Controlling User Access to Portlets	237
Disabling Portlets.....	237
Restricting User Access.....	238
Creating and Distributing a Default Dashboard	240
Creating Custom Portlets	240

Kintana Portlets to Enable for Use on the Dashboard.....	241
Configuring Reports	241
Chapter 11	
Rolling Out Your Deployment Process	243
Test the Deployment System - Checklists	243
General Deployment System Configuration Checklist.....	244
Workflow Checklist	246
Object Type Checklist	249
Environments Checklist.....	250
Security / User Access Checklist	251
Dashboard / Portlet Checklist	252
Cross-Entity Checklist.....	253
Migrate Your Configuration Data into Production.....	254
Enable Entities and User Access	255
Educating Your Users	255
Additional Resources for Education and Roll-Out.....	256
<i>Kintana Education and Online courses</i>	<i>256</i>
<i>Site Help</i>	<i>256</i>
<i>Online Help.....</i>	<i>256</i>
Appendix A	
Advanced Workflow Topics.....	257
Using Subworkflows.....	257
Transitioning to a Subworkflow	258
Transitioning From a Subworkflow.....	260
Package - Request Workflow Integration.....	262
Setting Up the 'WF - Jump/Receive Step Labels' Validation	264
Generating a Jump Step Source.....	266
Generating a Receive Step Source	267
Including the Jump/Receive pair in Workflows.....	269
Using Condition Steps.....	271
AND	271
OR	272
SYNC.....	272
FIRST LINE	273
LAST LINE.....	275
Setting the Reopen Step for Request Workflows	275
Modifying Workflows in Use.....	276
Copying and Testing the Workflow	277
Moving Requests Out of a Step.....	277
Disabling a Workflow Step.....	278

<i>Redirecting the Workflow</i>	278
Setting Up Execution Steps	279
Modifying Workflow Step Security – Performance Consideration.....	279
Verifying Workflow Logic.....	280
Using Workflow Parameters	280
Creating a Workflow Parameter	280
Example: Building a Loop Counter	281

Appendix B

Validations	287
What are Validations	288
Validation Component Types - Overview	288
Creating a Validation	291
User Data on the Validation Value.....	291
Editing Validations	293
Creating a URL to Open the Validation Window.....	294
Deleting Validations	295
Static List Validations.....	295
Dynamic List Validations.....	297
SQL Validation	297
SQL Validation Tips	299
Command Validation	299
Using Auto-Complete Validations	300
Validation by Command With Delimited Output	301
Validation by Command With Fixed Width Output.....	304
User-Defined Multi-Select Auto-Complete Fields.....	306
Example: Token Evaluation and Validation by Command with Delimited Output	307
Special Case - Limiting the Number of Returned Rows	310
Using Directory and File Choosers	312
Directory Chooser	313
File Chooser	313
Creating 1800 Character Text Areas	316
Configuring the Table Component	316
Define the Table Component in the Validation Workbench	317
Creating a Table Rule	320
Example: Using a Table Component on an Order Form.....	321
<i>Tokens in the Table Components</i>	325
Calculating Column Totals	325
Add the Table Component to a Request Type.....	327
Package and Request Group Validations.....	329
Package and Request Groups.....	329
Request Type Category	330

Validation Special Characters	331
System Validations	331
.....	347
Appendix C	
Tokens	349
Chapter 12	
User Data Creation and Processing.....	351
Creating and Editing Kintana User Data	352
Adding User Data Fields.....	353
Copying a Field's Definition	354
Editing User Data Fields.....	355
Configuring User Data Field Dependencies	356
Removing Fields	358
Modifying the User Data Layout	359
Changing Column Width	360
Moving a Field	360
Swapping Positions of Two Fields.....	361
Previewing the Layout	361
Creating and Editing Context Sensitive User Data	362
Creating Context Sensitive User Data	363
Defining the Context Field	363
Defining a Context Value	364
Defining the Context Sensitive Fields.....	365
Editing Context Sensitive User Data	365
Changing the Context Field	365
Changing the Context Value.....	366
Editing Context Sensitive Fields	367
Deleting Context Sensitive User Data.....	367
Copying Context Sensitive User Data	367
Example - Using Context Sensitive User Data for a Field in a Request Header Type	368
Setting Up the Context Sensitive User Data	369
Example: Configuring the Validations	370
Example: Modifying the SQL	372
Example: Resulting Behavior	373
Project/Task User Data Roll-Up.....	375
Creating Project/Task User Data Roll-Up.....	375
Example: Using Project/Task User Data Roll-Up	376
Editing Project/Task User Data Roll-Up	379
Deleting Project/Task User Data Roll-Up	381
Example: Creating and Using Project/Task User Data Roll-Up	382
Referring to User Data	388
Migrating User Data	388

Migrating User Data Values	388
Migrating User Data Contexts	389
Chapter 13	390
Appendix D Configuration Worksheets	391
Participant and Security	398

Chapter 1 Introduction

Kintana Deliver allows companies to automate and manage the deployment of packaged applications, custom applications, legacy systems, Web content, and more. Kintana enforces deployment processes and performs all the tasks required to install software changes correctly across the Development, Test, Stage, and Production system landscape.

This document provides instructions for configuring a deployment system using Kintana. This includes requirements gathering, modeling your processes in a Kintana Workflow, defining commands used by Kintana's execution engine, and rolling out this system to your users.

This document discusses the following topics:

- *Key Concepts*
- *Developing Your Kintana Configurations (Using Migrators)*
- *Configuring your Deployment System - Process Overview*
- *Gathering Process Requirements and Specifications*
- *Mapping your Process into a Kintana Workflow*
- *Constructing the Object Type*
- *Defining your Environments*
- *Integrating Participants into Your Deployment System*
- *Setting Up Communication Paths*
- *Rolling Out Your Deployment Process*
- *Validations*
- *Tokens*
- *User Data Creation and Processing*

- [Configuration Worksheets](#)

Who should read this guide

This document provides details for defining, configuring, and rolling out a deployment system in Kintana.

This business application guide is used primarily by:

- Business or technical users who configure and maintain a distribution and deployment system using Kintana (Kintana Deliver)
- Users responsible for deploying software and applications using Kintana
- Managers responsible for reporting on software and application deployments
- Release Managers



Note

You must have a Deliver Power license to access the screens and windows described in this document. You must also belong to a Security Group with the correct access grants in order to define and process Releases. See "[Kintana Security Model](#)" for details.

How to use this guide

This document provides background information and details for configuring Kintana to manage your distribution and deployment processes. Navigate to the desired topic using the Table of Contents or use the Index to find information related to key words.

If viewing this guide online, you can use the Kintana Library page's search functionality to quickly locate desired topics in this and other Kintana publications.

What this guide is NOT

This business application guide is not meant to provide detailed information on every screen and field in Kintana. Nor will this document provide comprehensive instructions on configuring your Object Type commands. For detailed screen and field information refer to the Kintana Application Reference Guides, accessible from the Kintana Library. See [“Additional Resources”](#) on page 13 for a list of the most relevant documents.

Additional Resources

Kintana provides the following additional resources to help you successfully implement, configure, maintain and fully utilize your Kintana installation:

- [Kintana Documentation](#)
- [Kintana Services](#)
- [Kintana Education](#)
- [Kintana Support](#)

Kintana Documentation

Kintana product documentation is linked from the Kintana Library page. This page is accessed by:

- Selecting **HELP > KINTANA LIBRARY** from the Kintana Workbench menu.
- Selecting **HELP > CONTENTS AND INDEX** from the menu bar on the HTML interface. You can then click the **KINTANA LIBRARY** link to load the full list of product documents.

Kintana organizes their documents into a number of user-based categories. The following section defines the document categories and lists the documents currently available in each category.

- [Kintana Business Application Guides](#)
- [User Guides](#)
- [Kintana Application Reference Guides](#)
- [Kintana Instance Administration Guides](#)
- [External System Integration Guides:](#)

- [Kintana Solution Guides](#)
- [Kintana Accelerator Guides](#)

Kintana Business Application Guides

Provides instructions for modeling your business processes in Kintana. These documents contain process overviews, implementation instructions, and detailed examples.

- Configuring a Request Resolution System (Create)
- Configuring a Deployment and Distribution System (Deliver)
- Configuring a Release Management System
- Configuring the Kintana Dashboard
- Managing Your Resources with Kintana
- Kintana Reports

User Guides

Provides end-user instructions for using the Kintana products. These documents contain comprehensive processing instructions.

- Processing Packages (Deliver) User Guide
- Processing Requests (Create) User Guide
- Processing Projects (Drive) User Guide
- Navigating the Kintana Workbench:
Provides an overview of using the Kintana Workbench
- Navigating Kintana:
Provides an overview of using the Kintana (HTML) interface

Kintana Application Reference Guides

Provides detailed reference information on other screen groups in the Kintana Workbench. Also provides overviews of Kintana's command usage and security model.

- Reference: Using Commands in Kintana
- Reference: Kintana Security Model

- Workbench Reference: Deliver
- Workbench Reference: Configuration
- Workbench Reference: Create
- Workbench Reference: Dashboard
- Workbench Reference: Sys Admin
- Workbench Reference: Drive
- Workbench Reference: Environments

Kintana Instance Administration Guides

Provides instructions for administrating the Kintana instances at your site. These documents include information on user licensing and archiving your Kintana configuration data.

- Kintana Migration
- Kintana Licensing and Security Model

External System Integration Guides:

Provides information on how to use Kintana's open interface (API) to access data in other systems. Also discusses Kintana's Reporting meta-layer which can be used by third party reporting tools to access and report on Kintana data.

- Kintana Open Interface

Kintana Solution Guides

Provides information on how to configure and use functionality associated with the Kintana Solutions. Each Kintana Solution provides a User Guide for instructions on end-use and a Configuration Guide for instructions on installing and configuring the Solution.

Kintana Accelerator Guides

Provides information on how to configure and use the functionality associated with each Kintana Accelerator. Kintana Accelerator documents are only provided to customers who have purchased a site-license for that Accelerator.



Note

Kintana provides documentation updates in the Download Center section of the Kintana Web site (http://www.kintana.com/support/download/download_center.htm).

A username and password is required to access the Download Center. These were given to your Kintana administrator at the time of product purchase. Contact your administrator for information on Kintana documentation or software updates.

Kintana Services

Kintana is a strategic partner to its clients, assisting them in all aspects of implementing a Kintana technology chain - from pilot project to full implementation, education, project turnover, and ongoing support. Our Total Services Model tailors solution and service delivery to specific customer needs, while drawing on our own knowledgebank and best practices repository. Learn more about Kintana Services from our Web site:

<http://www.kintana.com/services/services.shtml>

Kintana Education

Kintana has created a complete product training curriculum to help you achieve optimal results from your Kintana applications. Learn more about our Education offering from our Web site:

<http://www.kintana.com/services/education/index.shtml>

Kintana Support

Kintana provides web-based interactive support for all products in the Kintana product suite via Contori.

<http://www.contori.com>

Login to Contori to enter and track your support issue through our quick and easy resolution system. To log in to Contori you will need a valid email address at your company and a password that will be set by you when you register at Contori.

Chapter 2 Key Concepts

The following key concepts and definitions are used when creating a deployment system in Kintana.

- *Kintana Interface and the Workbench*
- *Deployment*
- *Object Types*
- *Package*
- *Workflow*
- *Object Type - Workflow Integration*
- *Environments*
- *Commands*
- *Validations*
- *Tokens*
- *Security Groups*
- *Participants*
- *Integration with Kintana Products and Solutions*
- *Reports*

Kintana Interface and the Workbench

The Kintana Product Suite features two interfaces: the Kintana interface and the Kintana Workbench interface. The Kintana interface uses HTML and

Javascript to provide users with access to many key areas of functionality. The Kintana interface lets users of each product in the suite perform common tasks without requiring a Power License.

The Kintana Workbench is a Java applet designed to help Kintana Administrators, product configurers, and Power Users to perform advanced configuring and processing tasks, such as creating entities (like Request Types, Object Types, and Workflows). The Kintana Workbench can also query detailed information on a specific entity, such as a particular Package. Most entities within the Kintana Product Suite can be accessed through the Workbench.

When configuring your deployment system in Kintana, you will work primarily in the Kintana Workbench. Also, end-user Package creation and the majority of Package processing will occur within the Workbench. See "[Using the Kintana Workbench](#)" for information on standard Workbench navigation techniques.

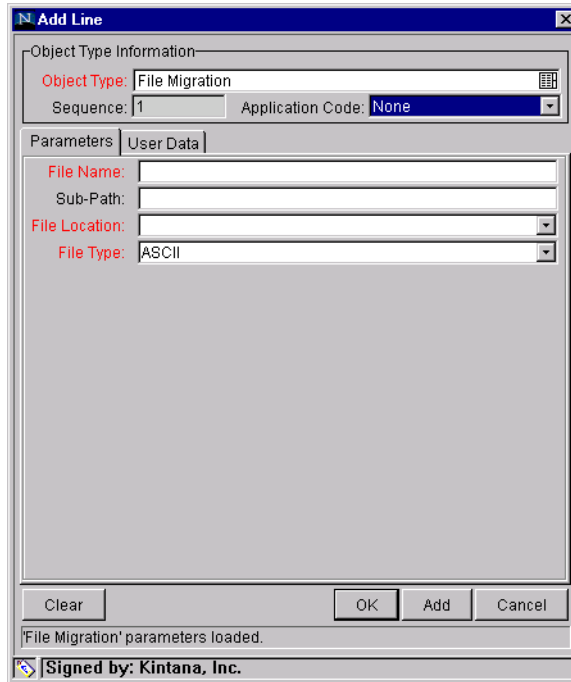
Deployment

Kintana can be configured to automate your deployment processes. Deployment is the act of moving an object or objects (file, script, code, full application, etc.) between two or more instances. For example, you can deploy a file from a development instance to a testing instance and finally into one or more production instances. Deployment typically involves connecting from one machine to another, moving files, and running any required scripts or compilers.

Object Types

Kintana Deliver automates complex software deployment processes. While Kintana Workflows define the process, Object Types are used to define the technical steps required to deploy a particular object. For example, a FILE MIGRATION Object Type may contain the information and commands required to transfer a file from one machine to another, while a SQL SCRIPT Object Type might address the migration and execution of database scripts.

Object Types are used by users who create and process Packages. Each Package Line in a Package consists of one object of a specific Object Type. When defining a Package Line, the user will select an Object Type in the ADD LINE window in the Package screen. Fields dynamically appear that are required to process that type of object.



Each Object Type contains:

- Object Type Fields
- Object Type Commands

Object Type fields describe the object – what it is, where it is, its name, what needs to be done to it, if it needs to be compiled, etc. You can configure the field prompts, tokens, behaviors and validations for each Object Type. For example, to migrate a file, Kintana must know the file name. A field named FILE NAME can be created to capture that information.

Object Type Commands are instructions interpreted by the Kintana Execution Engine and translated into operating system commands to be dynamically executed. Object Type commands are typically a blend between shell scripts and Kintana system Special Commands. Object Type Commands allow the automation of an entire sequence of commands that would previously have been run manually. For example, these command sequences

can automate source code compilation, check files into version control, or run a report.

Package

Kintana gathers all information required for a successful deployment (such as information on environments and objects to be migrated) into a single logical unit called the **Package**. The Package, consisting of the migrating objects, is then processed through a business Workflow. This results in a successful, easy-to-track software or application change.



A Package:

- Is the fundamental work unit of Kintana Deliver.
- Represents a logical unit of objects that should be moved and tracked together.
- Contains all the information needed to process the Package, including the Package Lines, priority, and status.
- Specifies the Workflow to be used to deploy the change.
- Contains a list of all objects to be tracked and/or migrated as the Package moves through its Workflow.

A Package consists of objects, each of which is on a separate Package Line. While each line can be acted upon separately, the group of Package Lines (objects) represent a logical unit that should be moved and tracked together. The processing of a Package and Package Lines can vary greatly depending upon the Workflow specified for that Package. *Figure 2-1* shows a sample Package.

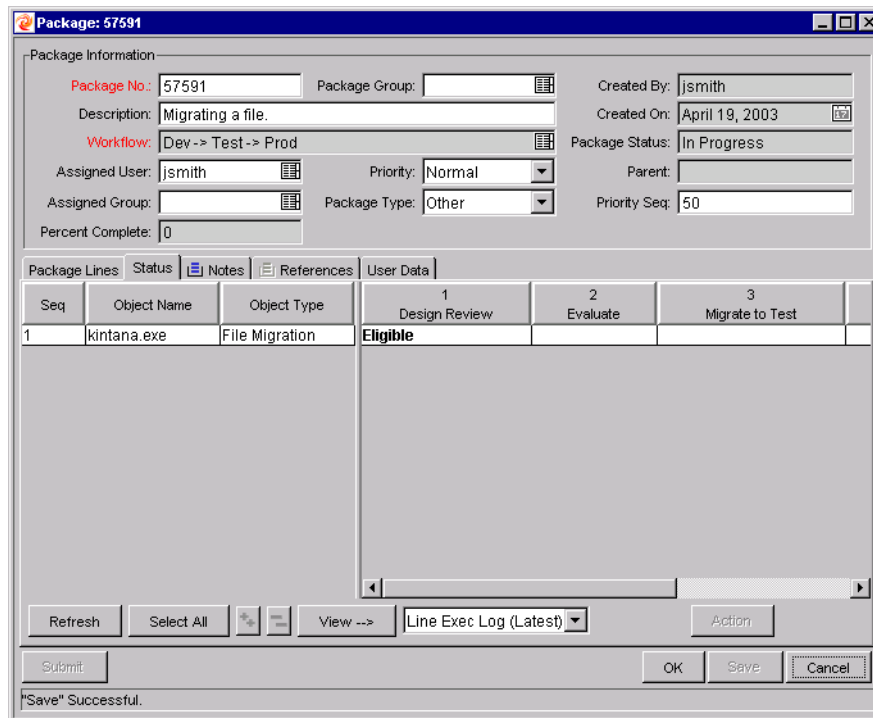


Figure 2-1 Sample Package

Package Lines

Packages can be used to deploy multiple objects. Each object is specified on a separate Package Line.

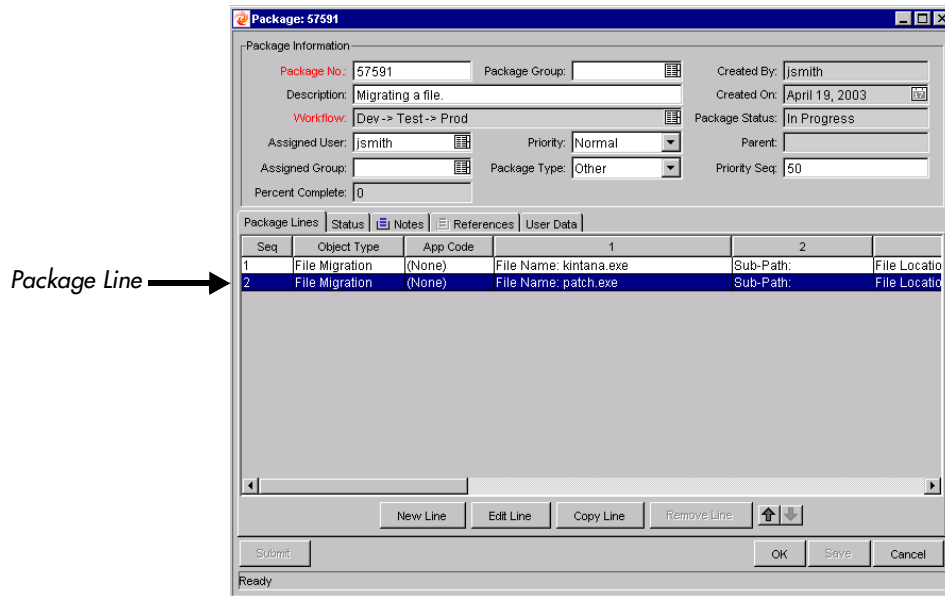


Figure 2-2 Sample Package Line

You can configure your Workflow to process different types of objects (Package Lines) along different processes. For example, you might want your Java code to undergo more approvals than a basic README file. The processing of a Package and Package Lines can vary greatly depending on the Workflow specified for that Package.

Workflow

A Workflow consists of a logical series of steps that define the path followed by objects (Package Lines) in a Package. Workflow configuration and routing is a customizable feature of Kintana. The Workflow engine can handle virtually any business practice. This allows a department to generate Workflows to automate existing processes, rather than forcing users to adopt a new set of processes to perform their work.

Workflow steps can range in usage from functional approvals to actual migrations. For example, you can create a migration step to automatically move specified objects from your source Environment to the destination Environments. Environments are also configured in Kintana.

A sample Workflow is shown in [Figure 2-3](#):

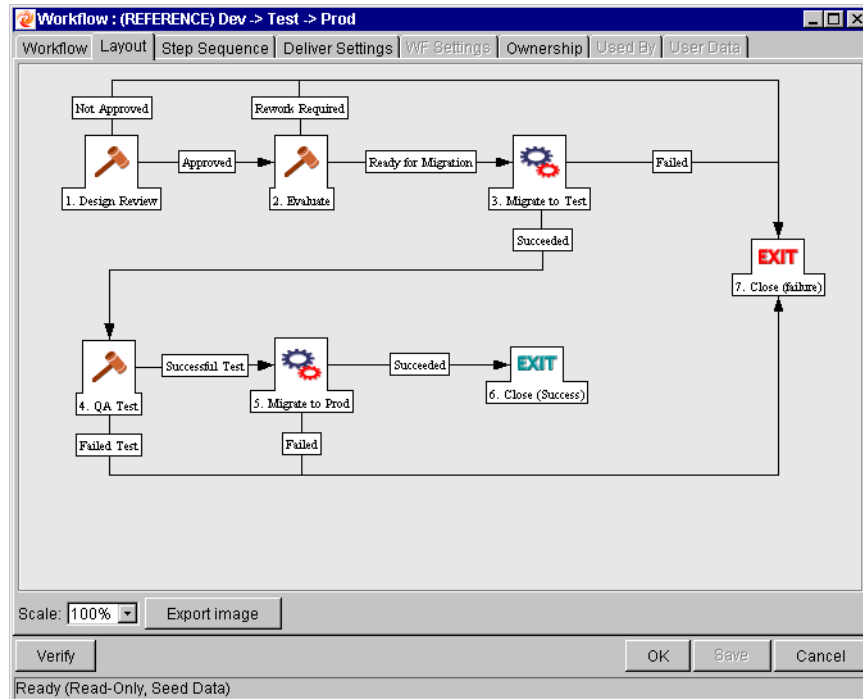


Figure 2-3 DEV -> TEST -> Prod Workflow

Object Type - Workflow Integration

Object Types are tightly integrated with the Kintana Workflow engine. You can configure the Workflow to execute the commands contained in the Object Type at specific points in the process (Workflow step). The Object Type commands are executed at EXECUTION Workflow steps.

It is important to note the following concepts regarding Command/Workflow interaction:

- To execute Object Type commands at a particular Workflow step, the Workflow step must be configured with the following parameters:
 - o Workflow step must be an EXECUTION type step.
 - o WORKFLOW SCOPE = **PACKAGES**.

- o EXECUTION TYPE = **BUILT-IN WORKFLOW EVENT**.
- o WORKFLOW COMMAND = **EXECUTE_OBJECT_COMMANDS**.
- When the object (Package Line) reaches the Workflow step (with WORKFLOW COMMAND = **EXECUTE_OBJECT_COMMANDS**), all of the Object Type commands whose conditions are satisfied will be run in the order they are entered in the Object Type's command panel.
- The Object Type can be configured to run only certain Commands at a particular step. To do this, specify a Command Condition (see [“Creating Object Type Commands”](#) on page 156 for details).
- Each Object Type command can be configured so that only certain steps (within a command) are executed within a particular Workflow step. This is done using conditional statements within the actual commands.



Example

When a file is migrated from one location to another, it may be necessary to change directories (`cd`) on either the source or the destination machines. If the command is programmed to automatically `cd` to a directory that does not exist, an error message appears and the migration will be cancelled. To avoid this, the following conditional statement can be used to ensure that the desired directory exists before the `cd` command is issued:

```
if [ ! -d [P.P_SUB_PATH] ]; then mkdir -p [P.P_SUB_PATH]; fi
cd [P.P_SUB_PATH]
```

Setting up commands to run in Kintana is a two step process.

1. Include a step in the Workflow that will execute the commands (**EXECUTE_OBJECT_COMMANDS**).
2. Program the commands (within the Object Type) to be run at the designated Workflow step.

Environments

As a software deployment application, one of Kintana's primary responsibilities is to migrate and execute file system objects. To automate the migration of these objects, Kintana must have knowledge of the sources and destinations for the various objects. This data is stored in Kintana Environments, which are then referenced through Workflows and Object Types.

A Kintana Environment consists of a server, a single database instance, and an associated remote client machine. Not all of these components need be present in a single Environment. (For example, it is possible to have an Environment which does not contain a database).

When migrating objects, Kintana Deliver connects to remote computers in the same way any other user would (using FTP, SCP, SSH or Telnet). Kintana Deliver can logon using any existing username and password. However, it is recommended that a new user 'Kintana' be generated on each computer that Kintana Deliver will access. This will help clarify the setup and relieve some administrative burden. The 'Kintana' user should have full access to the Kintana home directory as well as the correct read and write permissions on other required directories. In addition, on Windows NT computers, the 'Administrators' group must have read access to Kintana's home directory. (Any Windows NT computer that Kintana Deliver will access should have been configured as directed in Kintana Installation Guide, which is available on the Kintana Web Site).

Environment Groups

Situations may arise where it is desirable to execute a Workflow Step on multiple Environments. For example, it may be necessary to migrate an object to multiple testing Environments for different targeted tests. These multiple Environments can be referenced together in one Environment Group.

Environment Groups define a set of Kintana Environments which can be referenced as the Source or Destinations for object migrations and executions. Environment Groups are defined and edited using the Environment Group Workbench.

Commands

Commands are instructions interpreted by the Kintana Execution Engine and translated into operating system commands to be dynamically executed. Commands are typically a blend between shell scripts and Kintana system Special Commands. Commands in Kintana allow the automation of an entire sequence of commands that would previously have been run manually. For example, these command sequences can automate source code compilation, checking files into version control, or running a report.

When configuring a deployment system, most of your Commands will be included in Object Type and Workflow step definitions. Commands can be used with the following entities in Kintana:

- Object Types
- Workflow Steps
- Request Types
- Report Types
- Validations

See "[Using Commands and Tokens](#)" for more instructions and examples on using Commands in Kintana.

Special Commands

In order to simplify programming commands, Kintana provides a predefined set of Special Commands. These commands perform a variety of common functions, such as copying files between environments and establishing connections to environments for remote command execution. Kintana features two types of Special Commands:

- **SYSTEM SPECIAL COMMANDS** - These commands are shipped with Kintana. System Special Commands are read-only and have the naming convention "KSC_COMMAND_NAME." System Special Commands always begin with "KSC_."
- **USER DEFINED SPECIAL COMMANDS** - These commands are user-defined and have the naming convention "SC_COMMAND_NAME." User-defined Special Commands must begin with "sc_." User defined Special Commands can contain one or more of Kintana's system Special Commands.

Validations

Validations determine the acceptable input values for user-defined custom fields. Validations maintain data integrity by ensuring that the correct information is entered in a field before it is saved to the database. For example, Validations can be used to ensure that no textual information is entered into a numeric field, or that dates are entered into a field in the proper format. More complex Validations can be used to verify that only appropriate Kintana users are assigned to a task. The values in selection Validations (drop down lists and auto-complete lists) can be configured by either listing the values or performing a SQL query.

Validations are used throughout Kintana:

- Every custom field generated for an Object Type, Report Type, Request Type, or User Data has a Validation.
- Every decision and execution Workflow Step has a Validation.
- Every drop down list and auto-complete list in all Kintana windows are based upon a Validation. However, it is not always possible to change the Validations associated with predefined fields.

Tokens

While configuring certain features in Kintana, it is often necessary to reference information in variables that is undefined until Kintana is actually used a particular context. Instead of generating objects that are valid only in those specific contexts, these variables can be used to facilitate the creation of general objects that can be applied to a variety of contexts. These variables are called **Tokens**.

There are two types of tokens used within Kintana: standard tokens and custom tokens. Standard tokens are provided with the product. Custom tokens are generated to represent your specific entity configurations (Object Type fields, Request Type fields, Workflow parameters, etc.). Each field of the following Kintana entities can be referenced as a token:

- Object Types
- Request Types
- Report Types
- User Data

- Workflow Parameters

Tokens can be used in many Kintana entity windows:

- Object Type commands
- Request Type commands
- Validation commands and SQL statements
- Report Type commands
- Executions and notifications for a Workflow
- Workflow Step commands
- Notifications in a Report Submission
- Special Command commands
- Notifications for Tasks
- Workflow Step security

Security Groups

Security Groups are constructed to provide a set of users with specific access to screens and functions within Kintana. Each Security Group is configured with a set of Access Grants that enable specific access. Users are then associated with one or more Security Groups.

A user's Security Group memberships determine which windows user can view or edit, which Workflows a user can use, and which Workflow Steps a user has authority to act on. Each Kintana user can be a member of multiple Security Groups. The collection of Security Groups to which a user belongs defines that user's role and access within Kintana.



Since users can be members of as many Security Groups as necessary, it is recommended that specific Security Groups are generated, each with a smaller range of responsibilities. Users can then be added to many different Security Groups to grant them their full range of access.

Security Groups control product access on the following levels:

- **Screen Security:**
Each Security Group contains a list of Access Grants that determine a user's screen security. Access Grants are used to grant access to edit, view, manage or submit a specific Security Group. By controlling the set of Access Grants for each user, specific functional roles for the user community can be defined.
- **Workflow Step Security:**
Each Workflow Step can be linked to a unique set of Security Groups. By adding or removing specific Security Groups from a Workflow Step in the Workflow window, you can control which Kintana users can act on that step. This security level provides an extremely detailed level of control over each Kintana user's actions.
- **Workflow Security:**
(This security level applies to Kintana Deliver only.) Security Groups can also control which Workflows users can select to deploy their objects. When Kintana users generate a new Package, they must choose the Workflow that the requested changes will follow. The list of Workflows from which the user can choose is determined by that user's Security Group membership.
- **Application Code Security:**
(This security level applies to Kintana Deliver only.) For complex Environments, information is often segmented in subsections called Environment Applications. Application Code security can be defined to further restrict a user's ability to cross functional boundaries and apply unwanted changes to applications that are managed by other divisions.

See "[Kintana Security Model](#)" for details on configuring security and user access around your deployment system.

Participants

Users who are involved in moving a Package through a Workflow are considered to be **Participants** in that Package. A Participant can be the:

- ASSIGNED TO user
- A member of the ASSIGNED GROUP
- The creator of the Package

- A member of a Security Group associated with any of the Workflow Steps contained in the Workflow.

You can configure Kintana so that a Package is not visible to users who are not Participants. This means users will only see Packages relevant to their business role in their organization. Additionally, users running Reports will only see information for Packages for which they are considered to be Participants.

Integration with Kintana Products and Solutions

This document focuses on configuring the deployment functions within Kintana Deliver. With additional Kintana products and licenses, you can address the full set of challenges across your IT department. All Kintana solutions and products can be seamlessly integrated to provide a complete solution to your IT management needs.

Other Kintana products include:

- *Kintana Solutions*
- *Kintana Dashboard*
- *Kintana Drive*
- *Kintana Create*
- *Kintana Accelerators*

Kintana Solutions

Kintana's Enterprise Application for IT includes a set of proven solutions to support key IT processes and functions. Each Solution introduces specialized business content developed to address specific business needs. Based on proven business practices, these solutions can be implemented modularly. Kintana supports the following solutions:

- Demand Management: ensures that all demands, regardless of type or source, are captured, evaluated, prioritized, and resolved efficiently.
- Portfolio Management: ensures alignment of strategic IT initiatives with business strategy.

- Program Management Office: ensures IT projects are delivered with very high quality and functionality, on time, within budget.
- Enterprise Change Management: ensures that IT delivers software for business use efficiently, with the highest quality and functionality, on time, and at low risk to production systems.

Kintana Solutions are licensed separately. For more information on implementing Kintana Solutions at your site, refer to Kintana's web site (<http://www.kintana.com>).

Kintana Dashboard

Intended for large and complex environments, Kintana Dashboard provides 360° visibility and control over technology-based initiatives and IT operational tasks. Configurable, role-based visual displays called “portlets” provide relevant summary information and highlight exception conditions in your Kintana-managed initiatives. Users can then drill down to any desired level of detail.

For example, a CIO may want to see the status of the major initiatives undertaken by the IT department. Instead of relying on weekly reports patched together from different sources and often compiled from out-of-date or incomplete information, he can go directly to Kintana Dashboard. The Dashboard displays the true status of the initiatives -- based on current data captured automatically as part of actually performing the work. Kintana Dashboard clearly identifies any initiative that is behind schedule, or in any other exception state, and displays the causes for the delay.

Kintana Dashboard is beneficial to all participants throughout the Technology Chain. For example, developers can use Kintana Dashboard to view all of their own action items, and end-users can consult their own Dashboards to see the status of all the Requests they have submitted.

Kintana Drive

Kintana Drive adds a critical dimension, automated execution, to complex project management in large IT organizations. Unlike static project management tools that simply schedule the tasks, dates, and resources, Drive's automation proactively pushes project tasks to the assigned resource, links with Kintana Create and Kintana Deliver to automatically perform issue resolution and deployment tasks, and automatically updates and reports project

status as task are completed. Project managers guide projects from concept to completion from Kintana Drive's centralized environment.

Packages (used in your deployment process) can be added to the Kintana Drive project plan. Dependencies can be set between Packages and Tasks on the project. This ensures that the technical aspects of the deployment process is respected by other resources on the project plan.

Kintana Create

Kintana Create accelerates the request resolution process from inception through implementation. Businesses use Kintana Create to model and enforce their best practice request management processes. As each new request is entered, email and pager notifications speed the request through the process, freeing users to perform required triage, approval and other resolution tasks. Higher priority requests receive appropriate treatment automatically. Resolution performance monitoring by activity, request, or even across the organization ensures SLAs are met.

Requests (in Kintana Create) can be functionally linked to Packages (in Kintana Deliver). This enables you to utilize the request resolution features of Kintana within your deployment system.

Kintana Accelerators

Kintana Accelerators simplify the complex activities required to maintain large enterprise applications like Oracle, PeopleSoft, SAP, and Siebel and Web applications built using Java, Oracle and others. These applications are constantly changing as new modules are added, customizations developed, configurations modified, patches applied, etc. The changes must be done precisely across the Development, Test, Stage and Production system landscape, usually by highly paid and hard-to-find specialists. Kintana Accelerators automate these precise tasks using best practice processes designed specifically for each application.

Reports

Kintana features two types of reports: standard reports and Decision Support System (DSS) reports. Kintana's standard reports output text that provides information on your specific entities or configurations. Kintana's DSS reports feature a graphical data display which helps evaluate key system and process performance.

See "[Kintana Reports](#)" for a complete list of the reports used in commonly used in deployment systems.

Chapter 3

Developing Your Kintana Configurations (Using Migrators)

Kintana controls the deployment of objects to mission critical applications. Before rolling-out new or modified functionality in Kintana, you should thoroughly test the changes in a Development or Testing instance. For example, before rolling out a new WEB UPDATE process to manage deployments to your company's web site, you should test the Kintana Workflow and Object Types used to perform the deployments.

Kintana provides functionality to help with this process: the Kintana Migrators. The Kintana Migrators are used to capture and move Kintana configuration data (for example, Workflow or Object Type definitions). This allows you to share configuration data between multiple Kintana instances. You can configure and test your Kintana configurations in a TEST instance, and then migrate your configurations to the PRODUCTION instance.

This chapter provides an overview of how to use multiple instances of Kintana to configure and deploy your Kintana configurations. It explains the concepts and basic architecture of this model, but does not provide implementation details. See the following documents for additional details:

- *"Kintana System Administration Guide"* for instructions on setting up multiple Kintana instances.
- *"Kintana Migrators"* for detailed instructions on using Migrators to move Kintana configuration data.
- *"Kintana Installation Guide"* for instructions on installing new Kintana instances.



Note

This chapter represents a change management implementation recommendation. Using the concepts and procedures listed in this chapter can reduce the risk of down time when rolling-out your Kintana processes.

See "[Kintana Migrators](#)" for detailed instructions on using Kintana Migrators.

This chapter discusses the following topics:

- [Using Multiple Kintana Instances - Overview](#)
- [Migrating your Kintana Configurations](#)
- [Archiving your Kintana Configurations](#)

Using Multiple Kintana Instances - Overview

Kintana recommends that you use multiple instances when configuring the entities and processes in the Kintana product suite. In the following sections, we will discuss the simplest multi-instance configuration, consisting of two instances: DEV (development) and PROD (production) located on different machines. You can extend the basic migration principles to support the number of Kintana instances used at your site.

There are two implementation scenarios for employing multiple Kintana instances. The process for implementing multiple Kintana instances differs depending on the following scenarios:

- **Single PRODUCTION instance is currently in use.**
Requires you to clone the PRODUCTION instance (file system and database) to create the DEV instance.
- **New Kintana implementation.**
Requires that you run the Kintana install multiple times.

Single PRODUCTION instance is currently in use.

For this scenario, you need to clone the PRODUCTION instance. Each Kintana instance consists of a file system and an Oracle database. These can exist on Unix or Windows machines. Contact your Kintana System Administrator for details about your site's configuration.

To move from a single live Kintana instance to multiple instances:

1. Clone the PROD instance. This includes the file system, database, and license information. Details for this procedure are included in the Kintana System Administration Guide. You should work with your Kintana System Administrator to implement this configuration.
2. Configure any changes to Kintana in the DEV instance. This includes creating or modifying Workflows, Object Types, Validations, Security Groups, Environments, etc.
3. Configure a Package Workflow to migrate the Kintana configuration data from DEV to PROD. Kintana recommends that this process is configured in the PROD instance.
4. Migrate data from the DEV instance into the PROD instance. Again, this activity is performed from the PROD instance. Therefore, it may help you to think of migrating the data as an “import” process.

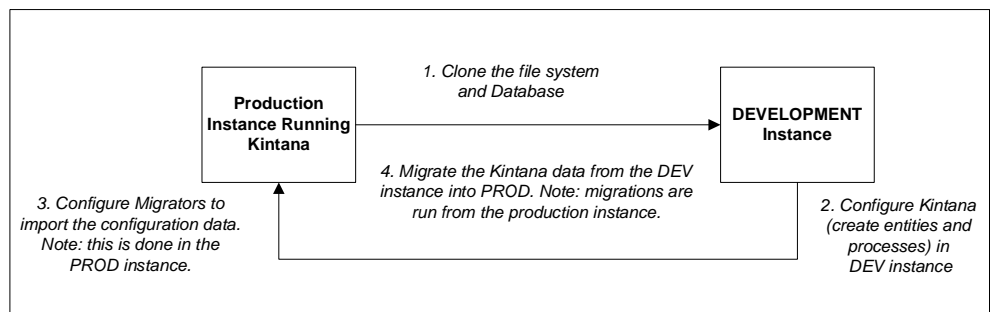


Figure 3-1 Cloning instance and configuring Kintana Migrators

New Kintana Implementation

When first implementing Kintana at your site, you can choose to immediately set up multiple Kintana instances. One can be configured as the DEV instance,

and the other can be configured as the PROD instance. By creating two blank instances up front, you avoid the need to clone existing data from one instance into another. When this scenario is exercised, you can follow the instructions included in the "[Kintana Migrators](#)" document.

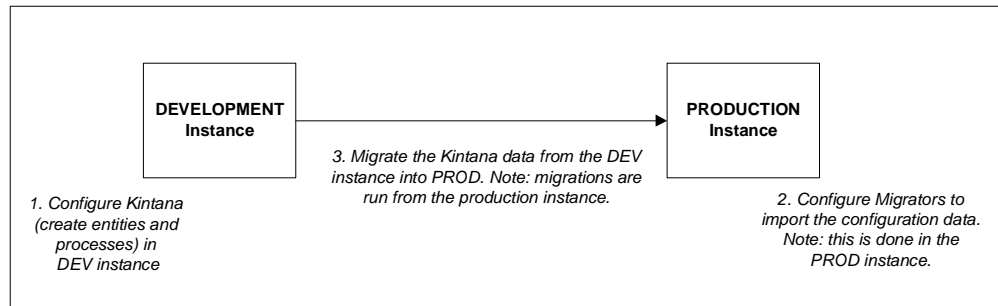


Figure 3-2 Migrating Kintana data between DEV and PROD

Migrating your Kintana Configurations

This section provides an overview of the requirements and processes for using Migrators. These are provided to help you communicate with your Kintana Administrator (who maintains the Kintana instances and license information) and your Kintana System Administrator (who maintains the Kintana server) when developing your deployment and distribution processes in Kintana.

The following topics are discussed:

- [How Kintana Migrators Work](#)
- [Using the Kintana Migrators - Overview](#)
- [Instance Requirements for Using Kintana Migrators](#)

How Kintana Migrators Work

Kintana Migrators are provided as Kintana Deliver Object Types. These Migrator Object Types are run through a Kintana Workflow. Each supported entity type has its own Object Type. For example, to migrate a Workflow from one instance to another, use the KINTANA WORKFLOW MIGRATOR Object Type.

Kintana Packages are used to process and audit the migration of configuration changes. When the Package (containing a Migrator object) enters the appropriate execution step in a Workflow, the Migrator's commands are executed. The commands extract the data that defines the entity into text (XML) files. These text files are then imported into the target instance.

Figure 3-3 represents this process.

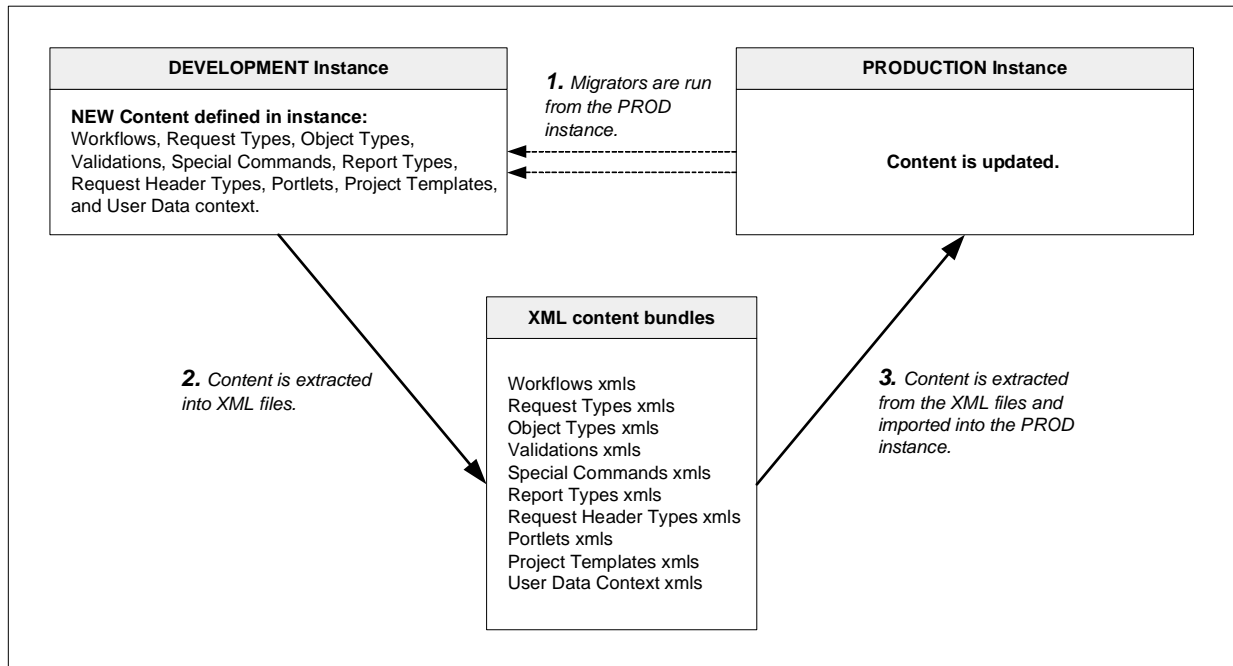


Figure 3-3 Migrator content extraction and import overview

Using the Kintana Migrators - Overview

To use the Kintana Migrators to migrate Kintana configuration information between instances:

1. Configure the appropriate Environments (DEV and PROD) to represent Kintana instances involved in the migration.
2. Define a Kintana Package Workflow to model the migration process you want.
3. Build a Package using this Workflow, using the appropriate Kintana Migrator Object Types to create the Package Lines.

4. Submit the Package and migrate the Package Lines. Use the execution log generated to evaluate the results of the migration.

Detailed instructions for using the Kintana Migrators are included in the "[Kintana Migrators](#)" document.

Instance Requirements for Using Kintana Migrators

To use the Migrators:

- You must have two working Kintana instances (DEV and PROD)
- The instances must be accessible over a network

Kintana Requirements for PROD Instance

The following items must be configured in the PROD instance. This is the destination instance that will receive the configuration data from the DEV instance. See "[Kintana Migrators](#)" for details on each of the below points.

Requirements for a successful migration:

- Environments (PROD and DEV) defined in the Environment screen in the Kintana Workbench.
- You must have at least one Kintana Workflow (SCOPE = **DELIVER**) to run the migration. This Workflow must contain at least one execution step with the source and destination Environments configured to DEV and PROD, respectively.
- Migrator Object Types must be enabled. There is a different Object Type for each of the following entities that can be migrated: Validation, User Data, Special Command, Workflow, Report Type, Object Type, Request Type, Request Header Type, Project Template, Portlet, User Data Contexts.
- The user creating, submitting, and processing the migrations must have a Deliver power license and proper screen access. See "[Kintana Security Model](#)" for details.

Archiving your Kintana Configurations

Kintana Migrators also let you document your processes by exporting configuration information to text files. These text files conform to the XML (eXtended Markup Language) specification and are suitable for storage in many archiving systems including source control systems. Source control check-in can be integrated into the Kintana Migrator Object Types, allowing organizations to maintain a detailed record of the specific changes made to their production Kintana configurations.

To archive your Kintana configurations:

1. Perform an Extract only using the appropriate Kintana migrator. The content is extracted into .xml files and grouped (by entity) into .zip files.
2. Check the extracts (.zip files) into your source control. Each extract contains a file (Source_Descriptor.xml) that describes the Kintana version and date of extraction.
3. You can then import these files back into a Kintana instance (of the same Kintana version) at any point in the future.

Chapter 4

Configuring your Deployment System - Process Overview

This chapter provides an overview of the process used to configure a deployment system in Kintana. It summarizes each phase of configuration. Additional details for configuration, are included in the referenced chapters.

Note

Kintana recommends that you develop your configurations in a DEV instance. See [“Developing Your Kintana Configurations \(Using Migrators\)”](#) on page 37 for an overview of using multiple Kintana instances for developing your processes in Kintana.

To configure a deployment system or process in Kintana, you will follow the process described below.

1. *Gathering Process Requirements and Specifications*

Before you begin configuring the Kintana product to manage your deployment processes, you need to collect specific related information. This includes gathering information on your business process, technical process, the types of objects that will be deployed, source and destination environments, participants who will create and process Packages, and the communication devices surrounding the process.

2. *Mapping your Process into a Kintana Workflow*

Using the information gathered in the [Gathering Process Requirements and Specifications](#) chapter, you will build your Workflow in Kintana. This includes setting up required workflow step sources, creating Validations to be used by the transitions, and adding steps and transitions to your Workflow.

3. *Constructing the Object Type*

Using the information gathered in the *Gathering Process Requirements and Specifications* chapter, you will build your Object Type(s) in Kintana. This includes creating and configuring Object Type fields and adding commands to your Object Type.

4. *Defining your Environments*

You must define all of the Environments involved in your deployment and distribution process. This includes setting up Environments and Environment Groups and then adding them to your Workflow definition.

5. *Integrating Participants into Your Deployment System*

After the Workflow and Object Types are constructed, you need to construct security around the process. Ultimately, you need to specify who can do what within your process. This includes specifying: who can create and process Packages, who can act on a particular Workflow step, and who can alter the process (Workflow, Object Types, Environments, etc.).

6. *Setting Up Communication Paths*

Kintana also includes a number of features that enable high visibility into Packages in your deployment process. You can create notifications for Workflow steps, configure portlets to provide additional real-time visibility, and use Kintana's reports.

7. *Rolling Out Your Deployment Process*

Kintana recommends that you follow a formal change management process when configuring Kintana. This includes testing your configurations, migrating them into your production instance, enabling the processes, and training your user base.

Example: Configuring a Deployment System

This section provides a business case example for configuring a deployment system in Kintana. The following example is used throughout this document to discuss Kintana configuration techniques.

Example: Deployment System for Financial Application system

The IT group at ACME Company receives hundred of requests every year for new and enhanced functionality for their FINANCIAL APPLICATION business system. The IT group is also responsible for applying regularly published application patches to this system. This system consists of over ten modules (billing, accounts payable, accounts receivable, fixed asset management, inventory, reporting, payroll, cash management, etc.).

The deployment of changes to the different modules can require unique processing items (destinations, environment management, post deployment processing steps, etc.). The ACME IT group needs to create a process that can address the complications related to deploying changes to this system.

Additionally, they need to address the following requirements:

- The TESTING environment must use the code and data that is housed in the version control system.
- Only certain users can approve and migrate changes.
- The PRODUCTION update must occur between 1:00 and 2:00am on Friday. Any additional system downtime could result in financial loss.

Chapter 5

Gathering Process Requirements and Specifications

Before you begin configuring the Kintana product to manage your deployment and distribution processes, you need to collect certain related business and technical information. This includes information on the:

- **Business process:**
What are the steps in the process and which steps need to be reviewed and approved?
- **Technical process:**
Which steps require objects to be deployed and scripts to be run?
- **Types of objects that will be deployed:**
Will the same process be used to deploy different type of objects (files, data, scripts, etc.)?
- **Source and destination environments**
- **Participants who will create and process Packages:**
What level of security do you want to place on this system?
- **Communication devices surrounding the process:**
Do you want to communicate using notifications, the Kintana Dashboard, or reports.

You need to consider all of the above topics when configuring a new process in Kintana. This chapter discusses the information that you need to collect and provides examples and worksheets to help you manage this information. This chapter includes the following sections:

- *Define the Deployment Process*
- *Determine Information to Describe Objects*
- *Determine Commands Needed for Objects*

- *Gather Information on Environments*
- *Identify Participants and Security*
- *Establish Communication Points and Visibility*

Define the Deployment Process

The first step to configuring your deployment and distribution process is to define the process – the actual steps required to deploy an object. This includes process information such as when to obtain reviews and approvals on the object to be deployed, when to deploy objects, and what’s the path (transitions) between steps in the process.

The following sections discuss the specific information that you need to gather to help you later when configuring your Kintana Workflow:

- *Process (Workflow) Considerations*
- *Define the Business Flow*
- *Defining the Technical Flow*
- *Gather Information on Each Step in the Process*
- *Consider Using Subworkflows*
- *Consider Using Kintana Release Management*

Process (Workflow) Considerations

The Kintana Workflow includes a number of features that you should consider when defining your process. The following section describes a few of the notable features. For a more comprehensive discussion of Workflow features and configuration techniques, refer to *“Mapping your Process into a Kintana Workflow”* on page 79.

Decision versus Execution steps

Decision Steps

Decisions are workflow steps that require an external process to decide their outcome. Typically, this is an individual logged into the system (such as a QA Manager approving a bug fix). Decisions are used for a wide variety of purposes within a workflow. They may be used to gather approvals before code is migrated, or they may be used to request additional information on a Request (within Kintana Create).

Decision steps are represented on the Workflow by the following default icon:



Execution Steps

Executions are steps that perform actual work. This work can consist of activities like object migrations, the creation of a new Package or the completion of a Request. Executions can be immediate, manual or scheduled. Technical processing of the object or Environment occurs at an Execution step.

Execution steps are represented on the Workflow by the following default icon:



Immediate Versus Manual Executions

Timing is often very important in your deployment process. Kintana provides functionality that allows you to control exactly when certain execution steps are run. You can configure your process with an **Immediate** execution step that will execute the step at the instant that the step becomes eligible (Package enters the step). You can also configure your process so that you need to manually execute a step.

Timeouts

Another element of process timing lies within the timeout. If a process step is in a specific state for a predetermined period of time, the process can “timeout.” The process can then process based on the timeout event. This allows you to configure your process to avoid potential bottlenecks.

Define the Business Flow

Map your business process. This consists of identifying all of the steps (decisions, conditions, points of execution) and transitions needed to deploy your changes. It is helpful to graphically map these processes. In this phase, you should:

- Identify all decision points in the process
- Determine a flow between steps (transitions). You should consider all possible exit values from each step (approved, not approved, rework, error, etc.)
- Identify process closure points (success or failure)

The following example provides an illustration of the design issues that you should consider.

Example: Defining the Business Flow

ACME Company needs to configure a deployment process for changes to their Financial Applications system. This system consists of over ten modules (billing, accounts payable, accounts receivable, fixed asset management, inventory, reporting, payroll, cash management, etc.). Deployment to the different modules can require unique processing items (destinations, environment management, post deployment processing steps, etc.). ACME’s IT group needs to create a process that can address the complications related to deploying changes to this system.

Additionally, they need to address the following requirements:

- The TESTING environment must use the code and data that is housed in the version control system.
- Only certain users can approve and migrate changes.

- The PRODUCTION update must occur between 1:00 and 2:00am on Friday. Any additional system downtime could result in financial loss.

Example: ACME defines a high-level process flow

ACME first creates a high-level business process. The process begins after the software engineers and IT staff develop changes to the Financial Applications module, and check their code into their version control system. ACME's deployment process begins with the MIGRATE DEV TO TEST step.

1. **Migrate DEV to TEST:**
Migrate the changes from the Development area (DEV) to the Testing area (TEST). This includes checking the code out from the version control system into the DEV area, transferring the files, and then compiling the code on the TEST instance.
2. **Validate changes in TEST:**
Validate the changes in TEST using standard Quality Assurance processes. Any required rework is routed back to the appropriate engineering staff.
3. **Migrate TEST to PROD:**
Migrate the changes from TEST to the Production area (PROD). This, again, involves checking out files from the version control system, transferring the files and compiling the code.

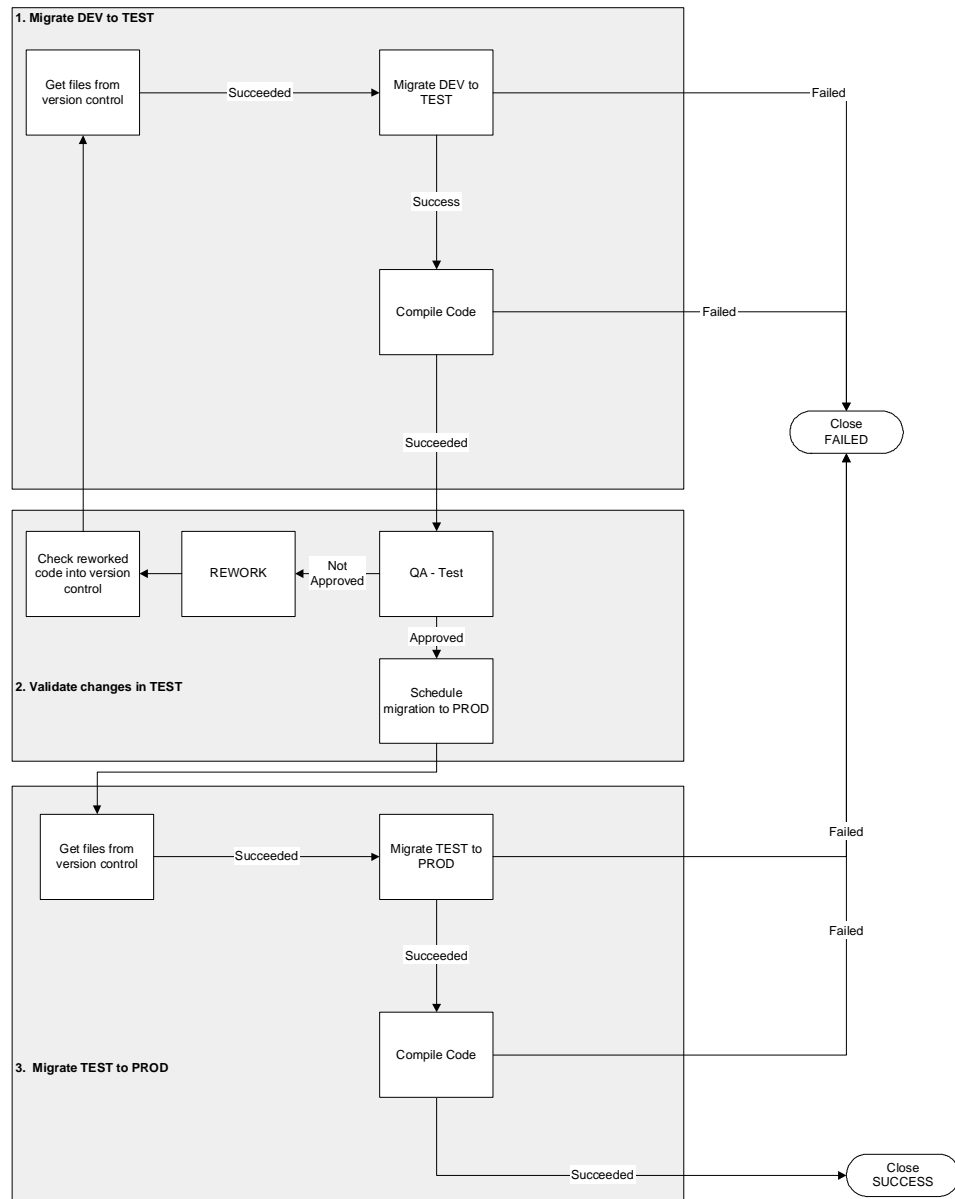


Figure 5-1 Deployment Process - High Level

Defining the Technical Flow

Once the high-level business process is defined, you can overlay a technical process. Defining the technical process includes identifying:

- Types of objects that you will be deploying

- Dependencies between objects, environments, and workflow steps
- Any required system level commands
- Where to use condition steps (AND, SYNC, OR, FIRST LINE, LAST LINE)

Example: Defining the Technical Flow

After investigating some more technical requirements of their system, ACME identified the following information:

Types of objects to be deployed:

- HTML Files
- Java Files
- Database changes: changes to the schema as well as additional system data for existing tables

Object-related dependencies:

- The server and database are located on different machines. Therefore, database-related objects need to be deployed to a different machine than the other objects (files).

Additional process requirements:

- The server must be stopped before the migration.
- The code can not be compiled until all objects have been migrated.
- Following a FAILED execution step, you should be able to reset the execution.

Example: Detailed Process

ACME updated their process to address the additional information, resulting in the following process.

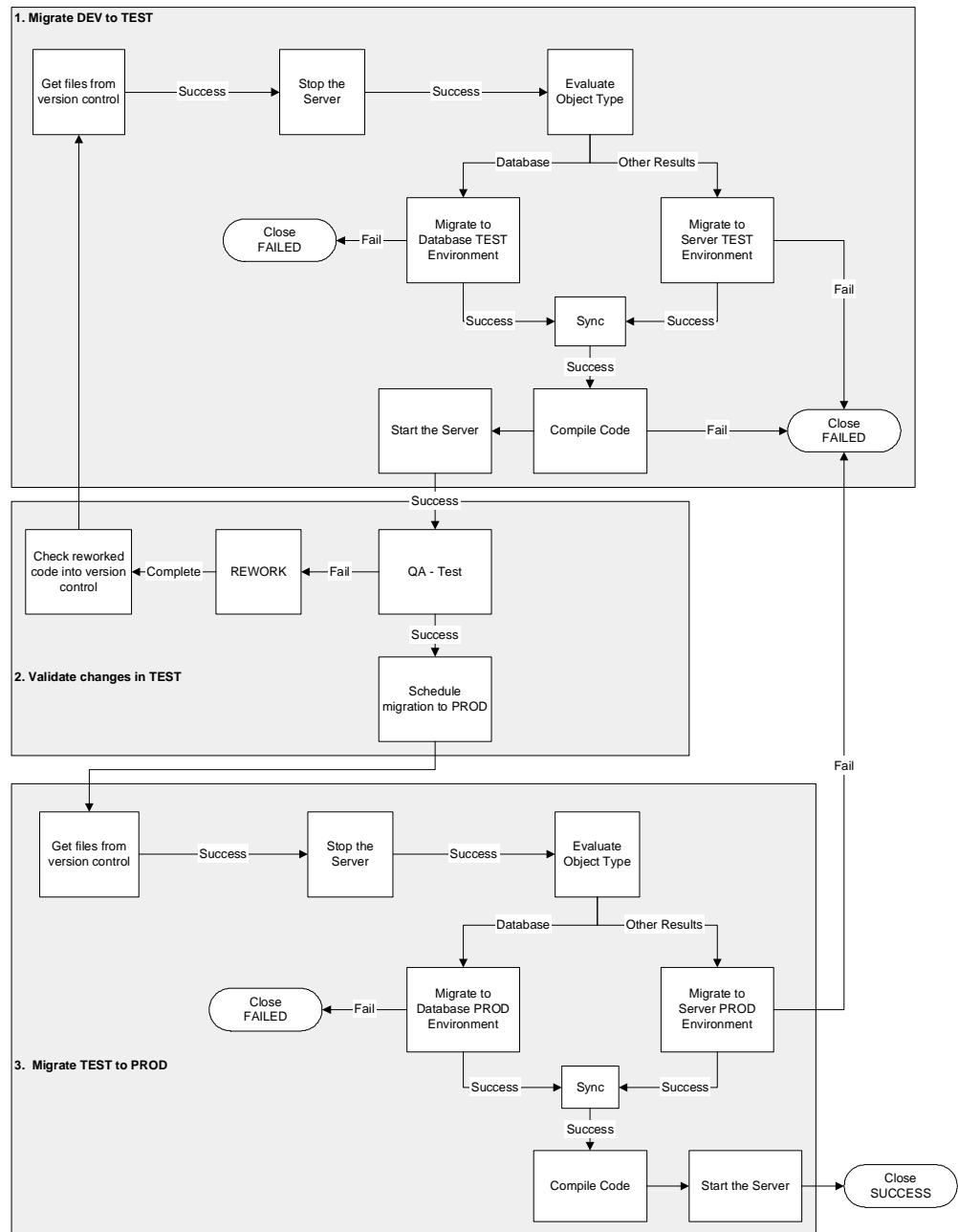


Figure 5-2 Deployment Process - Detailed Level

To address the technical requirements of their deployment process, ACME introduced the following changes to their workflow:

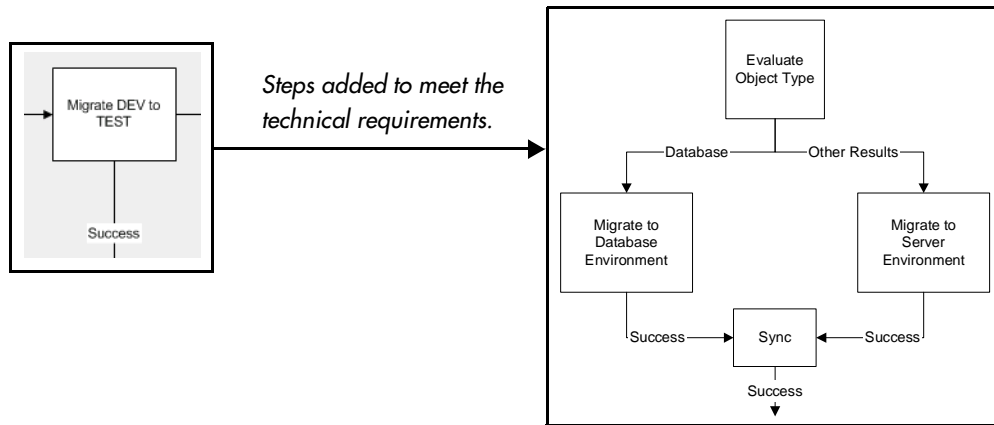
Process change 1:

Requirement:

The server and database are located on different machines. Therefore, database-related objects need to be deployed to a different machine than the other objects (files).

Result:

To address this requirement, ACME added the EVALUATE OBJECT TYPE step. If the object being routed through the deployment process is a database-related object, then it is deployed to the database environment. All other objects are routed to the server environment.



Process change 2:

Requirement:

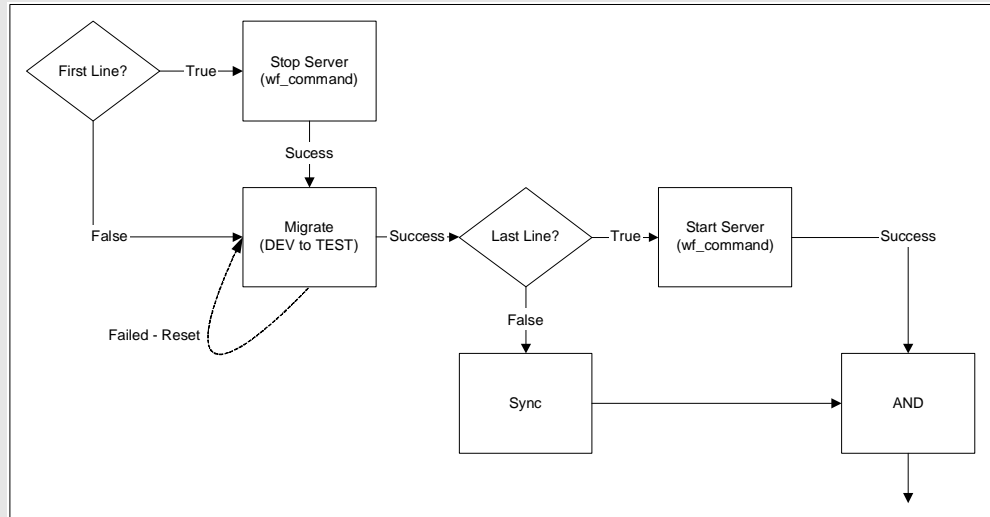
The server must be stopped before the migration

Result:

To address this requirement, ACME added a STOP THE SERVER step before the files are transferred. This step appears during both DEV TO TEST and TEST TO PROD migrations.



You can also use the **FIRST LINE** and **LAST LINE** functionality to stop and start the server. The first package line (object) that is processed through the workflow stops the server, and the last package line starts it. This is shown in the following diagram:



The Sync step is used to ensure that all package lines move in unison to the AND step. When the last line starts the server, it proceeds to the AND step and the Package can continue.

See [“Advanced Workflow Topics”](#) on page 257 for additional details on this configuration.

Process change 3:

Requirement:

The code can not be compiled until all objects have been migrated.

Result:

To address this requirement, ACME introduced a SYNC step following the migration steps. This ensures that all objects (Package Lines) reach a specific point before the Package can continue along its process. It also ensures that the “branched” objects (database versus others) are reunited in the process.

Gather Information on Each Step in the Process

After gathering the business and technical process steps of your deployment process into a single workflow, you need to gather detailed information on each step and transition in your process. This section discusses the information that you should collect. “*Configuration Worksheets*” on page 391 includes a worksheet that will help you collect the required information.

For each step in your process, collect the following information:

- Step name
- Description: Describe the goal of the step. This is especially helpful for execution steps.
- Step Type: decision, execution, condition, or subworkflow.
 - o Decision step specific information: number of approvals required, timeouts, etc.
 - o Execution step specific information:
 - The desired results of the execution. This will help you to choose the execution type and build any required commands.
 - Execution timing. Determine whether you need the execution to occur immediately or to be processed manually.
 - o Subworkflow step specific information
- Transition values and validation:

Transition values are the possible results for the step. Depending on the result, the process will proceed in different directions. You can use one of Kintana’s system Validations or create your own.

Note

For each step, you will also need to collect information on Environments, Participants, and Notifications. This is discussed in the following sections:

- “*Gather Information on Environments*” on page 69
- “*Identify Participants and Security*” on page 71
- “*Establish Communication Points and Visibility*” on page 76

The “*Configuration Worksheets*” on page 391 provide tools for capturing all of the Workflow step information in one place.

Example: Gathering Workflow Step Information

ACME begins capturing the step information for the Migrate DEV to TEST portion of their process.

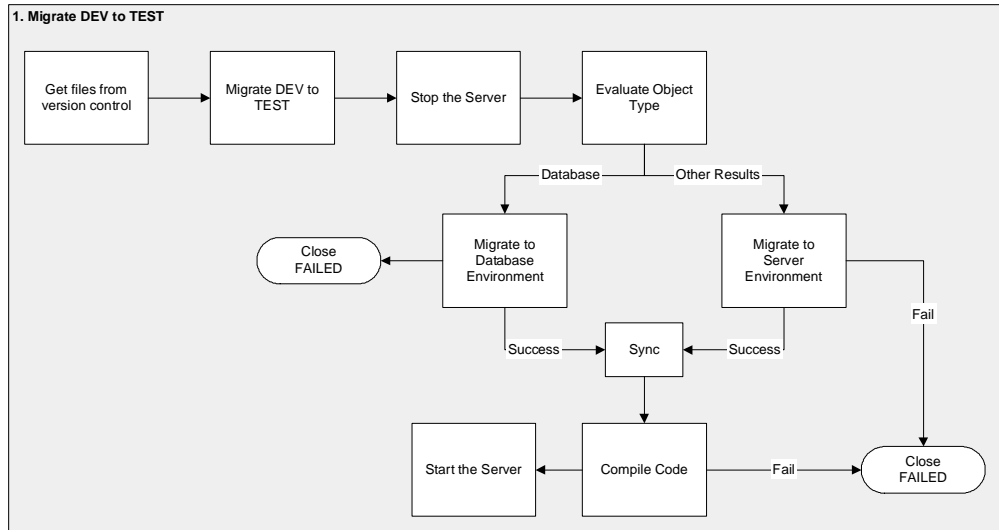


Table 5-1. ACME process workflow step information

Step Name	Type	Transition Values	Validation	Description
Get files from version control	Execution	Succeeded Failed	WF - Standard Execution Results	Connect to the version control system and check out files into the DEV instance.
Migrate DEV to TEST	Decision	Approved Not Approved	WF - Approval Step	Decide whether to begin the migration process.
Stop the Server	Execution	Succeeded Failed	WF - Standard Execution Results	Connect to the server and stop the processes running on it.
Evaluate Object Type	Execution	Database File SQL Script	Custom Validation defined at site.	Evaluate the object type for each Package Line. Resolve the Object Type Token.
Migrate to Database Environment	Execution	Succeeded Failed	WF - Standard Execution Results	Migrate the database changes to the TEST database environment. To do this, execute commands located in the Object Type.

Table 5-1. ACME process workflow step information

Step Name	Type	Transition Values	Validation	Description
Migrate to Server Environment	Execution	Succeeded Failed	WF - Standard Execution Results	Migrate the changes to the TEST server environment. To do this, execute commands located in the Object Type.
Sync	Condition	Success	WF - Standard Condition Results	Have all Package Lines enter this step before any continue to the next process step.
Compile Code	Execution	Succeeded Failed	WF - Standard Execution Results	Connect to the server and compile the code located on it.
Start the Server	Execution	Succeeded Failed	WF - Standard Execution Results	Connect to the server and start the processes on it.
Close FAILED	Execution	Succeeded	WF - Standard Execution Results	When a Package Line (object) enters this step, close the Package.



Using the “*Configuration Worksheets*” on page 391, you can also gather the information related to Workflow security, notifications, and Environments.

Consider Using Subworkflows

A Subworkflow is any Workflow that is referenced from within another Workflow. Subworkflows allow you to model complex business processes into logical, more manageable and reusable sub-processes.

Workflows can be used as Subworkflows within a parent Workflow. An entire Subworkflow is represented by a single icon in the parent Workflow window’s Layout tab. This simplifies the potentially complex graphical layout and enables the easy reuse of common Workflow configurations.



Subworkflows will appear to the user processing the Package as expandable / collapsible sections in the PACKAGE STATUS panel. See the “*Processing Packages*” user guide for examples.

Example: Using a Subworkflow

ACME decides to use a Subworkflow for the object migration portion of their process. This Subworkflow can be referenced in two parts of the process. *Figure 5-3* shows where ACME could implement a Subworkflow.

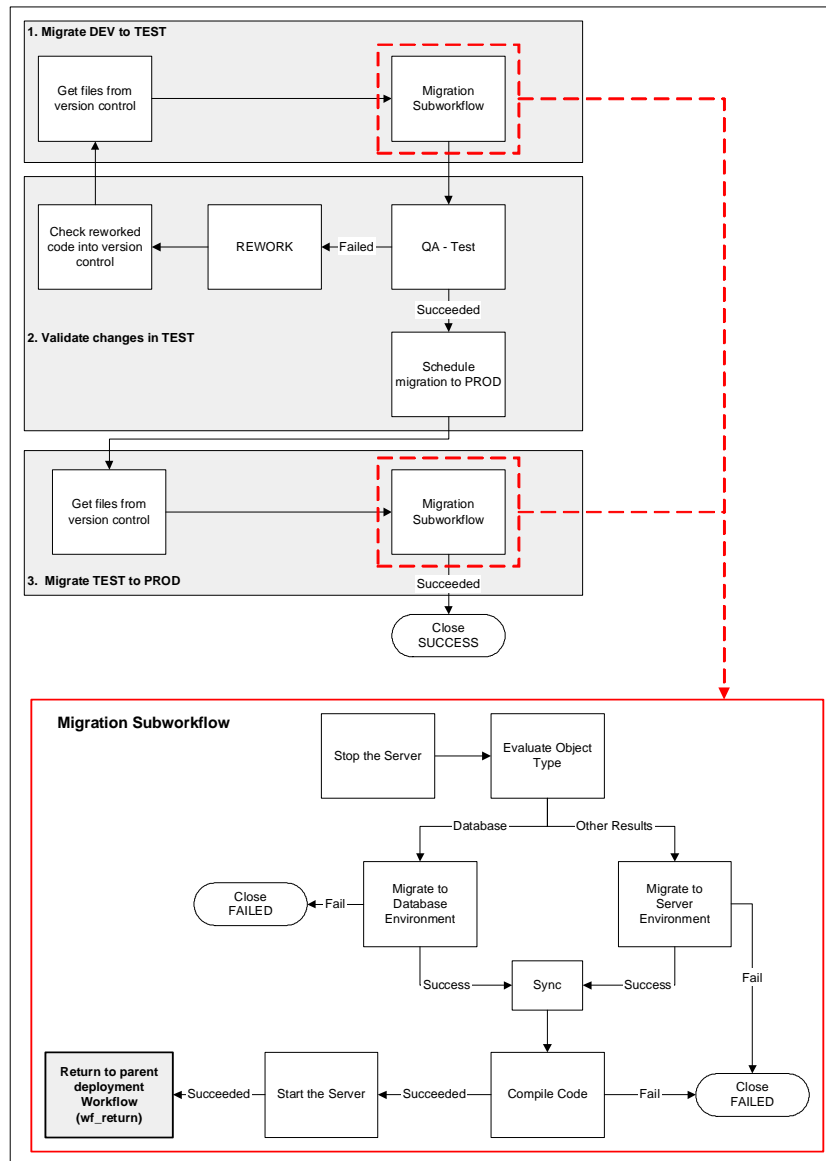


Figure 5-3 Using a Subworkflow in your deployment process (example)

Consider Using Kintana Release Management

Release Management introduces repeatable, reliable processes surrounding software and application releases. Kintana provides an interface for grouping

and processing the Packages and Requests associated with a specific Release. Groups of related Packages can then be activated from a single window.

Using Release Management, Kintana Release Managers can:

- Group related Packages and Requests in a single window
- Provide visibility into related Package statuses
- Set dependencies between Packages
- Define how a Release is distributed to different Environments

This consolidation of common Release Management activities provides a powerful tool for creating repeatable and reliable Releases.

You can configure your deployment process to feed Packages into a Release. A READY FOR RELEASE step can be included in the workflow. When a Package Line enters the READY FOR RELEASE step, the developer (or other Kintana user responsible for that Package) can select which Release they would like to add the Package to. The user selects the Release and adds the Package and its associated Package Lines to the Release. When all of the Package Lines are confirmed in the READY FOR RELEASE step, the Package is ready to be used in the Release.

See "[Configuring a Release Management System](#)" for more information on using Kintana's Release Management functionality.

Example: ACME Uses Release Management in their Deployment Process.

ACME decides to create a process that feeds the tested packages into a Release for final distribution. Their resulting process then substitutes the final migration to PROD with a READY FOR RELEASE step. The distribution workflow defined for the Release Management process then handles this final migration.

When the final distribution to PROD occurs, the Release Management process communicates with this deployment process and the package will close.

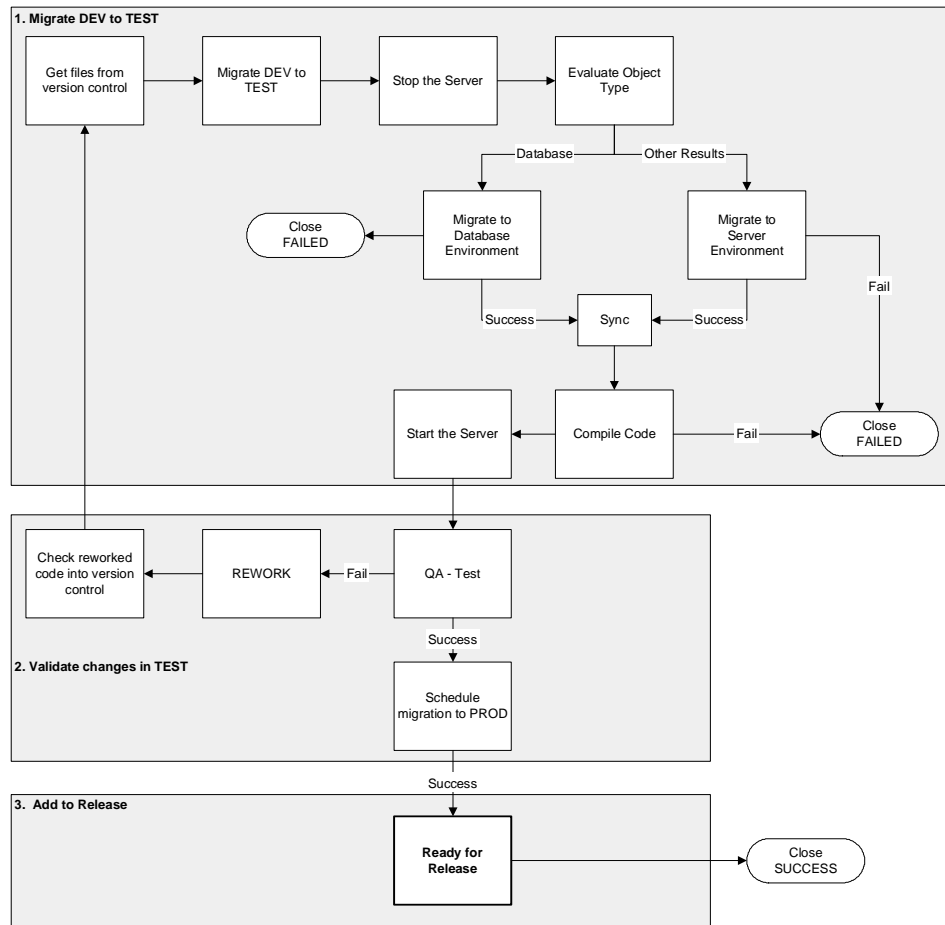


Figure 5-4 Using a Ready for Release step in the deployment process

Determine Information to Describe Objects

You deploy many types of objects through a Workflow: files, SQL scripts, data, etc. Each object requires different information to process it. These are defined in the Object Type fields. For example, to migrate a file, you need to know the File Name, its Location, and the File Type. Additional information such as whether or not you should compile the file after migration can also be specified on the field level (as well as in the commands).

Different Object Types can be processed through a single Workflow. Information contained on the Package Line (which is defined in the Object Type) works in conjunction with the workflow process to ensure that the object is correctly processed. For example, deploying an HTML file requires different processing than deploying a Java file. Therefore, it is likely that at least one

field on the Object Type will specify the type of object being processed. When the workflow deploys an HTML file, it will consider this field when routing or processing this object.

For each type of object that you will be deploying through your process, collect the following information:

- The name of the object
- Object category (optional -- used for reporting purposes)
- Parameters that describe the object: what it is, where it is, its name, what needs to be done to it, etc. This information will translate into Object Type fields. For each parameter (field), define the following:
 - o Field name
 - o Validation and component type (dictated by the validation)
 - o Field Behavior: whether it is displayed, required, any defaulting behavior, etc.
- Commands contained in the object. See *“Determine Commands Needed for Objects”* on page 68 for additional information. Object Type commands often reference information stored in the parameters. These commands are executed at specific points (executions steps) in the workflow.

Use the *“Configuration Worksheets”* on page 391 for assistance in collecting the correct data.



Note

Successfully defining object parameters and commands is essential to an effective deployment process. To provide additional guidance on creating Object Types for specific types of objects, we have included a number of examples in this document. See *“Constructing the Object Type”* on page 135 for some sample Object Types, fields and commands.

Example: ACME collects information on their objects

ACME needs to be able to deploy the following types of objects:

- HTML Files
- Java Files

- Database changes – changes to the schema as well as additional system data for existing tables

The following worksheet includes data for the FILE Object Type. To describe the File object, ACME decided they needed to define the following fields:

- File Location: whether the file located on the Environment's client or server.
- Sub path: the directory where the file is located.
- File Name: the name of the specific file.
- File Type: the type of file (Java or HTML)

ACME decided on the appropriate validations and field behavior and completed the worksheet shown in *Figure 5-5*.

Table 1. Java File - Object Type		
	Value	Description / Notes
Object Type Name	File	
Description		
Field 1		
Field Prompt	File Location	The name of the field.
Validation		
Existing Validation?	DLV - File Location	Specifies if the file is located on the Environment's client or server.
New Validation?	NA	NA
Validation type	Drop Down List	This field is a drop down list.
Validation definition	SQL	This validation is defined by SQL.
Field Behavior		
Attributes		
Display	Yes	This field is visible on the Package Line.
Editable	Yes	This field can be edited even after the Package is submitted.
Display Only	Never	This field can be edited. If set to "Always," you can never edit this field.
Required	Always	This field must contain a value.
Default Value	None	No defaults are set for this field.
Dependencies		
Clear When	None	No dependencies are set for this field.
Display Only When	None	No dependencies are set for this field.
Required When	None	No dependencies are set for this field.
Field 2		
Field Prompt	Sub Path	The name of the field. Use this field to select the directory containing the file to be migrated.
Validation		
Existing Validation?	Directory Chooser	Specifies if the file is located on the Environment's client or server.
New Validation?	NA	NA
Validation type	Directory Chooser	
Validation definition	Default behavior	
Field Behavior		
Attributes		
Display	Yes	This field is visible on the Package Line.
Editable	Yes	This field can be edited even after the Package is submitted.
Display Only	Never	This field can be edited. If set to "Always," you can never edit this field.
Required	Always	This field must contain a value.
Default Value	None	No defaults are set for this field.
Dependencies	None	No dependencies are set for this field.
Field 3		
Field Prompt	File Name	The name of the field. In this field, users can select a file to deploy.
Validation		
Existing Validation?	File Chooser - Full File Name	Validation that allows you to select a file on the client or server.
New Validation?	NA	NA
Validation type	File Chooser	
Validation definition	Default Behavior	
Field Behavior		
Attributes		
Display	Yes	This field is visible on the Package Line.
Editable	Yes	This field can be edited even after the Package is submitted.
Display Only	Never	This field can be edited. If set to "Always," you can never edit this field.
Required	Always	This field must contain a value.
Default Value	None	No defaults are set for this field.
Dependencies	None	No dependencies are set for this field.
Field 4		

Figure 5-5 ACME data collected to describe objects for deployment.

Determine Commands Needed for Objects

When defining your deployment process, you must also consider what commands need to be executed to achieve the desired results. Commands tell Kintana precisely which steps must be executed for each Workflow step to complete successfully. This can involve such things as migrating a file, executing a script, or compiling some code.

At early stages of your process development, it often helps to list the functional steps and desired results of the commands. It also helps to specify when in the deployment process these commands should be executed. You can then use this information to build your commands adhering to Kintana's command standards; or you can deliver these as design specifications for an engineer in your group.

Collect the following information for each Object Type that you design:

- The goal/purpose of the commands.
- Functional steps within the commands.
- When the commands should be run.

See *"Using Commands and Tokens"* for additional information on building commands in Kintana.



The above information can be collected using *"Configuration Worksheets"* on page 391.

Commands		
Goal of Command		
Command Steps		
Conditions (when to run)		

Example: High level Command design

To deploy the Java and HTML files, ACME must include commands in the FILE Object Type. As part of their design, they determine that the following command steps must be executed:

Goal of command:

To copy the file from the source environment to the destination environment.

Steps to achieve goal:

1. Check to see if the source file is located on the client or the server.
2. Connect the destination environment to the source (client or server).
3. Check to see if the directory exists in the destination. If it doesn't, create the directory.
4. Copy the file from the source to the destination.

When to run the commands:

These commands should be run during the MIGRATE TO SERVER ENVIRONMENT Workflow step.

Gather Information on Environments

Environments are specified in execution type Workflow steps that require Environment information to complete their executions. For your deployment process, you should collect the following environment requirements for each appropriate Workflow execution step:

- Execution Step Name
- Source Environment (or Environment Group)
- Destination Environment (or Environment Group)

This information can be collected using the Workflow step worksheet in *“Configuration Worksheets”* on page 391.

Each Environment must be carefully configured to ensure that passwords, communication protocols, and transfer protocols are configured properly. See *“Defining your Environments”* on page 163 for instructions on configuring your Environments. You can then use these newly configured Environments in your deployment process.



Note

If you have multiple applications on a single environment, you can use the App codes feature in the Environment definition. See *“Using App Codes with Your Environment”* on page 170 for additional details.

Example: ACME specifies the Environments

ACME determines that the following workflow steps require the Environment settings specified below:

Workflow Execution Step	Source Environment	Destination Environment
Get files from Version Control	VERSION CONTROL	DEV Server
Stop the Server	DEV Server	TEST Server
Migrate to Server Environment	DEV Server	TEST Server
Compile Code	DEV Server	TEST Server
Start the Server	DEV Server	TEST Server
Migrate to Database Environment	DEV Database	TEST Database
Stop the Server	TEST Server	PROD Server
Migrate to Server Environment	TEST Server	PROD Server
Compile Code	TEST Server	PROD Server
Start the Server	TEST Server	PROD Server
Migrate to Database Environment	TEST Database	PROD Database

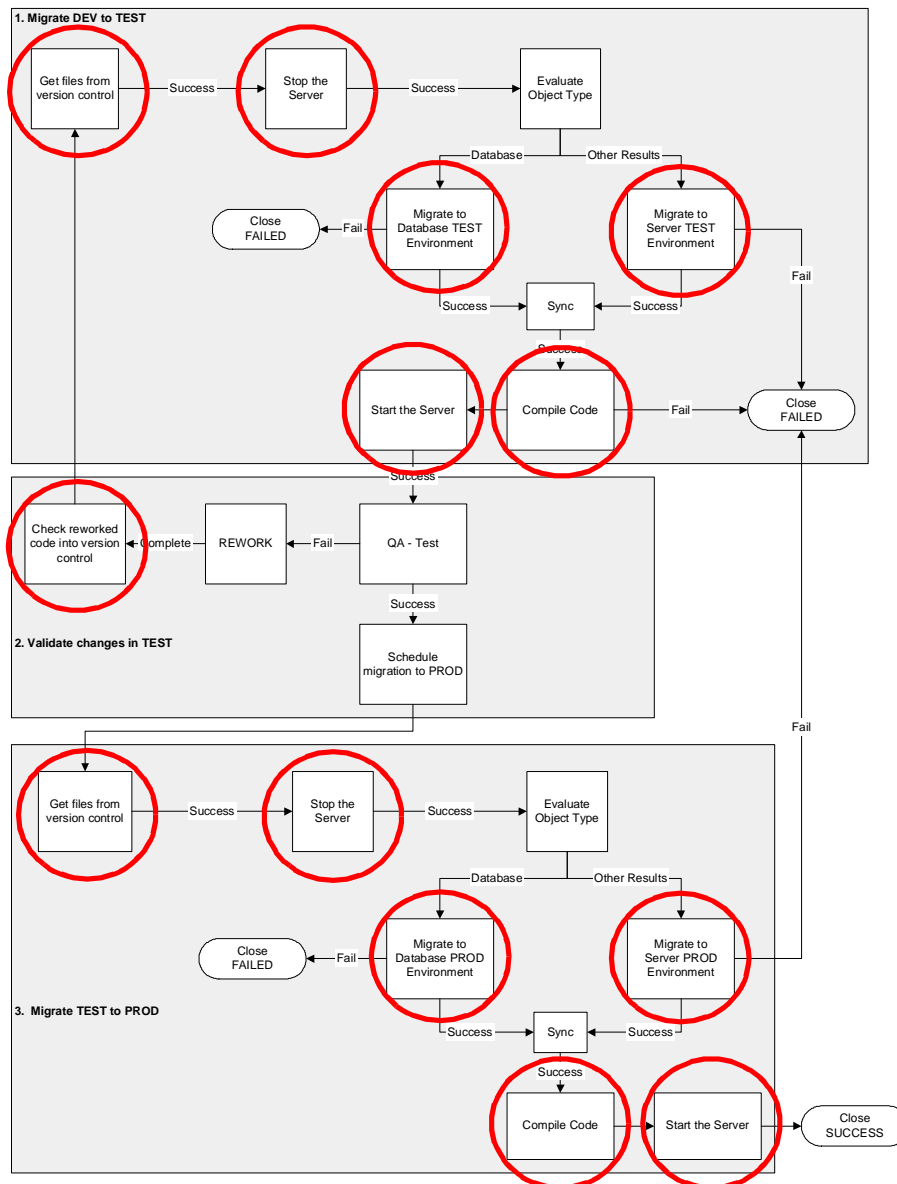


Figure 5-6 Workflow steps that require Environment specification (circled)

Identify Participants and Security

Kintana allows you to exercise a great deal of control over who can participate in your deployment process. You can restrict users' actions around:

- **Package creation:**

- o Who can create Packages.
- o Who can use a specific Workflow.
- o Who can use specific Object Types.
- **Package processing:**
 - o Who can approve / process each step in the Workflow. For this restriction, you can enable access by specifying specific users or security groups. You can also provide access dynamically by having a Kintana Token resolve to provide access. See [“Enabling Users to Act on a Specific Workflow Step”](#) on page 207 for more information.
 - o Whether you only want “Participants” to process the Packages. Participants are defined as the Assigned User, the creator of the Package, members of the Assigned Group, or any users who have access to the Workflow step(s).
- **Managing your deployment process:**
 - o Who can change the Workflow.
 - o Who can change each Object Type.
 - o Who can modify Security Groups.
 - o Who can modify Environments.



Kintana recommends using Security Groups or dynamic access (Tokens) whenever possible. You should avoid specifying a list of users to control an action; for example, specifying a list of users who can act on a Workflow step. If the list of users changes (due to a departmental reorganization), you would have to update that list in many places on the workflow. By using a Security Group instead of a list of users, you can update the Security Group once, and the changes are propagated throughout the Workflow steps.

For your deployment process, you should collect the above information using the [“Configuration Worksheets”](#) on page 391. This activity includes identifying users, grouping them into security groups, and restricting access to certain functions in Kintana.



This information is gathered in multiple worksheets. For example, Workflow security is captured in the Workflow step worksheet and Security Group membership is captured in the Security Groups worksheet.

See the "*Kintana Security Model*" for a comprehensive discussion of Kintana screen, entity and user security.

Example: ACME determines participants and security

ACME’s IT department has a single team responsible for deployments to the Financial Applications system. The team’s name is “Dev - Financial Apps.” The team consists of the following members:

- John Smith - manager of Dev - Financial Apps team
- Wendy Jones - functional designer and engineer
- Pat Lee - engineer
- Joe Franklin - QA manager
- Matt Davis - QA engineer and tester
- Raj Satish - Database expert and engineer

Within this group of users, there are some logical divisions of labor. Using this division, ACME constructs the following Security Groups.

Table 5-2. ACME’s Security Groups

Security Group	Members	Responsibilities
Financial Apps - Manage Deployment	John Smith Joe Franklin	Responsible for deployment process. These people have the ability to modify the deployment process (Workflow, Object Types, and Security Groups). They can also act on any step in the process.
Financial Apps - Database	Raj Satish	Responsible for deployments to the database environment. Also responsible for the correct setup of the database environment and its definition in Kintana.
Financial Apps - Engineer	Wendy Jones Pat Lee	Responsible for designing changes and deploying them to the server. Also responsible for the commands used in the Object Type to deploy objects to the server.
Financial Apps - QA	Joe Franklin Matt Davis	Responsible for testing the changes and reporting on the outcome.

Using these Security Groups and user definitions, ACME collects specific information related to their deployment process. This information will be considered later when defining your Security Groups and Workflows. The information is gathered in the following sections:

- Table 5-3, “ACME Package Creation Security,” on page 74
- Table 5-4, “ACME Package Processing Security - Deployment Workflow,” on page 75
- Table 5-4, “ACME Package Processing Security - Deployment Workflow,” on page 75

Package Creation Security:

Table 5-3. ACME Package Creation Security

Action	Users allowed to perform action	Controlled by: (Users, Security Group, Token)
Create a Package	Wendy Jones; Pat Lee; Raj Satish	Financial Apps - Engineer Financial Apps - Database Financial Apps - Manage Deployment
Use the Deployment Workflow	Wendy Jones; Pat Lee; Raj Satish	Financial Apps - Engineer Financial Apps - Manage Deployment
Use the File Object Type	Wendy Jones; Pat Lee; Raj Satish	Financial Apps - Engineer Financial Apps - Database Financial Apps - Manage Deployment
Use the Database Object Type	Raj Satish	Financial Apps - Database Financial Apps - Manage Deployment

Notice that the Financial Apps - Manage Deployments group was added to each action. This provides a single group with override privileges to keep the process moving.

Package Processing Security:

ACME decides not to use Kintana’s Participant Restriction functionality in their deployment process. See [“Restricting Package Processing to Participants”](#) on page 209 for additional details on this configuration option. The following table documents which users can act on a specific step in the Workflow. ACME also indicates how they would like to control which users can act on each step. They select to exclusively use Security Groups and Tokens. Notice that you can specify multiple criteria to enable access to a single step: for example, you could specify two security groups and a TOKEN

[PKG.CREATED_BY] to enable access. Users who meet any of the requirements (members of at least one Security Group or the contextual value of the Token) can act on the step.

Only a sub-set of the workflow steps are included in the below table. See [Figure 5-2 on page 56](#) to see the process referenced in this table.

Table 5-4. ACME Package Processing Security - Deployment Workflow

Workflow Step Name	Users allowed to act on	Controlled by: (Users, Security Group, Token)
Stop the Server	Wendy Jones; Pat Lee; Raj Satish	Financial Apps - Engineers; Financial Apps - Manage Deployment
Migrate to Database Environment	Raj Satish	Financial Apps - Database
Migrate to Server Environment	Wendy Jones; Pat Lee	TOKEN (PKG. CREATED_BY); Financial Apps - Manage Deployment
QA - Test	Joe Franklin; Matt Davis	Financial Apps - QA Financial Apps - Manage Deployment
Rework	Wendy Jones; Pat Lee; Raj Satish	TOKEN (PKG.ASSIGNED_TO_USERNAME);
Schedule migration to PROD	John Smith; Joe Franklin	Financial Apps - Manage Deployment

ACME must also specify who can modify the existing process. This level of security is configured using Kintana’s Ownership settings and Security Group access grants. See ["Kintana Security Model"](#) for more information on these topics.

Table 5-5. ACME - Security around managing the Process

Action	Users allowed to perform action	Controlled by: (Users, Security Group, Token)
Modify the Workflow	John Smith; Joe Franklin	Financial Apps - Manage Deployment
Modify the File Object Type	Wendy Jones; Pat Lee	Financial Apps - Engineering
Modify the Database Object Type	Raj Satish	Financial Apps - Database
Modify the Environment definitions in Kintana	Raj Satish	Financial Apps - Database

Establish Communication Points and Visibility

You must determine the communication points and methods for providing visibility into your process and Package statuses. Kintana provides three helpful features to help you increase visibility:

- Notifications on the Workflow Step
- Portlets on the Dashboard
- Reports

This section lists the information that you should gather to define your Notifications. For more information on defining and using Portlets and Reports, refer to the following links:

- ["Configuring the Kintana Dashboard"](#)
- ["Kintana Reports"](#)

You can send a notification when a workflow step becomes eligible, has a specific outcome, or has a specific error. For each Workflow step in your process, collect the following information using the ["Configuration Worksheets"](#) on page 391:

Table 5-6. Information to gather for Workflow Step Notifications

Workflow step name	Include Notification for step? (Yes / No)
Step 1 - Name	Yes
Step 2 - Name	No
Step 3 - Name	No

For each step that requires a Notification, gather the following information:

Table 5-7. Information to gather for Workflow Step Notifications

Parameter	Description
Workflow Step Name	The name of the step that requires a workflow.

Table 5-7. Information to gather for Workflow Step Notifications

Parameter	Description
Notification Event (All, Eligible, Specific Result, Specific Error)	Specifies the event that triggers the notification. the possible values are ALL , ELIGIBLE , SPECIFIC RESULT , or SPECIFIC ERROR .
Value (for Specific Result)	Specifies that a notification is sent for the selected result.
Error (for Specific Error)	Specifies that a notification is sent for the selected error.
Recipient	Determine who should receive the message. you can choose to send the notification to users based on: USERNAME , EMAIL ADDRESS , SECURITY GROUP , STANDARD TOKEN , or USER DEFINED TOKEN .
Message	Determine what the message will say. Also determine if it will contain a link to the Package.

Example: ACME configures notifications

ACME determines that they would like to add a notification to the following steps:

Table 5-8. ACME - Workflow steps with Notifications

Workflow step name	When to send notification	Recipients
Migrate to Database environment	Failed (specific result)	Financial Apps - Database
Migrate to Server environment	Failed (specific result)	Financial Apps - Engineer
Compile Code	Failed (specific result)	Financial Apps - Engineer
QA - Test	Eligible	Financial Apps - QA
Rework	Eligible	Financial Apps - Engineer Financial Apps - Database
Schedule Migration to PROD	Eligible	Financial Apps - Manage Deployment

Chapter 6

Mapping your Process into a Kintana Workflow

This chapter provides an overview for how to set up a Kintana Workflow skeleton: all workflow steps, transitions and validations included in your process. It illustrates how the information gathered in “*Gathering Process Requirements and Specifications*” on page 49 or “*Configuration Worksheets*” on page 391 can be used to quickly build Workflow steps, transitions and validations.

This chapter discusses the following topics:

- *Building the Workflow Skeleton - Overview*
- *Create the Required Step Source*
- *Configure the Step’s Transition Values (Validation)*
- *Add Steps and Transitions to the Workflow Layout*

Note

This chapter presents a number of configuration options available to you. It does not, however, provide comprehensive instructions on implementing these configurations. For configuration instructions, see the referenced sections from within the following sections.

Building the Workflow Skeleton - Overview

For each Workflow that you create, follow this general process:

1. Enter the general Workflow information in a new WORKFLOW window. Enter the NAME and WORKFLOW SCOPE in the **WORKFLOW** tab.

2. Create any new step sources using the **WORKFLOW STEP SOURCES** window. This includes:
 - a. Creating decision steps.
 - b. Creating execution steps.
 - c. Creating subworkflow steps.
 - d. Creating any new validations used by the above steps.
3. Add the Workflow steps to the **LAYOUT** tab.
4. Add transitions between the Workflow steps.
5. Add Security to the Workflow.
6. Add Notifications to select Workflow steps.
7. Enable the Workflow.

Required Workflow Settings for Deployment Process

- The **WORKFLOW SCOPE** must be set to **PACKAGES**
- The Workflow must be **ENABLED**
- You must add steps and transitions to the **LAYOUT** of your Workflow
- You must allow Object Types to be used with the Workflow (**DELIVER SETTINGS** tab)
- You must enable Security for each Workflow step. This allows users to act on the step.



Note

- Workflows are created and configured using the Kintana Workbench.
- Users must have a Power License and have the proper Access Grants in order to create and edit a Workflow. See "[Kintana Security Model](#)" for details.

Create the Required Step Source

Kintana provides a number of standard Workflow step sources that you can add to your Workflow. These sources are preconfigured with standard validations (transition values), workflow events, and workflow scope. These available steps specify the following common attributes, which are expected to remain consistent across all Workflows which use that step source:

- The Validation associated with the step (and thus the list of valid transition values out of the step).
- The voting requirements of the step.
- The default timeout value for the step. Each step can be configured to have a unique timeout value.
- The icon used for the step within the graphical layout.

You can browse through all of the Workflow Step Sources in Kintana using the AVAILABLE WORKFLOW STEPS window in the WORKFLOW WORKBENCH. When Kintana does not have a step source that meets your process requirements, you need to create one.



If Kintana has a Workflow step source that meets your process requirements, you can copy and rename it. This can save configuration effort and avoid user processing errors. For example, if you need a step to route a Package Line based on the Object Type, copy and use the CHECK OBJECT TYPE Workflow step source that is delivered with Kintana.

Kintana recommends copying the step source so that you can use it uniquely for your processes. This allows you to control who can edit the step source, ensuring that your process isn't inadvertently altered by another Kintana user.

Create a new step source when the step requires any of the following:

- Unique Validation leaving the step (This is the step's transition.)
- Unique execution on the step: PL/SQL function, Token, SQL function, or Workflow Step Commands
- Different processing type: immediate vs. manual
- Specific Workflow Scope

- Unique combination of the above settings

The following sections discuss when and how to use specific settings in the Workflow Step Source:

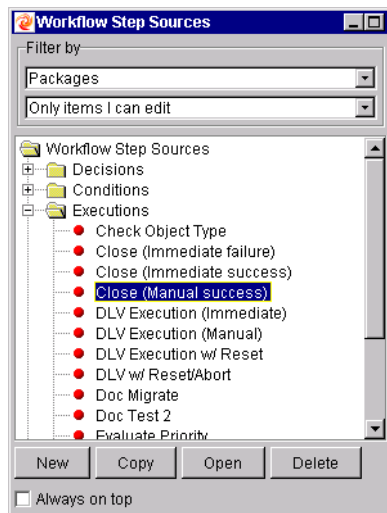
- [Creating a Workflow Step Source - Overview](#)
- [Creating a Decision Type Step](#)
- [Create an Execution Type Step](#)

Creating a Workflow Step Source - Overview

You can create new Workflow step sources from the WORKFLOW STEP SOURCES window on the Kintana Workbench. Using the information gathered in [“Gathering Process Requirements and Specifications”](#) on page 49 you can quickly create any required step sources.

To create a new Workflow Step Source:

1. Click the **CONFIGURATION** screen group and click the **WORKFLOWS** icon. The WORKFLOW WORKBENCH and WORKFLOW STEP SOURCES windows open.
2. Select the WORKFLOW STEP SOURCES window.



3. Select to FILTER BY **PACKAGES**.

4. Select the folder that corresponds to the type of Workflow step source that you would like to create. For example, to create an execution step, select the **EXECUTIONS** folder.
5. Click **NEW**. This opens a window that corresponds to the selected Workflow step source type. The **DECISION** and **EXECUTION** windows are shown below. For information on configuring Workflow Step Sources, see [“Creating a Decision Type Step”](#) on page 85 and [“Create an Execution Type Step”](#) on page 90.



Note

Condition step sources cannot be added, deleted or modified. Kintana supports a set number of process Condition steps. These can be added to the Workflow layout just as any other Workflow step source.

If you select a Condition step in the **WORKFLOW STEP SOURCES** window, the **NEW** button will not be enabled.

See [“Using Condition Steps”](#) on page 271 for more information.

The 'Decision' dialog box has a title bar with a close button. It contains several tabs: 'Decision', 'Ownership', 'User Data', and 'Used By'. The 'Decision' tab is active. It features a 'Name' text field, a 'Workflow Scope' dropdown menu set to 'Packages', and a 'Description' text area. Below these is a 'Validation' field with a list icon and 'New' and 'Open' buttons. To the right is a 'Decisions Required' dropdown menu set to 'One'. At the bottom left, there is a 'Timeout' field with a 'Days' dropdown and an 'Icon' text field. At the bottom right, there is an 'Enabled' section with radio buttons for 'Yes' (selected) and 'No'. At the very bottom are 'OK', 'Save', and 'Cancel' buttons. A status bar at the bottom left shows 'Ready'.

The 'Execution' dialog box has a title bar with a close button. It contains several tabs: 'Execution', 'Ownership', 'User Data', and 'Used By'. The 'Execution' tab is active. It features a 'Name' text field, a 'Workflow Scope' dropdown menu set to 'ALL', and a 'Description' text area. Below these is an 'Execution Type' dropdown menu set to 'Built-in Workflow Event' and a 'Workflow Event' dropdown menu set to 'wf_close_success'. To the right is a 'Processing Type' dropdown menu set to 'Manual'. Below these is a 'Validation' field with a list icon and 'New' and 'Open' buttons. At the bottom left, there is a 'Timeout' field with a 'Days' dropdown and an 'Icon' text field. At the bottom right, there is an 'Enabled' section with radio buttons for 'Yes' (selected) and 'No'. At the very bottom are 'Verify', 'OK', 'Save', and 'Cancel' buttons. A status bar at the bottom left shows 'Ready'.

6. Enter the required information and any optional information needed to define the step. See [“Creating a Decision Type Step”](#) on page 85 and [“Create an Execution Type Step”](#) on page 90 for detailed information about setting up the steps.
7. Select **YES** in the ENABLED radio button to be able to use this step in a Workflow.
8. Click the **OWNERSHIP** tab. Select which Ownership Groups will have the ability to edit this Execution or Decision.

9. Click **OK** to save the changes and close the EXECUTION or DECISION window.

The new Workflow step source is now included in the WORKFLOW STEP SOURCES window. It can be used in any new or existing Workflow with the corresponding WORKFLOW SCOPE.

Related Topics:

- [“Creating a Decision Type Step”](#) on page 85
- [“Create an Execution Type Step”](#) on page 90
- [“Advanced Workflow Topics”](#) on page 257

Workflow Step Source Configuration and Usage Restrictions

The following restrictions apply to Workflow step sources:

- You can not delete a step source that is being used in a Workflow.
- You can not change a validation for a step source that is being used. If you need to change the validation, copy the step source and configure a new validation.
- The Workflow step source must be ENABLED to use them on a Workflow.
- You can only add step sources to a Workflow when the Workflow has a matching WORKFLOW SCOPE, or the step source has a scope of **ALL**.
- You can not delete a step from a Workflow that has been used to process a Package. This would compromise data integrity. Instead of deleting the step, remove all transitions to and from the step and disable it. You can also select DISPLAY = **NEVER** in the Workflow step window to hide the step from users who are processing Packages using that Workflow.

Creating a Decision Type Step

Using the information gathered in [“Gathering Process Requirements and Specifications”](#) on page 49 you can configure decision steps for use on your Workflow. The following sections show how to use the information contained in [“Configuration Worksheets”](#) on page 391 when building your steps.

1. *Enter the general information on the Decision step source* (name, scope, description)
2. *Select a Validation*
3. *Specify the voting requirements on the step*
4. *Specify the default timeout value*

Information used to create the step source.

Table D-3. Workflow Step [Decision] -- Step Number ____.

	Value
Step Name	
Goal / Result of Step	
Validation*	
Decisions Required (Vote on Step's outcome?)	<ul style="list-style-type: none"> • One • At Least One • All
Timeout (Days)	
Security (who can act on step):	
<ul style="list-style-type: none"> • Security Group • User Name • Standard Token • User Defined Token 	
Include Notification (Yes/No)	
Notification Event	
Notification Recipient:	
<ul style="list-style-type: none"> • Username • Email Address • Security Group • Standard Token • User Defined Token 	
Notification Message	

Validation Information ³	Value
Existing Validation?	
New Validation?	
Validation Type: (text field, auto-complete, drop down list, etc.)	
Validation Definition (list of values or SQL)	

← Other information is used when adding the step source to the Workflow layout.

Figure 6-1 Worksheet - Information used to create the decision step source.

Enter the general information on the Decision step source

Enter the following information in the DECISION window.

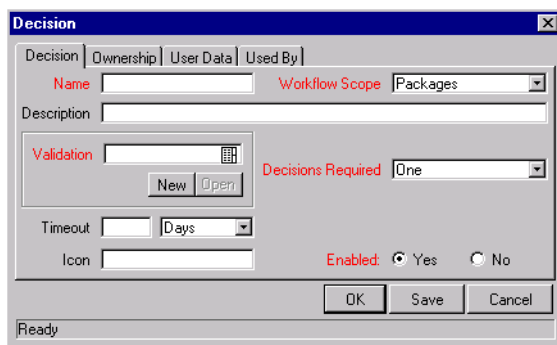


Table 6-1. Decision step source worksheet to window.

Field in Decision Window	Description
NAME	This is the name that describes the step source. The step can be renamed when added to the Workflow.
WORKFLOW SCOPE	Describes the type of workflow that will be using this step source. This should be set to ALL or PACKAGES for deployment processes.
DESCRIPTION	Description of the step source.
VALIDATION	Specifies the possible values that can exit the Workflow step. See <i>“Configure the Step’s Transition Values (Validation)”</i> on page 109.
DECISIONS REQUIRED	This specifies the number of people who need to approve a specific step. See <i>“Specify the voting requirements on the step”</i> on page 88.
TIMEOUT	If this Workflow Step remains eligible for the value entered in the Timeout value, the Package can be configured to send an appropriate Notification, as well as escalate to other steps in the Workflow.
ICON	You can specify a different graphic to represent steps of this source for use on the Workflow layout tab. This graphic needs to exist in the icons subdirectory of the directory specified by the server parameter BASE_PATH. If it is left blank, a default icon is used.
ENABLED	The step source must be enabled in order to add it to the Workflow layout.

Select a Validation

Select a Validation that has the transition values required for leaving the step. If Kintana doesn't provide a Validation that meets your requirements, you can create a new one from the WORKFLOW STEP SOURCE window. See [“Validations”](#) on page 287 for a list of Kintana's seeded Validations

See [“Configure the Step's Transition Values \(Validation\)”](#) on page 109 for additional details.

Specify the voting requirements on the step

When a Decision step is defined, the number of decisions required for that Workflow Step can be defined. [Figure 6-2](#) displays the available options for the DECISIONS REQUIRED field.

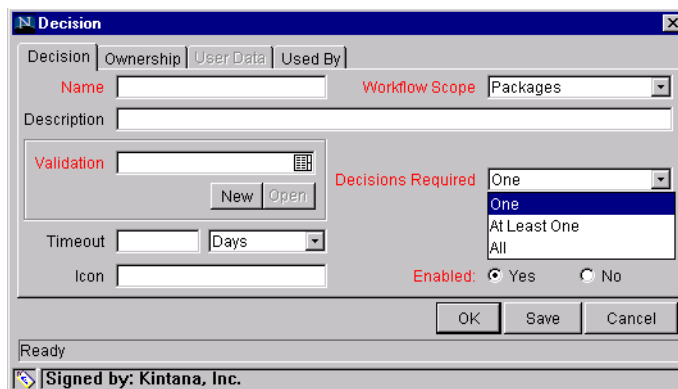


Figure 6-2 Decision Window - Decisions Required Drop Down List

The following choices are available for the Decisions Required:

- **ONE**

If **ONE** is selected, the Workflow Step can progress if any one user who is eligible to act on this step makes a decision.

- **AT LEAST ONE**

A Timeout period must be defined to use this choice. When **AT LEAST ONE** is selected for the Workflow Step, the step waits for the voters to vote on this step for a predefined amount of time, designated as the Timeout. If all voters mark their decisions before the timeout period, it takes the

cumulative decision as the decision for the step and proceeds forward. If any of the voting results differ before the ‘Timeout’ period, the step will immediately result in a ‘No consensus’ outcome.

Note

You can define **SPECIFIC ERRORS** in Workflow Steps such as ‘**TIMEOUT**’ and ‘**NO CONSENSUS**’ as either **Success** or **Failure** in the **DEFINE TRANSITION** window. For more information, see “[Adding Transitions Between Steps](#)” on page 122>.

If all voters decide on **APPROVE**, the final decision is **APPROVE**. If all voters decide on **NOT APPROVED**, the final decision is **NOT APPROVED**. If some voters decide on **APPROVED** and one voter decides on **NOT APPROVED**, the result is ‘**NO CONSENSUS**.’

If at the end of the Timeout, only a few voters (or only one voter) have cast their vote, the cumulative decision of the voters that voted will be used.

If at the end of the Timeout no one has voted, the step will result in a Timeout.

- **ALL**

The **ALL** step is also commonly used along with a specified Timeout period. Selecting **ALL** makes it mandatory for all voters to vote on the Workflow Step. The Workflow Step waits until the Timeout period for the voters to vote. If all voters vote, the cumulative decision is considered. If some or none of the voters voted, the step remains open or closes due to a timeout, depending on the configuration.

When using **ALL** or **AT LEAST ONE** all users must unanimously approve or not approve one of the validation’s selections. Otherwise, the result is **NO CONSENSUS**.

Specify the default timeout value

A timeout specifies the amount of time that a step can stay eligible for completion before completing with an error (if **Decisions Required** is **ALL** or **ONE**) or completing with a result (if **Decisions Required** is **AT LEAST ONE**). Timeouts can be by minute, hour, weekday or week. Timeout parameters for Executions and Decisions are a combination of a numerical timeout value and a timeout unit (such as weekdays).

If a Workflow Step remains eligible for the value entered in the Timeout value, the Package can send an appropriate Notification and proceed to other steps in the process.

Timeouts can be uniquely configured for each Workflow Step in the **LAYOUT** tab. The timeout value specified in the step source acts as the default timeout value for the step. When you add a step to the Workflow using this step source, you can specify a different timeout value for the step.

Create an Execution Type Step

Using the information collected in “*Gathering Process Requirements and Specifications*” on page 49 you can configure execution steps for use on your Workflow. The following sections show how to use the information contained in “*Configuration Worksheets*” on page 391 to building your steps.

1. *Enter the general information on the Execution step source* (name, scope, description)
2. *Define the Executions*
3. *Select a Validation*
4. *Specify the default timeout value*

Information used to create the step source.

Table D-2. Workflow Step [Execution] -- Step Number ____.

	Value
Step Name	
Goal / Result of Step	
Validation*	
Execution Type**	
Processing Type (IMMEDIATE or MANUAL execution?)	
Timeout (Days)	
Source Environment (Group)	
Dest Environment (Group)	
Security (who can act on step):	
<ul style="list-style-type: none"> • Security Group • User Name • Standard Token • User Defined Token 	
Include Notification (Yes/No)	
Notification Event	
Notification Recipient:	
<ul style="list-style-type: none"> • Username • Email Address • Security Group • Standard Token • User Defined Token 	
Notification Message	

Validation Information*	Value
Existing Validation?	
New Validation?	
Validation Type: (text field, auto-complete, drop down list, etc.)	
Validation Definition (list of values or SQL)	

Execution Type**	Value
Built-in Workflow Event:	
<ul style="list-style-type: none"> • Execute Commands • Close • Jump / Receive • Ready for Release • Return from Subworkflow 	
PL/SQL Function	
Token	
SQL Statement	
Workflow step commands	

Figure 6-3 Worksheet - Information used to create the execution step source.

Enter the general information on the Execution step source

Enter the following information in the EXECUTION window.

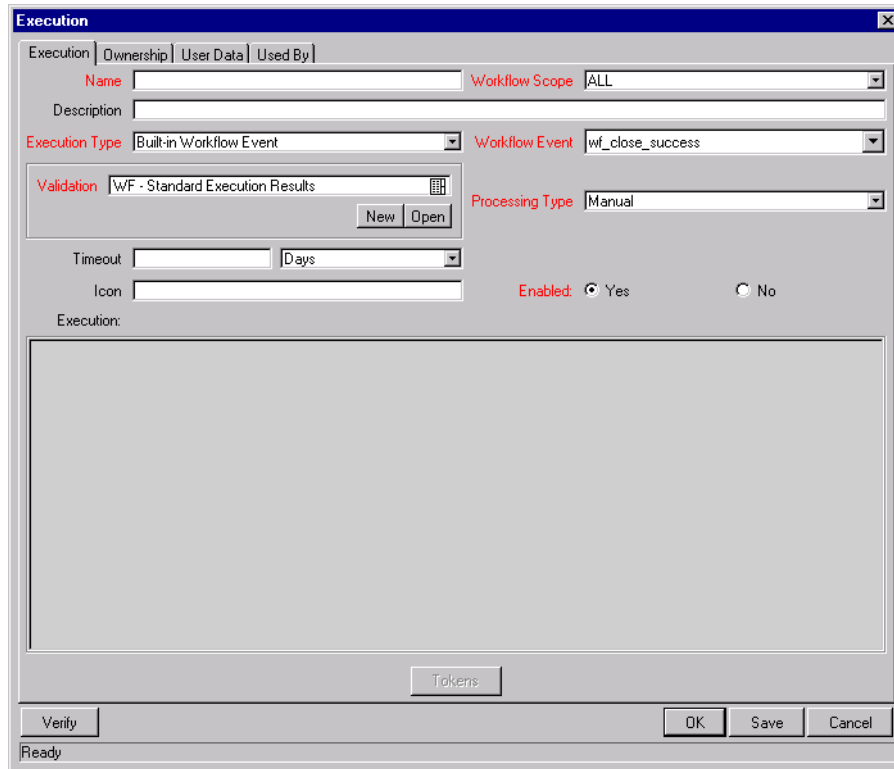


Table 6-2. Execution step source worksheet to window.

Field in Execution Window	Description
NAME	This is the name that describes the step source. The step can be renamed when added to the Workflow.
WORKFLOW SCOPE	Describes the type of workflow that will be using this step source. This should be set to ALL or PACKAGES for deployment processes.
DESCRIPTION	Description of the step source. This should specify the execution that will occur.

Table 6-2. Execution step source worksheet to window.

Field in Execution Window	Description
EXECUTION TYPE	<p>Used to select the type of execution to be performed. The choices are:</p> <p>Built-in Workflow Event: Executes a predefined command and returns its result as the result of the Step.</p> <p>SQL Statement: Executes a SQL statement and returns its result as the result for the Step.</p> <p>PL/SQL Function: Runs a PL/SQL function and returns its result as the result for the Step.</p> <p>Token: Calculates the value of a token and returns its value as the result for the Step.</p> <p>Workflow Step Commands: Executes a set of commands, independent of an Object, at a Workflow Step.</p>
WORKFLOW EVENT - (Built-In Workflow Event)	<p>For Executions of type BUILT-IN WORKFLOW EVENT, the specific event to perform must be selected. The available choices in the drop down list depend on which Workflow Scope has been selected. The choices are:</p> <p>execute_object_commands: Executes the Object Type commands for a Package Line.</p> <p>rm_ready_for_release: Sets a Package Line ready to be added to a Kintana Release.</p> <p>wf_close_success: Sets the Package Line as closed with an end status of 'Success.'</p> <p>wf_close_failure: Sets the Package Line as closed with an end status of 'Failed.'</p> <p>wf_jump: (Kintana Deliver and Kintana Create only) Instructs the Workflow to proceed to a corresponding Receive Workflow Step in another Kintana Workflow.</p> <p>wf_receive: (Kintana Deliver and Kintana Create only) Instructs the Workflow to receive a Jump Workflow Step and continue processing a Request or Package Line initiated in another Workflow.</p> <p>wf_return: (Kintana Deliver and Kintana Create only) Used to route a Subworkflow process back to its parent Workflow.</p>
PL/SQL FUNCTION	<p>For Executions of type PL/SQL Function, the actual function to run. The results of the function will determine the outcome of the step.</p> <p>Note: The results of the function must be a subset of the Validation values for that step.</p>
TOKEN	<p>For Executions of type Token, the Token that will be resolved. The results of the Token resolution will determine the outcome of the step.</p>

Table 6-2. Execution step source worksheet to window.

Field in Execution Window	Description
SQL STATEMENT	For Executions of type SQL Statement, the actual query to run. The results of the query will determine the outcome of the step. Note: The results of the query must be a subset of the Validation values for that step.
WORKFLOW STEP COMMANDS	For Executions of type Workflow Step Commands, the actual commands to run. The commands will result with a SUCCEEDED or FAILED value. Use a validation with those values to enable transitioning out of the step based on the execution results.
PROCESSING TYPE	Indicates whether the Execution is performed immediately (IMMEDIATE) when the Step becomes eligible or whether the Execution needs to be manually activated by a user (MANUAL).
VALIDATION	Specifies the possible values that can exit the Workflow step. See <i>“Configure the Step’s Transition Values (Validation)”</i> on page 109.
TIMEOUT	If this Workflow Step remains eligible for the value entered in the Timeout value, the Package can be configured to send an appropriate Notification, as well as escalate to other steps in the Workflow. See <i>“Specify the default timeout value”</i> on page 109.
ICON	You can select a different graphic to represent this steps of this step source. This graphic needs to exist in the icons subdirectory of the directory specified by the server parameter BASE_PATH. If it is left blank, a default icon is used.
ENABLED	The step source must be enabled in order to add it to the Workflow layout.

Define the Executions

Execution steps are used to perform specific actions. Kintana provides a number of number of built in Workflow events for processing some common execution events (running Object Type commands, closing a Package, etc.). Kintana also provides the flexibility to create your own executions based on SQL, PL/SQL, Token resolution, and Kintana commands.

This section discusses when to use specific types of executions and provides references for configuring these executions.

Execution steps can be created to perform the following actions:

- Execute the Object Type Commands and transition based on the success or failure of those commands.
- Close the Package and mark it as a SUCCESS
- Close the Package and mark it as a FAILURE
- Transition (jump) to another Workflow that is processing a Request
- Receive control from another Workflow that is processing a Request
- Set a Package READY FOR RELEASE for use in Kintana's release management
- Return from a subworkflow to the parent workflow
- Execute a PL/SQL function and then transition based on the result
- Execute a SQL statement and then transition based on the result
- Evaluate a Token and then transition based on the result
- Execute a number of system level commands and then transition based on the success or failure of those commands.
 - o Example: Start a server
 - o Example: Stop a server

Execute the Object Type Commands

Different objects (stored on the Package Line) require unique processing at different points in your process. For example, the commands needed to migrate a file are different than the commands needed to migrate data. Therefore, Kintana allows you to program the commands on a per-Object Type basis. You can then configure your Workflow to execute the Object Type commands at a specific step in the process. Each object (Package Line) will run its own commands, ensuring the correct execution for that Object Type.



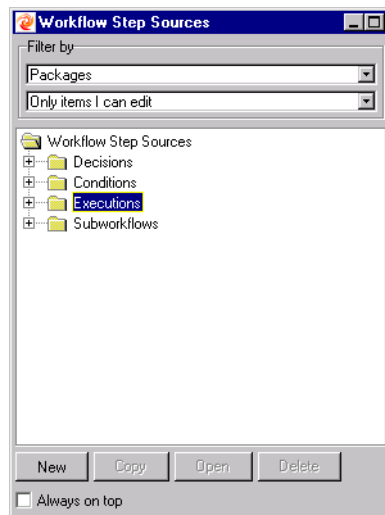
Note

Kintana provides a system step source that executes the Object Type commands. Use a copy of this step source to process this action. You can modify the copy if it doesn't meet your exact specifications (Validation, Processing Type, etc.).

Step Source = DLV Execution w/ Reset

To create an execution step source that will execute the Object Type commands for each Package Line in your Package:

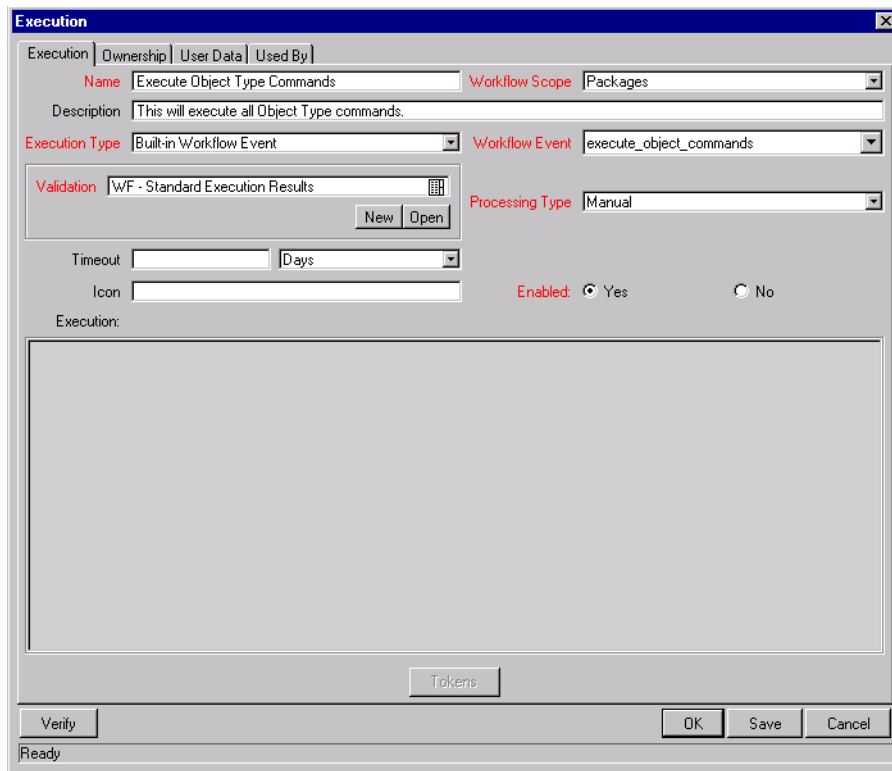
1. Open the WORKFLOW WORKBENCH.
2. Select the WORKFLOW STEP SOURCES window.
3. Select the EXECUTION directory.



4. Click **NEW**. The EXECUTION window opens.
5. Enter the following information:

Field in Execution Window	Value
NAME	Enter a descriptive name for the step source.
WORKFLOW SCOPE	PACKAGES
EXECUTION TYPE	Built-in Workflow Event

Field in Execution Window	Value
WORKFLOW EVENT	execute_object_commands
PROCESSING TYPE	MANUAL or IMMEDIATE
VALIDATION	WF - STANDARD EXECUTION RESULTS (This is the default selection. You can select another existing or create a new validation.)
ENABLED	YES



Close the Package Line and mark it as a Success

You can create an execution step that closes a Package Line. When all Package Lines are closed, the Package will close. Each Package Workflow should resolve with a closed Package. You can then report on all Packages that were closed successfully.



This type of step is also required when integrating Request and Package Workflows. If a Package has been created using the Request Workflow step “Create Package and Wait,” the Request Workflow will not proceed until the Package Workflow has closed.

To configure an execution step source to close a Package Line and mark it as a Success, create an execution step source with the following settings:

Field in Execution Window	Value
NAME	Enter a descriptive name for the step source.
WORKFLOW SCOPE	PACKAGES
EXECUTION TYPE	Built-in Workflow Event
WORKFLOW EVENT	wf_close_success
PROCESSING TYPE	MANUAL or IMMEDIATE
VALIDATION	WF - STANDARD EXECUTION RESULTS (This is the default selection. You can select another existing or create a new validation.)
ENABLED	YES



Kintana provides a system step source that performs this task. Use this step source unless it doesn't meet your exact specifications (Validation, Processing Type, etc.).

STEP SOURCE = **CLOSE (IMMEDIATE SUCCESS)** or **CLOSE (MANUAL SUCCESS)**

Close the Package and mark it as Failed

You can create an execution step that closes a Package Line and marks it as Failed.

To configure an execution step source to close a Package and mark it as a Failed, set the following in the EXECUTION window:

Field in Execution Window	Value
NAME	Enter a descriptive name for the step source.
WORKFLOW SCOPE	PACKAGES
EXECUTION TYPE	Built-in Workflow Event
WORKFLOW EVENT	wf_close_failure
PROCESSING TYPE	MANUAL or IMMEDIATE
VALIDATION	WF - STANDARD EXECUTION RESULTS (This is the default selection. You can select another existing or create a new validation.)
ENABLED	YES

The screenshot shows the 'Execution' configuration window. The 'Name' field is empty. The 'Workflow Scope' is set to 'Packages'. The 'Execution Type' is 'Built-in Workflow Event'. The 'Workflow Event' is 'wf_close_failure'. The 'Validation' is 'WF - Standard Execution Results'. The 'Processing Type' is 'Manual'. The 'Enabled' radio buttons are set to 'Yes'. The 'Timeout' field is empty, and the 'Days' dropdown is set to 'Days'. The 'Icon' field is empty. The 'Tokens' button is visible. The status bar shows 'Ready'.



Kintana provides a system step source that performs this task. Use this step source unless it doesn't meet your exact specifications (Validation, Processing Type, etc.).

STEP SOURCE = **CLOSE (IMMEDIATE FAILURE)**

Transition (jump) to a Workflow that is Processing a Request

Package Workflows can communicate with Request Workflows at specific jump and receive points. To effectively utilize this functionality, you need to properly configure both the jump and receive execution Workflow steps. See [“Package - Request Workflow Integration”](#) on page 262 for additional details.

Receive control from a Workflow that is Processing a Request

Package Workflows can communicate with Request Workflows at specific jump and receive points. To effectively utilize this functionality, you need to properly configure both the jump and receive execution Workflow steps. See [“Package - Request Workflow Integration”](#) on page 262 for additional details.

Set a Package “Ready for Release” for use in Kintana’s Release Management

To configure an execution step source to feed a Package into a Kintana Release, set the following in the Execution window:

Field in Execution Window	Value
NAME	Enter a descriptive name for the step source.
WORKFLOW SCOPE	PACKAGES
EXECUTION TYPE	Built-in Workflow Event
WORKFLOW EVENT	rm_ready_for_release
PROCESSING TYPE	MANUAL or IMMEDIATE
VALIDATION	RM - READY FOR RELEASE

Field in Execution Window	Value
ENABLED	YES



Kintana provides a system step source that performs this task. Use this step source unless it doesn't meet your exact specifications (Validation, Processing Type, etc.).

STEP SOURCE = **READY FOR RELEASE**

See "[Configuring a Release Management System](#)" for more detailed information on using Kintana's Release Management functionality.

Return from a Subworkflow to the Parent Workflow

Execution steps can be configured to automatically return from a subworkflow to its parent workflow. Create an execution step (to be used on the Subworkflow) with the following configuration:

Field in Execution Window	Value
NAME	Enter a descriptive name for the step source.
WORKFLOW SCOPE	PACKAGES
EXECUTION TYPE	Built-in Workflow Event
WORKFLOW EVENT	wf_return
PROCESSING TYPE	MANUAL or IMMEDIATE
VALIDATION	WF - STANDARD EXECUTION RESULTS (This is the default selection. You can select another existing or create a new validation.)
ENABLED	YES

See "[Using Subworkflows](#)" on page 257 for additional details.



Note

For a Package Line or Request to transition back to the parent Workflow, the Subworkflow must contain a Return step. The transitions leading into the Return step must match the Validation established for the Subworkflow Step. Users must verify that the Validation defined for the Subworkflow Step is synchronized with the transitions entering the Return Step. The Subworkflow Validation is defined in the Workflow window.



Tip

Kintana provides a system step source that performs this task. Use this step source unless it doesn't meet your exact specifications (Validation, Processing Type, etc.).

Step Source = RETURN FROM SUBWORKFLOW

Execute a PL/SQL function and then transition based on the result

A PL/SQL function execution step runs a PL/SQL function and returns its results as the result of that workflow step. Include an execution step with the following source configuration:

Field in Execution Window	Value
NAME	Enter a descriptive name for the step source.
WORKFLOW SCOPE	PACKAGES
EXECUTION TYPE	PL/SQL Function
PROCESSING TYPE	MANUAL or IMMEDIATE
VALIDATION	Select or create a validation that includes all of the possible values of the SQL query. Tip: You can create a validation validated by SQL. Use the same SQL from the execution minus the WHERE clause.
EXECUTION	Enter the SQL query.
ENABLED	YES

Execute a SQL statement and then transition based on the result

SQL statement Execution steps are used when a workflow needs to be routed based on the result of a query. A SQL statement execution step runs a SQL query and returns its results as the result of that workflow step.

Include an execution step with the following source configuration:

Field in Execution Window	Value
NAME	Enter a descriptive name for the step source.
WORKFLOW SCOPE	PACKAGES
EXECUTION TYPE	SQL Statement
PROCESSING TYPE	MANUAL or IMMEDIATE
VALIDATION	Select or create a validation that includes all of the possible values of the SQL query. Tip: You can create a validation validated by SQL. Use the same SQL defined for the execution minus the WHERE clause.
EXECUTION	Enter the SQL query.
ENABLED	YES

Configuration notes:

- Only use Select statements
- You can use Kintana Tokens within the WHERE clause
- Query must return only 1 value

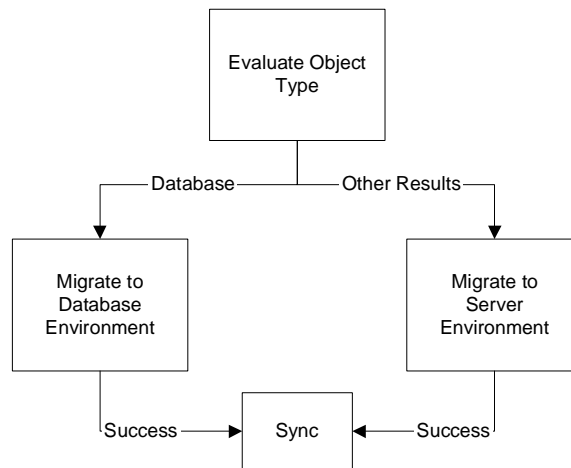
Evaluate a Token and then transition based on the result

Kintana has workflow Execution steps that may be used to set up data-dependent rules for the routing of workflow processes. Token Execution steps enable a workflow to be routed based on the value of any field within a particular Kintana entity. A Token Execution step references the value of a given Token and uses that value as the result of the workflow step.

You can transition based on the value stored in Kintana by using Tokens in your Execution step. Include an execution step with the following source configuration:

Field in Execution Window	Value
NAME	Enter a descriptive name for the step source.
WORKFLOW SCOPE	PACKAGES
EXECUTION TYPE	Token
PROCESSING TYPE	MANUAL or IMMEDIATE
VALIDATION	Select or create a validation that includes all of the possible values of the resolved Token. Tip: You can create a validation validated by SQL. Include the same token in the SQL.
EXECUTION	Enter the Token for the value that you would like to transition based on.
ENABLED	YES

For example, ACME needs to deploy changes to different servers depending on the type of object being deployed.



They decide to use an Execution step to automatically evaluate the type of object and route the Package line accordingly. They create an execution step source (Evaluate Object Type), configured with the following parameters.

Field in Execution Window	Value
NAME	Evaluate Object Type
WORKFLOW SCOPE	PACKAGES
EXECUTION TYPE	Token
PROCESSING TYPE	IMMEDIATE
VALIDATION	DLV - OBJECT TYPE - ENABLED
EXECUTION	[PKGL.OBJECT_TYPE]
ENABLED	YES

Execute a number of system level commands and then transition based on the success or failure of those commands.

System level commands can be run for execution steps of the following EXECUTION TYPE: **BUILT-IN WORKFLOW EVENT (EXECUTE_OBJECT_COMMANDS)** and **WORKFLOW STEP COMMANDS**. When either the Workflow or the Object Type commands execute at this step, the commands will either Succeed or Fail. To transition based on these results, the code for the validation values must have the following values:

- **SUCCESS**
- **FAILURE:**

Validation : WF - Standard Execution Results

Name: WF - Standard Execution Results
Description: WF - Standard Execution Results
Enabled: Use in Workflow?

Component Type: Drop Down List
Validated By: List

Validation Values:

Seq	Code	Meaning	Description	Enabled	Default
1	SUCCESS	Succeeded	Succeeded	Y	Y
2	FAILURE	Failed	Failed	Y	N

New Edit Delete Copy From ↑ ↓

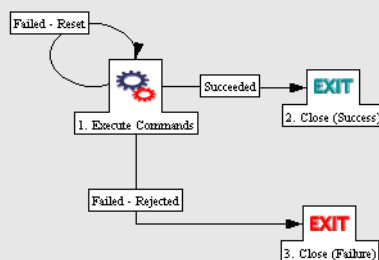
Used By Ownership OK Save Cancel

Ready (Read-Only, Seed Data)



You may want to retain the option of resetting failed execution steps, rather than immediately transitioning along a “failed” path. This is often helpful when troubleshooting the execution. To configure this:

1. Create an execution step source to execute the Workflow or Object Type commands.
2. Create a Validation with the following Validation Values.
 - a. SUCCEEDED
 - b. FAILED
 - c. FAILED - RESET
 - d. FAILED - REJECTED
3. Add the step to the Workflow **LAYOUT** tab.
4. Add transitions based on the following Specific Results:
 - a. SUCCEEDED
 - b. FAILED - RESET -- set the transition to return back into the same step.
 - c. FAILED - REJECTED



When the commands execute successfully, they will follow the Success transition path. However, when the commands fail, they will not transition out of the step because no transition has been defined for the Failed result. The user has to manually select the Package Line step and select Failed - Retry. The execution will re-run.

Select a Validation

Select a Validation that has the transition values required for leaving the step. If Kintana doesn't provide a Validation that meets your requirements, you can create a new one from the WORKFLOW STEP SOURCE window. See “[Validations](#)” on page 287 for a list of Kintana's seeded Validations

See “[Configure the Step's Transition Values \(Validation\)](#)” on page 109 for additional details.

Specify the default timeout value

Timeouts in the execution steps can be set at two levels:

- Step level: the amount of time that a step is eligible before completing with an error. This is set in the EXECUTION window.
- Command level: the amount of time that an execution is allowed to run before completing with an error. This applies to the Workflow Step Commands and Object Type Commands only. It is set in the COMMAND window.

Timeouts can be by minute, hour, weekday or week. Timeout parameters for Executions and Decisions are a combination of a numerical timeout value and a timeout unit (such as weekdays).

If this Workflow Step remains eligible for the value entered in the Timeout value, the Package can send an appropriate Notification and proceed to other steps in the Workflow.

Timeouts can be uniquely configured for each Workflow Step in the LAYOUT tab. The timeout value specified in the step source acts as the default timeout value for the step. When you add a step to the Workflow using this step source, you can specify a different timeout value for the step.

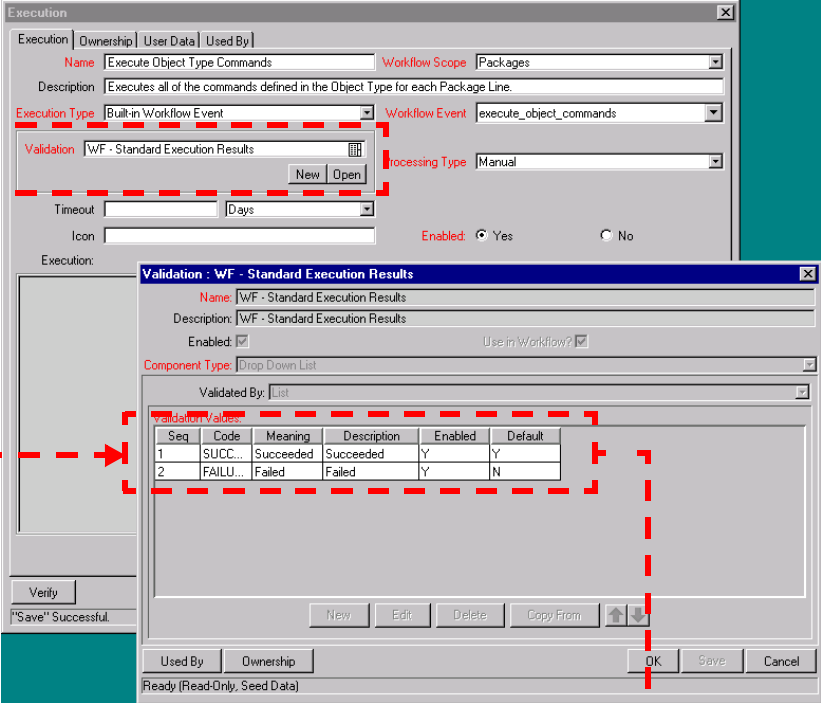
Configure the Step's Transition Values (Validation)

Kintana Workflows can be configured to transition based on values automatically returned from an execution or values selected by the user. For each Workflow step, you must define all of the possible values for the step's transition. This is set in the Validation field on the EXECUTION window or the

Configuring a Deployment and Distribution System

DECISION window. The Validation dictates the values in the SPECIFIC RESULT section on the ADD TRANSITION window.

1. Validation specifies all possible results for the step.



The Execution dialog box shows the following configuration:

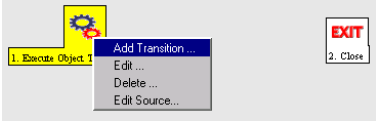
- Name: Execute Object Type Commands
- Workflow Scope: Packages
- Description: Executes all of the commands defined in the Object Type for each Package Line.
- Execution Type: Built-in Workflow Event
- Workflow Event: execute_object_commands
- Validation: WF - Standard Execution Results
- Processing Type: Manual
- Timeout: Days
- Icon: [Empty]
- Enabled: Yes

The Validation dialog box shows the following configuration:

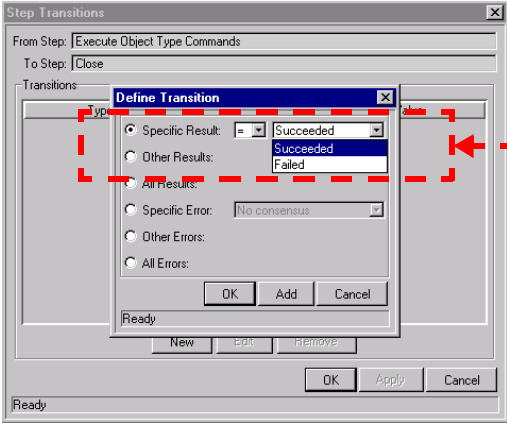
- Name: WF - Standard Execution Results
- Description: WF - Standard Execution Results
- Enabled:
- Use in Workflow?:
- Component Type: Drop Down List
- Validated By: List

Seq	Code	Meaning	Description	Enabled	Default
1	SUCC...	Succeeded	Succeeded	Y	Y
2	FAILU...	Failed	Failed	Y	N

2. Add a transition between two steps in the Workflow Layout tab.



3. Optionally base the transition on the values defined in the Validation.



The Step Transitions dialog box shows the following configuration:

- From Step: Execute Object Type Commands
- To Step: Close
- Transitions: Define Transition

The Define Transition dialog box shows the following configuration:

- Specific Result: Succeeded
- Other Results: Succeeded, Failed
- All Results: [Empty]
- Specific Error: No consensus
- Other Errors: [Empty]
- All Errors: [Empty]

When you specify the Validation for the execution step source, you specify all possible transition values in the Validation. When you use that step source on

the Workflow (add it to the Layout tab), you can decide to transition on one of the specific results, or a number of other transition options:

- Other Results
- All Results
- Specific Error
- Other Errors
- All Errors

Validations and Execution Type Relationships

There is a correlation between the Validation and the Execution Type. For data-dependent transitions (Token, SQL, PL/SQL), the Validation must contain all possible values of the query or token resolution. Otherwise, the execution step could result in a value that is not defined for the process, and the Package Line could become stuck in a Workflow step.

For most Built-In Workflow Events and executions that run commands, the Validation often includes the standard Workflow results (Success or Failure). If the commands or event execute without error, the result of Success is returned. Otherwise, Failure is returned.

The following table summarizes this relationship between Validations and Execution types.

Table 6-3. Relationship between Validation and Execution Type

Execution Types	Validation Notes
Built-in Workflow Event and Workflow Step Commands	Typically use a variation of the WF - Standard Execution Results Validation (SUCCEEDED or FAILED). A few of the Workflow Events have specific Validation Requirements: wf_return, wf_jump, wf_receive.
PL/SQL Function	Validation must contain all possible values returned by the function.
Token	Validation must contain all possible values for the Token.

Table 6-3. Relationship between Validation and Execution Type

Execution Types	Validation Notes
SQL Statement	<p>Validation must contain all possible values for the SQL query.</p> <p>Tip: You can use the same SQL in the Validation (drop down or auto-complete list) minus the WHERE clause.</p>



You can use the information captured in the “*Configuration Worksheets*” on page 391 to construct your validation.

Consider copying existing Validations to save time and ensure that the SQL or other Validation technique is configured properly.

Add Steps and Transitions to the Workflow Layout

Build your process graphically by dragging and dropping Workflow step sources onto the WORKFLOW window’s LAYOUT tab. When a Workflow step source is included in a Workflow, it is then referred to as a “Workflow step.” Use the information gathered in “*Gathering Process Requirements and Specifications*” on page 49 to determine the appropriate step source for the step. If a step source does not exist that meets your requirements (decision, execution, correct transition Validation values, processing type, etc.) you can create one.

When you add the step source to the LAYOUT tab, you will be required to provide supplemental information. The following sections discuss the configuration required when:

- *Adding Decision Steps*
- *Adding Execution Steps*
- *Adding a Subworkflow*
- *Adding Transitions Between Steps*

Adding Decision Steps

Using the information gathered in *“Gathering Process Requirements and Specifications”* on page 49 you can configure decision steps for use on your Workflow. The following sections show how to apply the information contained in *“Configuration Worksheets”* on page 391 to configure your steps:

- *Enter the general information on the Decision step*
- *Specify the Security*
- *Configure Notifications for the Workflow Step*

Table D-3. Workflow Step [Decision] -- Step Number ____.

	Value
Step Name	
Goal / Result of Step	
Validation*	
Decisions Required (Vote on Step's outcome?)	<ul style="list-style-type: none"> • One • At Least One • All
Timeout (Days)	
Security (who can act on step):	
<ul style="list-style-type: none"> • Security Group • User Name • Standard Token • User Defined Token 	
Include Notification (Yes/No)	
Notification Event	
Notification Recipient:	
<ul style="list-style-type: none"> • Username • Email Address • Security Group • Standard Token • User Defined Token 	
Notification Message	

Validation Information [†]	Value
Existing Validation?	
New Validation?	
Validation Type: (text field, auto-complete, drop down list, etc.)	
Validation Definition (list of values or SQL)	

← Information used when adding the step source to the Workflow Layout

Figure 6-4 Information used to create the decision step.

Enter the general information on the Decision step

Enter the following information in the WORKFLOW STEP window.

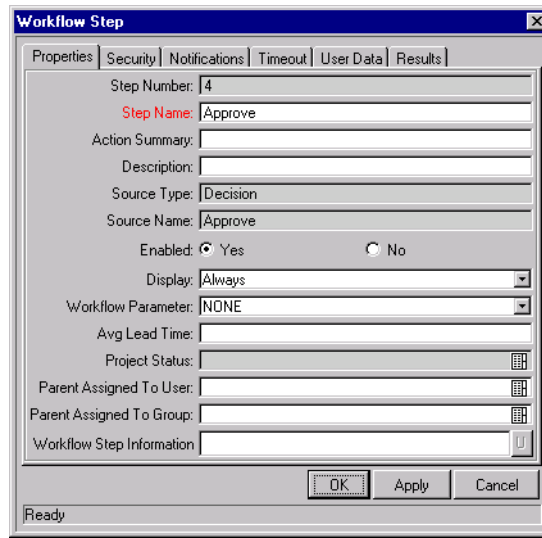


Table 6-4. Decision step worksheet to Workflow Step window.

Field/Tab on Workflow Step Window	Description
STEP NAME	This is the name of the step that appears on the Layout tab.
ACTION SUMMARY	The text that appears on the action button in the Package status panel.
DESCRIPTION	
ENABLED	
DISPLAY	Whether or not to show the step on the Package status panel.
WORKFLOW PARAMETER	Used to save the results of a workflow step for later use in the workflow processing.
AVG LEAD TIME	A user-specified metric for comparing actual performance to estimated goals. It does not affect any transactional logic.
PARENT ASSIGNED TO USER	If this field is not empty when the step becomes Eligible, the Assigned to User of the Package automatically changes to the user specified in the field.
PARENT ASSIGNED TO GROUP	If this field is not empty when the step becomes Eligible, the Assigned to Group of the Package automatically changes to the Security Group specified in the field.

Table 6-4. Decision step worksheet to Workflow Step window.

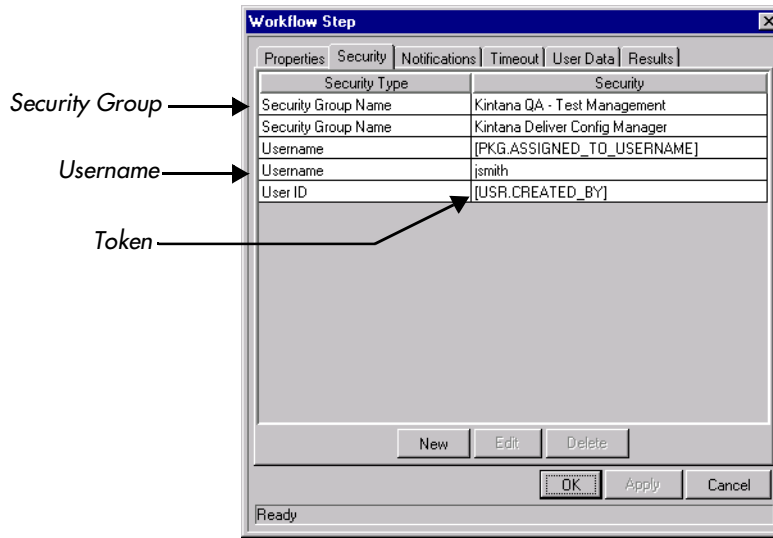
Field/Tab on Workflow Step Window	Description
WORKFLOW STEP INFORMATION	A text entry field in which any URL can be entered. This is where users can find documents, instructions or comments to aid them in processing the Workflow Step.
SECURITY TAB	See “ Setting Configuration Security ” on page 210 for configuration details.
NOTIFICATIONS TAB	Specify who will receive an email notification when this step becomes eligible or has a specific result or error.
TIMEOUT TAB	You can specify the Timeout value for this step. In the Timeout tab, select to use the Workflow step source timeout value or specify your own in the SPECIFIC VALUE section.

Specify the Security

“[Integrating Participants into Your Deployment System](#)” on page 195 provides information on setting up security for your deployment process. This includes such things as controlling who can create Packages and who can act on specific steps in the process. Security related directly to processing a Workflow step is configured in the WORKFLOW STEP window.

You can define who can act on the step by:

- Security Group
- Username
- Token (standard or user-defined)



Kintana recommends using Security Groups or dynamic access (Tokens) when defining the Workflow step security. You should avoid specifying a list of users to control an action; for example, specifying a list of users who can act on a Workflow step. If the list of users changes (due to an organizational reorganization), you would have to update that list in many places on the workflow. By using a Security Group instead of a list of users, you can update the Security Group once, and the changes are propagated throughout the Workflow steps.

Configure Notifications for the Workflow Step

“Setting Up Communication Paths” on page 215 provides information on setting up notifications for steps in your deployment process. This includes such things as configuring the notification’s recipients and message.

See the following sections for more details:

- *“Identify Participants and Security”* on page 71
- *“Setting Up Communication Paths”* on page 215

Adding Execution Steps

Using the information gathered in *“Gathering Process Requirements and Specifications”* on page 49 you can configure execution steps for use on your Workflow. The following sections show how to apply the information contained in *“Configuration Worksheets”* on page 391 to configure your steps:

- *Enter the general information on the Execution step*
- *Specify the Security*
- *Configure Notifications for the Workflow Step*

Information used when adding the step source to the Workflow layout.

Table D-2. Workflow Step [Execution] -- Step Number ____.

	Value
Step Name	
Goal / Result of Step	
Validation*	
Execution Type**	
Processing Type (IMMEDIATE or MANUAL execution?)	
Timeout (Days)	
Source Environment (Group)	
Dest Environment (Group)	
Security (who can act on step):	
<ul style="list-style-type: none"> • Security Group • User Name • Standard Token • User Defined Token 	
Include Notification (Yes/No)	
Notification Event	
Notification Recipient:	
<ul style="list-style-type: none"> • Username • Email Address • Security Group • Standard Token • User Defined Token 	
Notification Message	

Validation Information*	Value
Existing Validation?	
New Validation?	
Validation Type: (text field, auto-complete, drop down list, etc.)	
Validation Definition (list of values or SQL)	

Execution Type**	Value
Built-in Workflow Event:	
<ul style="list-style-type: none"> • Execute Commands • Close • Jump / Receive • Ready for Release • Return from Subworkflow 	
PL/SQL Function	
Token	
SQL Statement	
Workflow step commands	

Figure 6-5 Information used to create the execution step.

Enter the general information on the Execution step

Enter the following information in the **PROPERTIES** tab in the **WORKFLOW STEP** window.

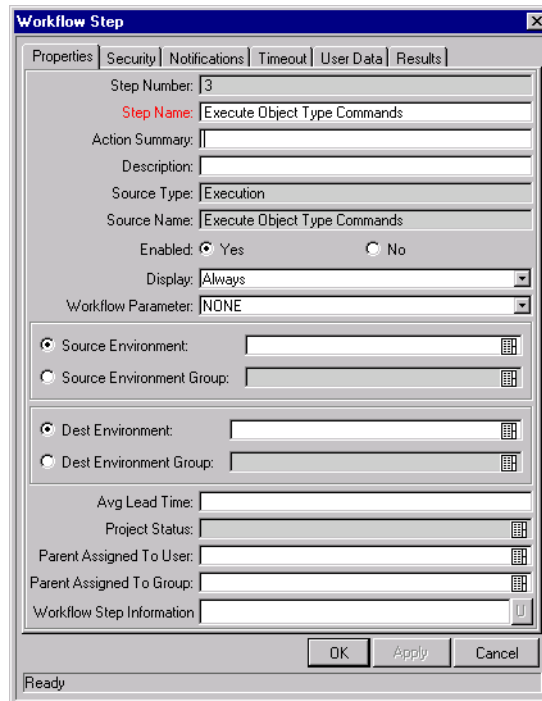


Table 6-5. Execution step worksheet to Workflow Step window mapping.

Field on Workflow Step Window	Description
STEP NAME	This is the name of the step that appears on the Layout tab.
ACTION SUMMARY	The text that appears on the action button in the Package status panel.
DESCRIPTION	
ENABLED	
DISPLAY	Whether or not to show the step on the Package status panel.
WORKFLOW PARAMETER	Used to save the results of a workflow step for later use in the workflow processing.

Table 6-5. Execution step worksheet to Workflow Step window mapping.

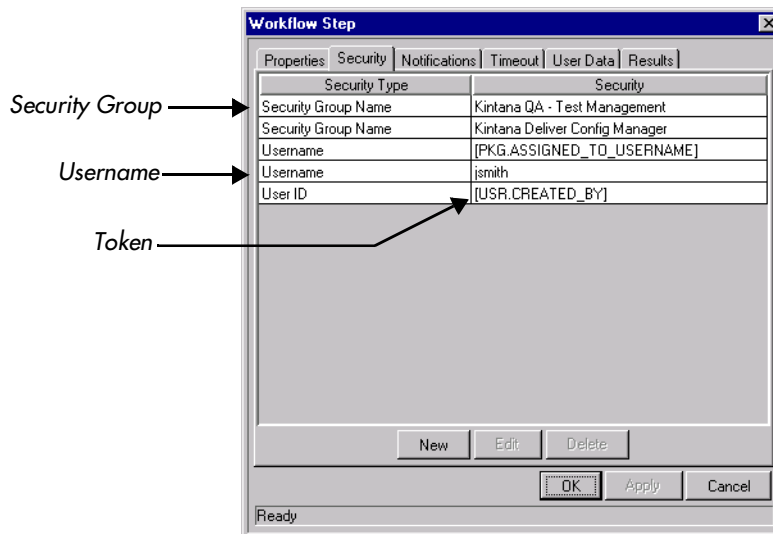
Field on Workflow Step Window	Description
SOURCE ENVIRONMENT	Specifies the Source Environment where the software that is to be changed is located.
SOURCE ENVIRONMENT GROUP	Specifies the Source Environment Group which contains the Environment from which the software change is obtained. The Source Environment Group can also be used in conjunction with the Environment Application Codes to provide a dynamic Source Environment selection.
DEST ENVIRONMENT	Specifies the Destination Environment to which the software change is deployed.
DEST ENVIRONMENT GROUP	Specifies the destination Environment Group. The destination consists of multiple Kintana Environments to which the software change is deployed.
AVG LEAD TIME	A user-specified metric for comparing actual performance to estimated goals. It does not affect any transactional logic.
PARENT ASSIGNED TO USER	If this field is not empty when the step becomes Eligible, the Assigned to User of the Package automatically changes to the user specified in the field.
PARENT ASSIGNED TO GROUP	If this field is not empty when the step becomes Eligible, the Assigned to Group of the Package automatically changes to the Security Group specified in the field.
WORKFLOW STEP INFORMATION	A text entry field in which any URL can be entered. This is where users can find documents, instructions or comments to aid them in working the Workflow Step.

Specify the Security

“Integrating Participants into Your Deployment System” on page 195 provides information on setting up security for your deployment process. This includes such things as controlling who can create Packages and who can act on specific steps in the process. Security related directly to processing a Workflow step is configured in the WORKFLOW STEP window.

You can define who can act on the step by:

- Security Group
- Username
- Token (standard or user-defined)



Kintana recommends using Security Groups or dynamic access (Tokens) when defining the Workflow step security. You should avoid specifying a list of users to control an action; for example, specifying a list of users who can act on a Workflow step. If the list of users changes (due to an organizational reorganization), you would have to update that list in many places on the workflow. By using a Security Group instead of a list of users, you can update the Security Group once, and the changes are propagated throughout the Workflow steps.

Configure Notifications for the Workflow Step

[“Setting Up Communication Paths”](#) on page 215 provides information on setting up notifications for steps in your deployment process. This includes such things as configuring the notification’s recipients and message.

See the following sections for more details:

- [“Identify Participants and Security”](#) on page 71
- [“Setting Up Communication Paths”](#) on page 215

Adding a Subworkflow

A Subworkflow can be selected from the WORKFLOW STEP SOURCES window and dragged onto the LAYOUT tab. When the Package, Request, or Release Distribution reaches the Subworkflow Step, it follows the path defined in that Subworkflow. The Subworkflow will either close within that Workflow or return to the parent Workflow.

To add an enabled Subworkflow to another Workflow:

1. Select the desired Subworkflow and drag it to the LAYOUT tab. The WORKFLOW STEP window opens.

The screenshot shows the 'Workflow Step' configuration window. The 'Properties' tab is selected. The 'Step Name' is 'Review and Test Changes'. The 'Source Type' is 'Workflow'. The 'Enabled' radio button is set to 'Yes'. The 'Display' dropdown is set to 'Always'. The 'Workflow Parameter' dropdown is empty. The 'Source Environment' and 'Dest Environment' radio buttons are selected. The 'Avg Lead Time' and 'Project Status' fields are empty. The 'Parent Assigned To User' and 'Parent Assigned To Group' fields are empty. The 'Workflow Step Information' field is empty. The 'OK', 'Apply', and 'Cancel' buttons are visible at the bottom.

This window contains preconfigured information which is specific to the selected Workflow Step.

2. Configure this step as you would configure an execution or decision step.
3. Click **OK**.

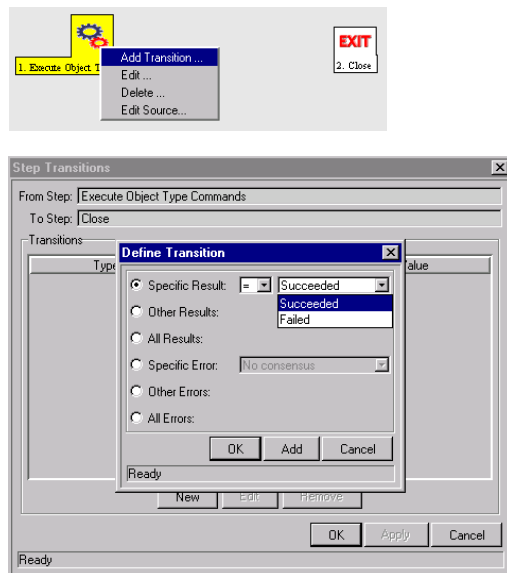
See *“Advanced Workflow Topics”* on page 257 for a detailed discussion of using Subworkflows in Kintana.

Adding Transitions Between Steps

After adding the steps to the Workflow **LAYOUT** tab, you need to configure the transitions between them. You can choose to transition between steps based on the following step results:

- Specific Result (based on the Validation configured in the step source)
- Other Results
- All Results
- Specific Error
- Other Errors
- All Errors

Select the proper transition between steps. The following section provides some example scenarios and transition configuration options:

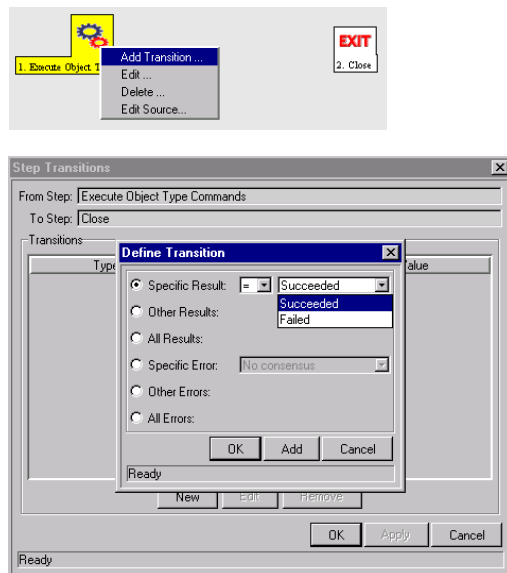


Transition based on a specific result

Transitioning based on the result of a specific decision or execution is the most basic transition method in Kintana. It allows you to branch your business process based on anticipated results of a step in the Workflow.

To transition based on a specific Workflow step result:

1. Add transition between two steps. The DEFINE TRANSITION window opens.
2. Select the SPECIFIC RESULT radio button.
3. Select the desired result from the drop down list. The values in this list will vary depending on the Validation set in the Workflow step source for the transitioning step.



Transition based on a value in a field

You can transition a Package based on the value in a particular field. This can be a general field in the Package definition (Priority, Assigned User, Package Group, etc.) or a custom field specified in the Package Line (defined on the Object Type). For example, if the Package's PRIORITY field is set to **CRITICAL**, then you may want the Package to follow a different, more robust process. This is done by resolving a Kintana field Token in a Workflow execution step. The Workflow engine evaluates the field's value at a specific step and then can route the Package Line accordingly.

To transition based on the value in a field

1. Add an immediate execution step source to the Workflow. You may have to create a custom Workflow step source for this operation. The step source should be configured as follows:

Field in Execution Window	Value
WORKFLOW SCOPE	PACKAGES
EXECUTION TYPE	Token
PROCESSING TYPE	IMMEDIATE
VALIDATION	Select or create a validation that includes all of the possible values of the resolved Token. For example, if you plan on branching based on the Priority field, use the [PKG.PRIORITY] token and the DLV - PACKAGE PRIORITY - ENABLED validation. The validation contains all possible values of the token.
EXECUTION	Enter the Token for the value that you would like to transition based on.
ENABLED	YES

2. Add transition between two steps. The DEFINE TRANSITION window opens.
3. Select the SPECIFIC RESULT radio button.
4. Select the desired result from the drop down list. The values in this list will vary depending on the Validation set in the Workflow step source for the transitioning step. For the above Priority example, the possible values will be the values allowed in the Package's PRIORITY field.

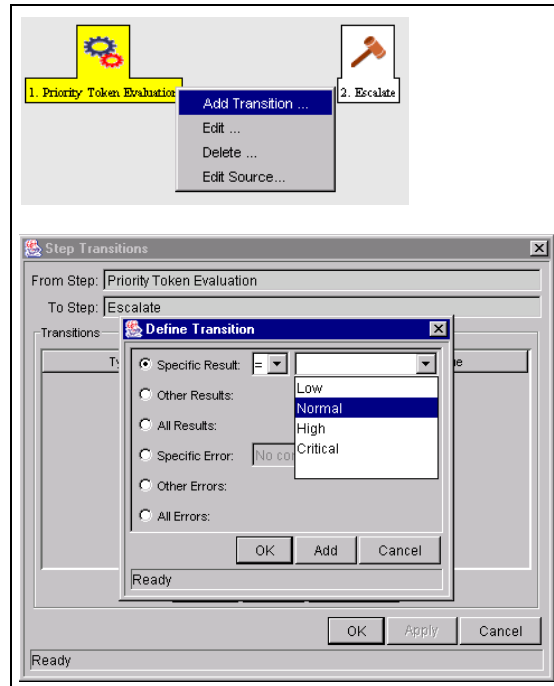


Figure 6-6 Example: Transitioning based on value in a field (Token)

Transition based on data in a table

You can transition based on information stored in a table. To transition using this method, you must use a Workflow execution step with an execution type of SQL. See [“Execute a SQL statement and then transition based on the result”](#) on page 104 for more information.

When transitioning from a properly configured execution step (EXECUTION TYPE = **SQL STATEMENT**), you will transition based on a Specific Result. The possible results are defined in the Workflow step source’s Validation. The values in this list are determined by a SQL query of a database table.

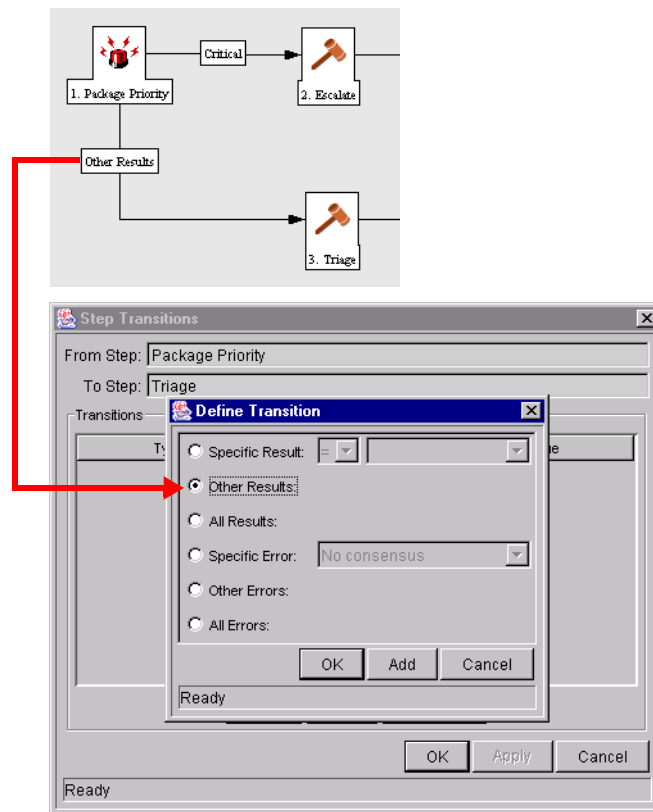
As with any execution step, you can configure this transition to be an immediate or manual step.

Transition based on all but one specific value

You can transition based on all but one specified value. For example, you want to transitional all “Critical” Packages one way and all other results another. To configure this:

1. Create a transition from a step based on a specific result.

2. Create another transition from the same step to another step and specify **OTHER RESULTS**.



In the above example, only Packages with a “Critical” priority will follow the ESCALATE path. All other results are sent to TRIAGE.

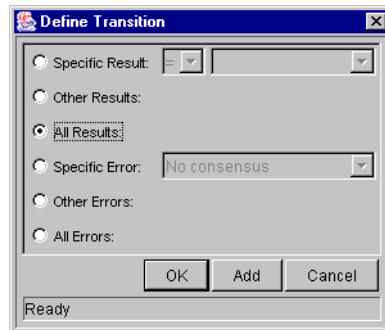


You can use **OTHER RESULTS** when multiple transitions are exiting a single step. **OTHER RESULTS** will act as the transition if none of the other explicit transition conditions are satisfied.

Transition based on all results

You can define a Package to transition regardless of the step’s actual results. For example, you want to run a subworkflow to perform server maintenance after the on-call server contact is identified. To do this, add a transition from the **SPECIFY CONTACT** step to the subworkflow. Because the next step in the process doesn’t depend on the result of the step, it is appropriate to use the **ALL RESULTS** transition.

To do this, define a transition from the step and select **ALL RESULTS**. The DEFINE TRANSITIONS window is shown below.



Transition based on error

You can transition based on a specific error that occurs during an execution step. This allows you to branch your business process based on likely execution errors such as **TIMEOUT**, **COMMAND EXECUTION** or **INVALID TOKEN**.

To transition based on a specific Workflow step error:

1. Add transition between two steps. The DEFINE TRANSITION window opens.
2. Select the SPECIFIC ERROR radio button.
3. Select the error from the drop down list. All values in this list are defined in [Table 6-6](#).

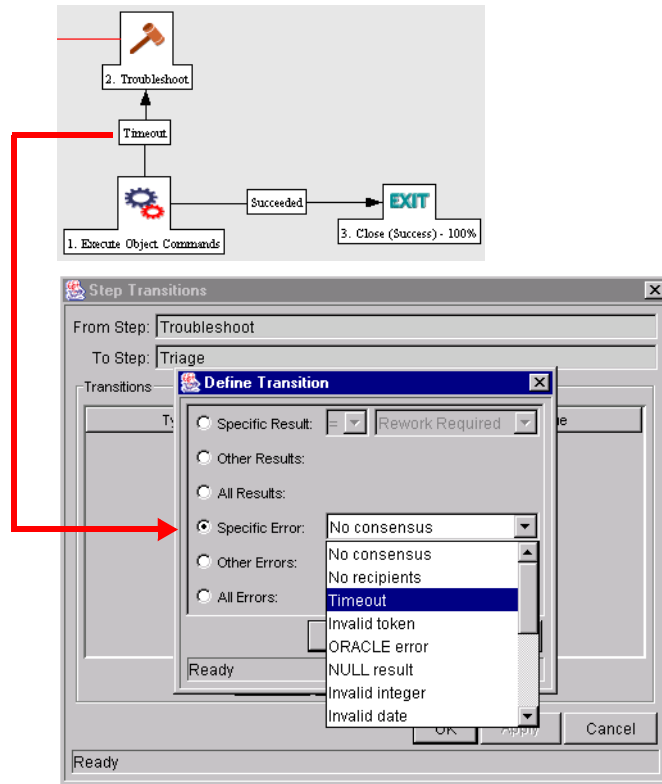


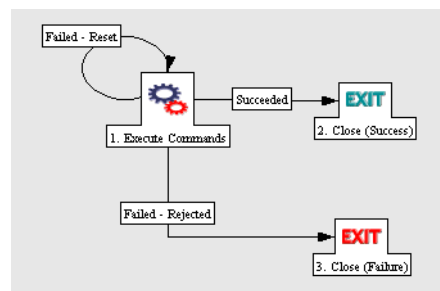
Table 6-6. Specific Error Step Transitions

Transition	Transition Option(s)	Meaning
SPECIFIC ERROR	MULTIPLE RETURN RESULTS	When the Package Level subworkflow receives multiple results from Package Lines that traversed through it.
	NO CONSENSUS	When all users of all Security Groups, or users linked to the Workflow Step need to vote, and there is no consensus.
	NO RECIPIENTS	When none of the Security Groups linked to the Workflow Step has users linked to it, no user can act on the Workflow Step.
	TIMEOUT	When the Workflow Step times out. Used for Executions and Decisions.
	INVALID TOKEN	Invalid Token used in the execution.
	ORACLE ERROR	Failed PL/SQL Execution.
	NULL RESULT	No result is returned from the execution.
	INVALID INTEGER	Validation includes an invalid value in the Integer field.
	INVALID DATE	Validation includes an invalid value in the Date field.
	COMMAND EXECUTION ERROR	Execution engine has failed or has a problem.
	INVALID RESULT	Execution or Subworkflow has returned a result not included in the Validation.
	PARENT CLOSED	For wf_receive or wf_jump steps, a Package Line is expecting a message from a Request that is cancelled or closed.
	CHILD CLOSED	For wf_receive or wf_jump steps, a Request is expecting a message from a Package Line that is cancelled or closed.
NO PARENT	For wf_receive or wf_jump steps, a Package Line is expecting a message from a Request that has been deleted.	
NO CHILD	For wf_receive or wf_jump steps, a Request is expecting a message from a Package Line that has been deleted.	
MULTIPLE JUMP RESULTS	For wf_jump steps in a Package Line, different result values were used to transition to the step.	
OTHER ERRORS	NA	All errors for which transitions have not been defined.
ALL ERRORS	NA	Any kind of error.

Transition back to the same step

You may want to retain the option of resetting failed execution steps, rather than immediately transitioning along a “failed” path. This is often helpful when troubleshooting the execution. To configure this:

1. Create an execution step source to execute the Workflow or Object Type commands.
2. Create a Validation with the following Validation Values.
 - a. SUCCEEDED
 - b. FAILED
 - c. FAILED - RESET
 - d. FAILED - REJECTED
3. Add the step to the Workflow **LAYOUT** tab.
4. Add transitions based on the following Specific Results:
 - a. SUCCEEDED
 - b. FAILED - RESET -- set the transition to return back into the same step.
 - c. FAILED - REJECTED



When the commands execute successfully, they will follow the Success transition path. However, when the commands fail, they will not transition out of the step because no transition has been defined for the Failed result. The user has to manually select the Package Line step and select Failed - Retry. The execution will re-run.



Be careful when using an immediate execution step that the “Failed” result isn’t feeding directly back into the execution step. This would result in a continual execution-failure loop.

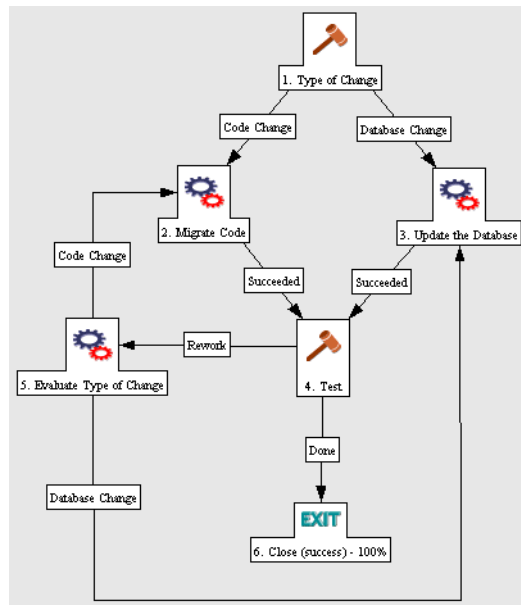
Transition based on a previous workflow step result (parameters)

You can use Workflow parameters to store the result of a workflow step. This value can then be used later to define a transition. To configure this, you need to:

1. Create a WORKFLOW PARAMETER in the Workflow window.
2. Specify that WORKFLOW PARAMETER in a Workflow step on the **LAYOUT** tab.
3. Create a token execution step that will resolve the value in the Workflow parameter.

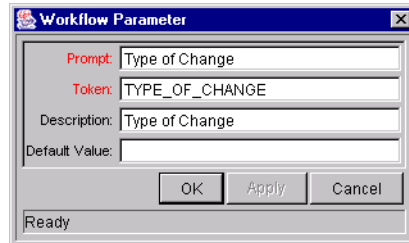
Example: Using a Workflow Parameter to Transition

One step in the process requires the user to route the Package based on the type of change (code or database). The decision made at this step is then considered later in the process to correctly route rework of the specific type.

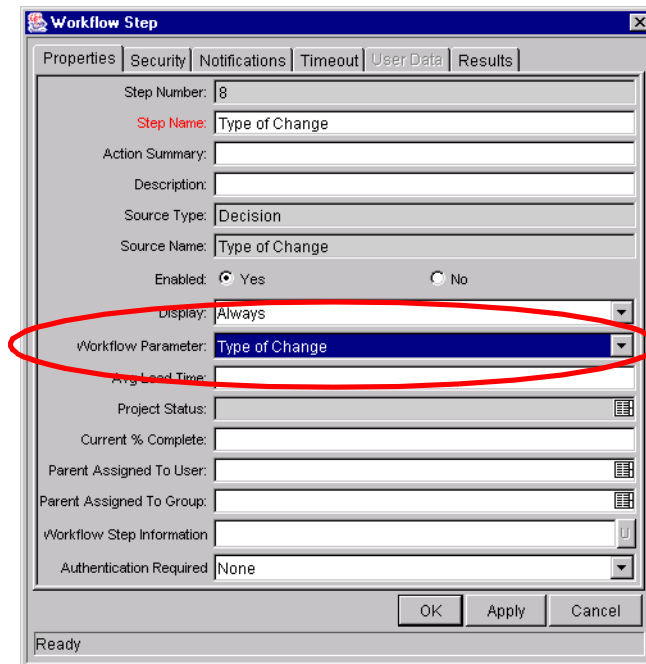


To enable this process, set the following in the Workflow:

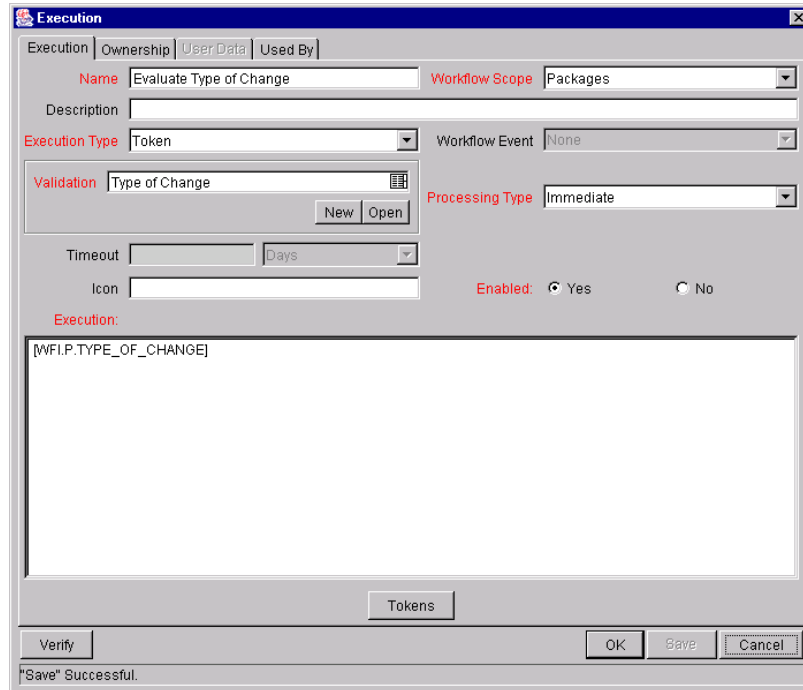
1. Create a WORKFLOW PARAMETER in the Workflow window. This is done by clicking **ADD** in the **WORKFLOW** tab on the WORKFLOW window.



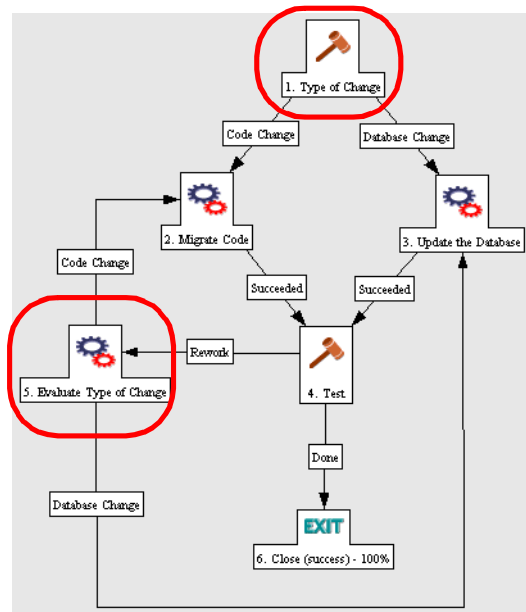
2. Select **TYPE OF CHANGE** for the WORKFLOW PARAMETER in TYPE OF CHANGE Workflow step on the **LAYOUT** tab.



3. Create a token execution step that will resolve the value in the Workflow parameter. Note that the Validation used in this step should contain the same values as the Validation specified in the TYPE OF CHANGE decision step.



4. Add the steps and transitions as shown below.



Transition to and from Subworkflows

There are special configuration requirements when transitioning to and from Subworkflows. See detailed instructions in the [“Using Subworkflows”](#) on page 257.

Transition to and from a Request Workflow

There are special configuration requirements when transitioning to and from a Request Workflow using Jump and Receive steps in the Workflows. See detailed instructions in [“Using Subworkflows”](#) on page 257.

Chapter 7

Constructing the Object Type

This chapter provides an overview for how to configure the Kintana Object Types that will be used to process objects (Package Lines) through your deployment Workflow. This includes configuring Object Type fields and Commands. It illustrates how the information gathered in “*Gathering Process Requirements and Specifications*” on page 49 can be used to construct the fields and commands required to correctly process your business objects.

This chapter discusses the following topics:

- *Creating an Object Type - Overview*
- *Creating Object Type Fields*
- *Creating Object Type Commands*

Using the information gathered in “*Gathering Process Requirements and Specifications*” on page 49 you can begin to configure your Object Type. This information is related to settings specified in the Object Type worksheets shown in *Figure 7-1*.

Table D-5. Object Type Information.

	Value
Object Type Name	
Description	

Table 13-18. Object Type Commands

Goal of Commands	
Command Steps	
Conditions (When to execute)	

#	Field Names	Description
1		
2		
3		
4		
5		
6		
7		
8		
9		
10		
11		
12		
13		
14		
15		
16		
17		
18		

Figure 7-1 Object Type Worksheets

Creating an Object Type - Overview

Object Types are created and configured using the Kintana Workbench. To create a new Object Type:

1. Click the **DELIVER** screen group on the Workbench and click the **OBJECT TYPES** icon. The OBJECT TYPE WORKBENCH window opens.
2. Click **NEW OBJECT TYPE**. The OBJECT TYPE window opens.
3. Enter the Object Type general information. This includes the Object Type's NAME, DESCRIPTION, ACCELERATOR association, and OBJECT CATEGORY.
4. Create fields that describe your Object. See "[Creating Object Type Fields](#)" on page 137. This includes configuring the following:

- o Field names
 - o Validations and component types (dictated by the validation)
 - o Field Behaviors: whether fields are displayed, required, any defaulting behavior, etc.
5. Configure the Fields' layout. See [“Modifying the Object Type Layout”](#) on page 150.
 6. Create the Object Type's commands. See [“Creating Object Type Commands”](#) on page 156.
 7. Set Ownership for the Object Type. This controls who can modify or delete the Object Type. See [“Kintana Security Model”](#) for details.



Tip

It is often more efficient to use the **COPY** functionality to copy an existing Object Type and then edit the new copy. To reduce the amount of editing required choose an existing Object Type similar to the Object Type to be generated.



Note

Only Kintana users with the appropriate security can create or edit Object Types. To edit Object Types, you must belong to a Security Group that has the access grant DELIVER: EDIT OBJECT TYPES. See the [“Setting Configuration Security”](#) on page 210 for more information.

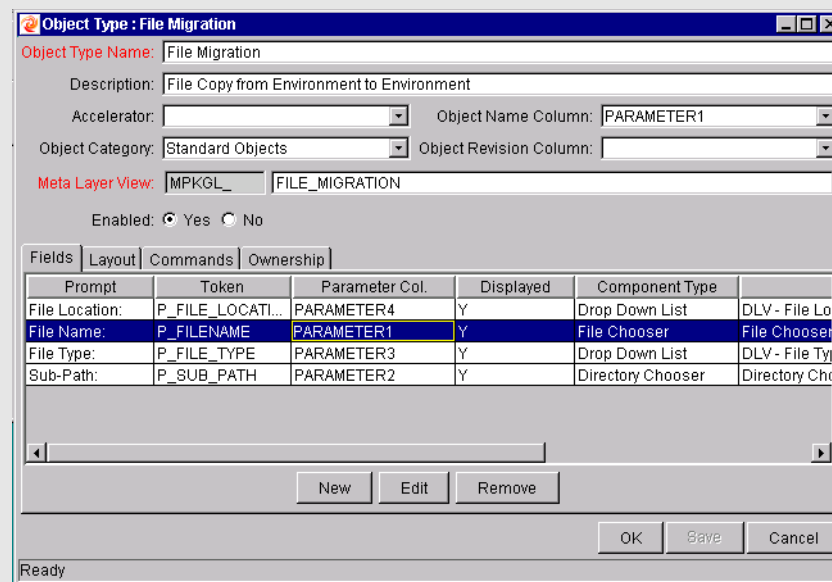
Creating Object Type Fields

Object Type fields define the information collected from the end users when a Package Line is generated. You can configure fields prompts, tokens, behaviors and validations for each Object Type.

Object Type fields are critical for Kintana deployments. They are often used by the Workflow for routing (Validations). Tokens associated with the field are also often referenced in the Object Type commands, Workflow step security, and notification settings.



To migrate a file, Kintana Deliver must know the file name. A field named “File Name” can be created to capture that information.



The general steps for configuring an Object Type field are:

1. Open the OBJECT TYPE window.
2. Click **NEW**. The FIELD window opens.
3. Enter the general field information: FIELD PROMPT, TOKEN, and DESCRIPTION.
4. Select a Validation for the field. If a Validation doesn't exist that meets your needs, you can create one. The Validation dictates the possible values that can be entered in the field. They also dictate the field type (text field, drop down list, date field, etc.).
5. Configure the field's behavior. This consists of setting options in the field's **ATTRIBUTES**, **DEFAULT** and **DEPENDENCIES** tabs. See “*Configuring Field Behavior*” on page 144. Note that some field behavior is dependent on other Object Type fields. You may have to revisit this step after creating the other fields in your Object Type.
6. Enable the field.



Example

ACME requires a File Type field to describe the objects to be deployed. On their Object Type, they add a field with the following settings:

The Validation is validated by a list. This is an appropriate choice because the selection is not expected to change.

Determining the Field Type (Selecting a Validation)

When configuring your Object Type, you can specify a different Validation for each field. The Validation dictates the possible values that can be entered in the field. It also dictates the field type (text field, drop down list, date field, etc.). The following sections provide some general information related to Validations.

- [“Available Field Types”](#) on page 139
- [“Selecting the Validation”](#) on page 141
- [“Building a Validation”](#) on page 142

See [“Validations”](#) on page 287 for more detailed implementation instructions.

Available Field Types

Fields located on the Object Type can be of the following types.

Table 7-1. Field types.

Field component	Description
Text Field, Text Area	Text fields and text areas are generic entry fields. Text fields are displayed on a single line, while text areas are displayed on multiple lines using a scroll bar if necessary. The values that are entered can be constrained. If an attempt is made to type non-conforming values into a text field or text area, the entries are ignored. For example, if the letter "A" is typed into a numeric field, the character does not appear.
Drop down list	Field that allows the user to select from a predefined set of values. The values in a drop down list can be specified in two ways: <ul style="list-style-type: none"> • In the VALIDATED BY field, by selecting LIST to enter specific values. • By selecting SQL to use a SQL statement to build the contents of the list.
Auto-complete list	Field that allows the user to select from a predefined set of values. The values in an auto-complete list can be specified in the following ways. In the VALIDATE BY field, select one of the following: <ul style="list-style-type: none"> • LIST: used to enter specific values. • SQL: uses a SQL statement to build the contents of the list. • COMMAND WITH DELIMITED OUTPUT: uses a system command to produce a character-delimited text string and uses the results to define the list. • COMMAND WITH FIXED WIDTH OUTPUT: uses a system command to produce a text file and parses the result on the basis of the width of columns, as well as the headers.
Radio Button (Yes/No)	Radio buttons are used for fields where there are two or more choices. Selecting on option disables the other associated options. For example, clicking Yes in a Yes/No radio button pair disables the No option. To select a choice, click the button to the left of the appropriate choice.
Date Field	Date fields can accept a variety of formats. The current date field Validations are separated into two categories: all systems, and systems using only the English language.
Web Address (URL)	The Web Address field is a generic text entry field in which any URL can be entered. When this field is used, a U button appears next to the field. If U is clicked, a Web page is opened using the field value as the Web address.

Table 7-1. Field types.

Field component	Description
Directory Chooser	<p>The Directory Chooser field can be used to select a valid directory from an Environment. Kintana Deliver connects to the first Source Environment on a Workflow and allows navigation through the directory structure and the selection a directory from the list.</p> <p>On every Object Type that a Directory Chooser is used, it is also necessary to have a field whose token is 'P_FILE_LOCATION' and whose validation is KINTANA DELIVER - FILE LOCATION. The possible values for this field are CLIENT and SERVER. If CLIENT is chosen, the Directory Chooser connects to the Client Base Path of the Source Environment. If SERVER is chosen, the Directory Chooser connects to the Server Base Path of the Source Environment.</p>
File Chooser	<p>A File Chooser field can be used to select a valid file from an Environment. Kintana connects to the first Source Environment on a Workflow and provides the ability to view all files within a specific directory and select one from the list.</p> <p>On every Object Type that a File Chooser is chosen, it is necessary to have two other fields defined. The first is a field for the File Location for the directory chooser, described in the previous section. The second is a field whose token is 'P_SUB_PATH'. This field is the directory from which the file is selected and is usually a Directory Chooser field.</p>
Password	<p>The Password Field component type creates a text field with an associated C button. Data is entered through a dialog that asks for the new password and a verification of the password. The text is then displayed in the field as *****.</p>

Note: Kintana Create and Kintana Drive also support the following additional field component types:

- Table Component
- Budget
- Staffing Profile
- Resource Pool

See “*Validations*” on page 287 for details.

Selecting the Validation

Use the information gathered in “*Gathering Process Requirements and Specifications*” on page 49 to determine the appropriate Validation for the for the Object Type field. If a Validation does not exist that meets your

requirements (has the appropriate values) you can create one. See [“Validations”](#) on page 287 for a complete list of Validations that are delivered during a Kintana installation.

You can also select a Validation that has been configured for use at your site.



Be careful when using a Validation that has been configured for use in another process. If the owner of the other process changes the Validation, it will also be changed for the items in your process. Consider creating a new Validation by copying the existing one. You can then control who can alter the Validation values by setting Ownership on that Validation.

Building a Validation

If a Validation does not exist that meets your requirements (has the appropriate values) you can create a new one. Click **NEW** in the FIELD window to open the Validation window. Define your Validation using the instructions in [“Validations”](#) on page 287.

This section provides some guidance for when to use specific types of validations.

Table 7-2. When to use certain field component types.

Component Type	When to use:
Auto-complete list Drop down list	Use when presenting a list of options to the user. For example: <ul style="list-style-type: none"> • List of all users • List of all users in a specific security group (include on a package line to specify who should review a change) • Desired actions (copy only; copy and run commands; copy, run commands and delete; etc.) • List of information located in another (non-Kintana) system. (example: list of managers stored in PeopleSoft)
Directory chooser File chooser	Used to specify the location of objects to be deployed.

Table 7-2. When to use certain field component types.

Component Type	When to use:
Text field Text area Date field	Used to capture related information required for processing.
Radio button	Used when only two options exist. (Yes/No)

Auto-Complete Versus Drop Down List

Auto-completes and drop down lists are often applied in similar situations. They both present a predefined / limited list of choices to the user, but both have unique features which could be more appropriate for a given situation. Consider the following comparison chart when selecting to use a drop-down or auto-complete list.

Table 7-3. Auto-complete versus drop down comparison chart.

Action	Auto-complete	Drop-down
Can contain a static list of choices	Yes	Yes
Can contain choices derived from a database query	Yes	Yes
Can contain choices from system executions (commands)	Yes	
Can display multiple columns of information	Yes	
Allows users to select multiple values from the list	Yes	
List is determined at the time of page/screen load		Yes
List is determined when the field is selected. this is useful when making the values in the list dependent on other parameters in the screen. For example, listing only users in a specified security group.	Yes	
Allows for partial value returns (for example, type "A" and view only the choices beginning with "A.")	Yes	

A final consideration is usability. Drop down lists become less efficient when the selection list gets large. Consider using auto-completes in these situations.

Tips for Configuring Validations

Consider the following tips when creating a Validation for your Object Type:

- Be careful when creating Validations (drop down lists and auto-complete lists) that are validated by lists. Each time the set of values changes, you will be forced to update the Validation. Consider, instead, validating using a SQL query or PL/SQL function. For example, to create an auto-complete field that lists all Kintana users in a specific department, validate the list by SQL.

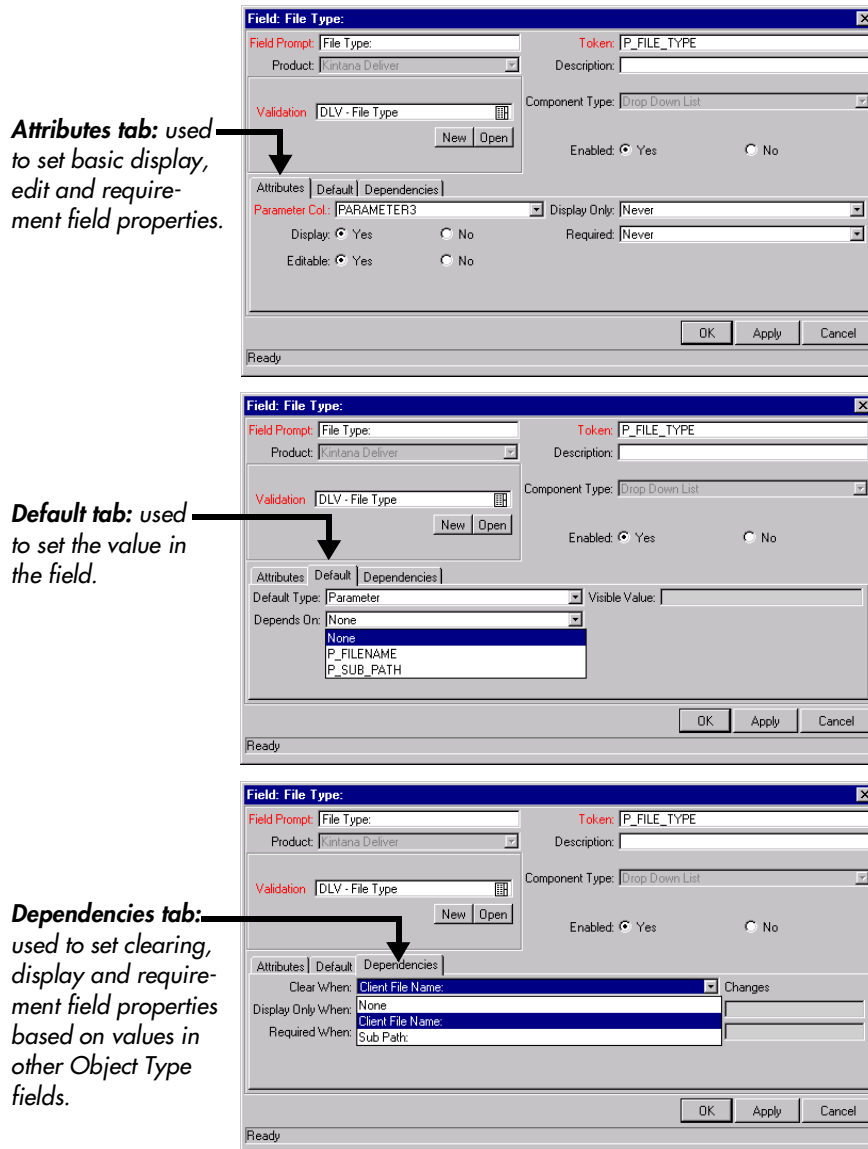
```
SELECT U.USER_ID, U.USERNAME, U.FIRST_NAME, U.LAST_NAME
FROM KNTA_USERS U, KNTA_SECURITY_GROUPS SG,
     KNTA_USER_SECURITY US
WHERE SG.SECURITY_GROUP_ID = US.SECURITY_GROUP_ID AND
      US.USER_ID = U.USER_ID
AND SG.SECURITY_GROUP_NAME = 'Support Team'
and UPPER(u.username) like UPPER('?%')
or u.username like lower(substr('?',1,1)) || '% '
order by 2
```

In the above example, when a new user is added to Kintana and included in the “Support Team” Security Group, that user will automatically be included in the auto-complete list.

- Reuse SQL and PL/SQL from existing Validations. Review Kintana’s seeded Validations (see “*System Validations*” on page 331) to see if there are other similar Validations in the system. If there are, copy the validation and modify the VALIDATED BY specifications to meet your requirements.
- You can often use the same Validations for Workflow step sources as you do for the field validations.

Configuring Field Behavior

You can configure each field to behave in a certain way using the FIELD configuration window in the OBJECT TYPE window. The FIELD window contains three tabs: **ATTRIBUTES**, **DEFAULT**, and **DEPENDENCIES**.



From the FIELD window you can configure:

- Whether the field is displayed (for example: you may need to store a value for later use in commands, but don't want to clutter the Package Line)
- Whether a field can be edited under different circumstances
- Whether the field is required under different circumstances
- Whether the field defaults to a certain value
- Dependencies to values in other fields in the Object Type

- o Clear the field's value when another field changes
- o Display only when another field has a specific value
- o Required only when another field has a specific value

Table 7-4 defines the behavior-related settings in the FIELD window.

Table 7-4. Attributes Tab - Fields window

Field	Description
PARAMETER COL.	<p>Determines the internal database column that the field value will be stored in. These values are then stored in the corresponding column in the Package Lines table for each Line of the given Object Type.</p> <p>Information can be stored in up to 30 columns and thus allow up to 30 fields/Parameters. No two fields in an Object Type can use the same column.</p>
DISPLAY ONLY	<p>Determines whether a field should be displayed using the following options: ALWAYS, NEVER or USE DEPENDENCY RULES. Select USE DEPENDENCY RULES to use the logic defined in the DEPENDENCIES tab.</p> <p>DISPLAY ONLY: ALWAYS means that the field is not editable. DISPLAY ONLY: NEVER means that the field is always editable.</p>
DISPLAY	Determines if this field is visible in the Package Line region of the PACKAGE window.
REQUIRED	Determines if a value needs to be specified for this field using the following options: Always , Never or Use Dependency Rules . Select Use Dependency Rules to use the logic defined in the Dependencies tab.
UPDATEABLE	After a Package Line has been entered and submitted, it starts moving through its Workflow. This attribute determines if the field can still be updated. For example, it may be necessary to ensure that a FILENAME field is not updateable once Package Lines of FILE Object Type start getting processed.

Table 7-5. Default tab - Fields window

Field	Description
Default Type	Defines if the field will have a default value. Either default the field with a constant value, default it from the value in another field, or default to a parameter.
Visible Value	If a DEFAULT TYPE of CONSTANT is selected, the constant value can be entered here. This value should be what the user would normally enter in the field.
Depends On	If defaulting from another field, enter the token name of that field. At runtime, when using this Object Type, every time a value is entered or updated in the source field, it will automatically be entered or updated in this destination field.

Table 7-6. Dependencies tab - Fields window

Field	Description
Clear When ___ Changes	Indicates that the current field should be cleared when the specified field changes.
Display Only When	Indicates that the current field should only be editable when certain logical criteria are satisfied. This field functions with two adjacent fields: a drop down list containing the logical qualifier and a text field. To use this functionality, select USE DEPENDENCY RULES from the first drop down list.
Required When	Indicates that the current field should be required when certain logical criteria are satisfied. This field functions with two adjacent fields: a drop down list containing logical qualifier and a text field. To use this functionality, select USE DEPENDENCY RULES from the first drop down list.


 Note

Because field behavior is often dependent on other fields in the Object Type, you often have to create the other Object Type fields before configuring a field's behavior.

Configuring Field Dependencies

Field behavior and properties can be linked to the value of other fields defined for that entity. For example, an Object Type field can become required when the value in another field in that Object Type equals the text “CRITICAL.”

A field can be configured to:

- Clear when another field changes.
- Become read only when another field meets a logical condition defined in [Table 7-7](#).
- Become required when another field meets a logical condition defined in [Table 7-7](#).

Table 7-7. Field Dependency logical qualifiers

Logical qualifier	Definition
like	A “like” condition looks for the specified value to find any matching values in the chosen field.
not like	A “not like” looks for values in the chosen field that are not equal to the specified value.
is equal to	An “is equal to” looks for an exact match of the specified Value to the contents of the field chosen.
is not equal to	An “is not equal to” is true when there are no results exactly matching the value of the field contents.
is null	An “Is null” is true when the field selected is blank.
is not null	An “Is not null” is true when the field selected is not blank.
is greater than	An “Is greater than” looks for a numerical value in excess of the value entered in the Value field.
is less than	An “Is less than” looks for a numerical value below the value entered in the Value field.
is less than equal to	An “Is less than equal to” looks for a numerical value below, or the same as, the value entered in the Value field.
is greater than equal to	An “Is greater than equal to” looks for a numerical value in excess of, or the same as, the value entered in the Value field.

To configure a field dependency:

1. Click the **DEPENDENCIES** tab in the **FIELD** window.

The screenshot shows the 'Field: File Type' configuration window. The 'Dependencies' tab is active. The 'Clear When' dropdown is set to 'Client File Name', 'Display Only When' is set to 'None', and 'Required When' is set to 'Client File Name'. The 'Enabled' radio buttons are set to 'Yes'. The 'Field Prompt' is 'File Type:', 'Token' is 'P_FILE_TYPE', 'Product' is 'Kintana Deliver', and 'Component Type' is 'Drop Down List'.

2. Set the field dependencies. Use one of the following options:
 - a. Select a field name from the **CLEAR WHEN** drop down list to indicate that the current field should be cleared when the selected field changes.
 - b. Select a field name from the **DISPLAY ONLY WHEN** drop down list to indicate that the current field should not be editable when certain logical criteria are satisfied. This field functions with two adjacent fields. These fields are a drop down list containing logical qualifier, and a field which dynamically changes to a date field, drop down list, or text field, depending on the selected field's validation.
 - c. Select a field name from the **REQUIRED WHEN** drop down list to indicate that the current field should be required when certain logical criteria are satisfied. This field functions with two adjacent fields. These fields are a drop down list containing logical qualifier, and a field which dynamically changes to a date field, drop down list, or text field, depending on the selected field's validation.
3. Click **OK**.

This adds the field dependencies to the **FIELDS** tab of the **OBJECT TYPE** window and closes the **FIELD: NEW** window.

Using Commands to Change Field Values

Kintana Commands can also be used to control certain behavior of Object Type fields. At specific points (Workflow execution steps) in your deployment process, you can select to run the commands stored in the Object Type. These commands can then manipulate the data inside an Object Type field. For example, you can construct a Command to consider a number of parameters and then default a field based on those parameters. This provides an advantage over the defaulting features in the FIELD window, which can only default based on a single field located on the same Object Type.

Kintana provides a system Special Command that can perform this function: `ksc_store`. See "*Using Commands and Tokens*" for information on using this and other commands in Kintana.

Controlling field values using Commands can be useful in the following situations:

- Storing a value from an execution into a custom field
- Clearing a field after evaluating a number of parameters

Modifying the Object Type Layout

The Object Type layout can be modified by:

- *Changing a Column Width*
- *Moving Fields*

Changing a Column Width

To change the column width of an Object Type field:

1. Open the OBJECT TYPE window.
2. Click the LAYOUT tab.
3. Select the field.
4. Select either **1** or **2** from the FIELD WIDTH drop down list.

The Layout editor will not allow changes to be made if the change conflicts with another field in the layout (for example, a field's width cannot be changed from one to two if another field exists in column two on the same row).

Additionally, for fields of component type Text Area, the number of lines the text area displays can be determined by clicking the Text Area type field and changing the value in the COMPONENT LINES attribute. If the selected field is not of type **TEXT AREA**, this attribute will be blank and non-updatable.

Moving Fields

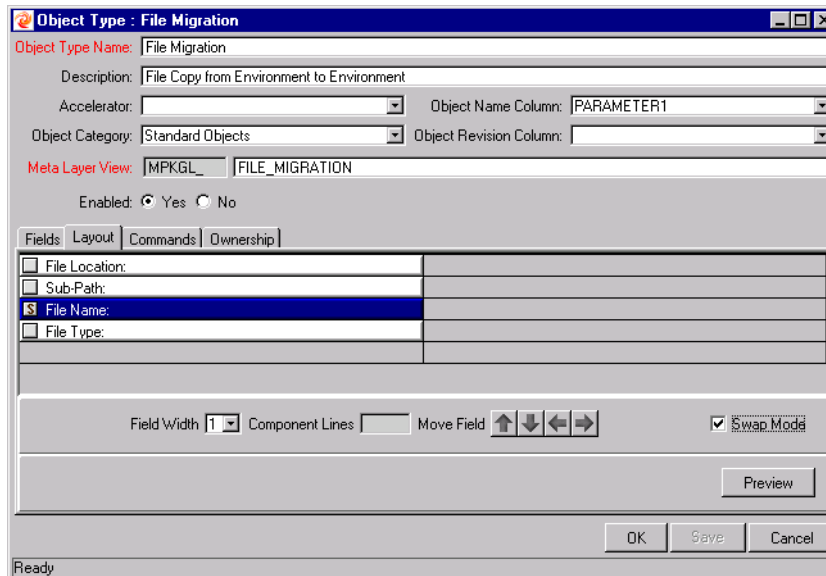
To move an Object Type field or a set of fields:

1. Open the OBJECT TYPE window.
2. Click the **LAYOUT** tab.
3. Select the field(s). To select more than one field, use the Shift key while selecting a range. It is only possible to select a continuous set of fields (i.e. the Ctrl-Select functionality is not supported).
4. Use the arrow buttons to move the fields to the desired location in the layout builder.



Note

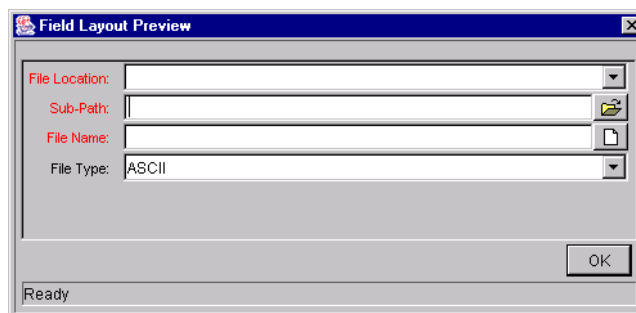
A field or a set of fields cannot be moved to an area where other fields already exist. The other field(s) must be moved out of the way first.



5. To switch the positions of two fields, select the first field and check the SWAP MODE check box. An “S” appears in the check box area of the selected

field. Then double-click the second field that you want to switch positions with the first. This causes the two fields to change positions. Following the switch, the **SWAP MODE** check box is turned off. To swap another set of fields, repeat this procedure.

6. To check what the layout looks like in actual use, click **PREVIEW**. This opens a small window that shows the fields as they will appear. It is important to note that:
 - If all the fields have a width of one column, all displayed columns will automatically span the entire available area when a Package Line of the given Object Type is viewed or generated.
 - Any rows with no fields are ignored. They do not show up as a blank line.
 - Any non-displayed fields do not affect the layout. They are considered the same as a blank field.



Setting the Object Name

When defining an Object Type, it is important to choose one field to represent the name of this object in a Package Line. This field is the object name. To designate a field as the object name, select that field's **PARAMETER COLUMN** in the **OBJECT NAME COLUMN** drop-down list.

For example, to designate "FILE NAME" as the object name field for a "FILE MIGRATION" Object Type, select the "FILE NAME" field's Parameter Column in the **OBJECT NAME COLUMN** drop-down list.

Object Type Name: File Migration

Description:

Accelerator: Object Name Column: **PARAMETER1**

Object Category: Standard Objects Object Revision Column:

Meta Layer View: MPKGL_ FILEMIGRATION_00

Enabled: Yes No

Prompt	Token	Parameter Col.	Displayed	Component Type	Validation	Requ
File Location:	P_FILE_LOCAT...	PARAMETER4	Y	Drop Down List	DLV - File Location	Y
File Name:	P_FILENAME	PARAMETER1	Y	Text Field	Text Field - 40	Y
File Type:	P_FILE_TYPE	PARAMETER3	Y	Drop Down List	DLV - File Type	Y
Sub-Path:	P_SUB_PATH	PARAMETER2	Y	Text Field	Text Field - 40	N

New Edit Remove

OK Save Cancel

Ready

In the PACKAGE window, the object name for each Package Line is displayed in the 'Object Name' column of the **STATUS** tab.

The object name field drives additional functionality:

- If the object name field is a FILE CHOOSER or an AUTO-COMPLETE field, multi-selection will automatically be enabled on this field when users add a line to a Package with this Object Type. If multiple values are selected, a new Package Line for each value will be created, allowing users to add multiple lines to a Package simultaneously.
- All migrations are tracked in the database tables KENV_ENV_CONTENTS and KENV_ENV_CONTENTS_HIST. The value of a Package Line's object name field is stored in these tables (along with other relevant data) whenever a migration occurs.
- You can query for the OBJECT NAME on the OBJECT TYPE WORKBENCH.

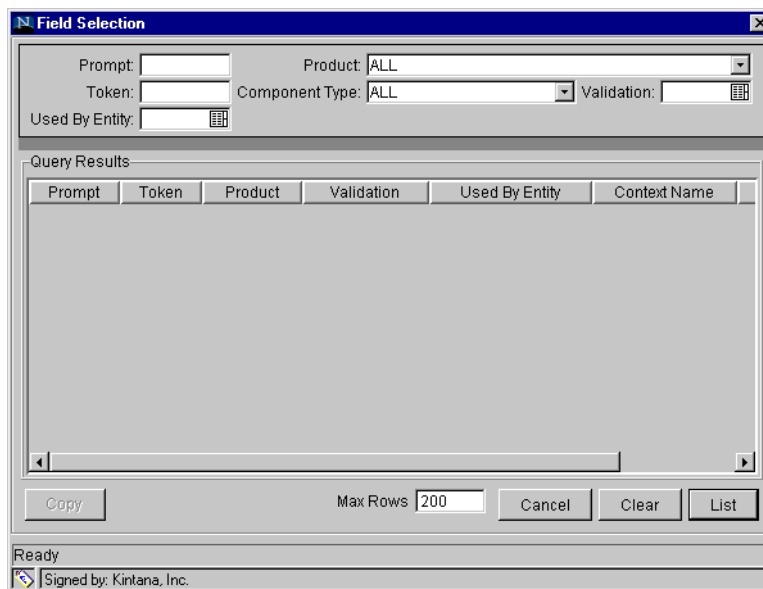
Setting the Object Revision

You can also create a field on the Object Type to represent the revision number of the object. This field will often be a numeric text field. You can then construct your deployment process to consider the object revision number when processing the Package.

Copying Object Type Fields

Use the **COPY FROM** functionality to streamline the process of adding fields to an Object Type by copying the definition of existing fields (from other Object Types). To copy a field:

1. Open the Object Type.
2. Click **NEW** in the **FIELDS** tab. The **FIELD** window opens.
3. Click **COPY FROM**. The **FIELD SELECTION** window opens.



4. Fields can then be queried by a number of criteria, such as the **TOKEN** name or field **PROMPT**. More complex queries can also be performed, such as listing all fields that reference a certain validation or are used by a certain entity. Due to the large number of Kintana fields, you should limit the list of fields by one or more of the query criteria.
5. Once a list of fields matching the selection criteria is obtained, highlight the desired field, and click **COPY**. This closes the window and copies the definition of the selected field into the **NEW FIELD** window.
6. Make any necessary modifications.
7. Click **OK**.

Editing Object Type Fields

To edit an existing field on an Object Type:

1. Open the Object Type.
2. Either double-click on the field in the **FIELDS** tab or select the field and click **EDIT**.

3. Make the desired changes in the header region, **ATTRIBUTES** tab, **DEFAULTS** tab, and **DEPENDENCIES** tab.
4. Click **APPLY** to apply the changes to the **FIELDS** tab without closing the field window, or select **OK** to apply the changes and close the **FIELD** window.



Note

Changes to fields for Object Types already used by existing Package Lines can have a significant impact. For example, if the column a field value gets stored in is changed, all existing Package Lines for this Object Type will now have incorrect data. Also, remember that Tokens can be used in Object Type commands and notifications; any changes to these could disrupt your system.

Changing information like **FIELD PROMPT** and **DESCRIPTION** should not affect the behavior of existing Package Lines.

Removing Fields

To remove a field permanently from an Object Type:

1. Open the Object Type.
2. Click the field in the **FIELDS** tab.

3. Click **REMOVE**.
4. Click **OK** or **SAVE** to save this change to the database.

This deletes the field from the list of fields.



Note

Removing a field from an Object Type does not change the historical information for existing Package Lines using the given Object Type. Any values for the deleted field remain in the Package Lines table in the column specified in the field definition.

Creating Object Type Commands

The following sections provide instructions and examples for configuring your Object Type commands.

- [Object Type Commands Overview](#)
- [Kintana Special Commands](#)
- [Command Steps](#)
- [Command Conditions](#)
- [Example Object Type Command Uses](#)

See "[Using Commands and Tokens](#)" for additional examples of using Commands in Kintana.

Object Type Commands Overview

Object Type commands define the heart of the execution layer within your deployment system. Commands tell Kintana precisely which steps must be executed at a specific Workflow step. This can involve such activities as migrating a file, executing a script, performing some data analysis, or compiling code.

Commands Interface

Commands are accessible through the **COMMANDS** tab of the Object Type screen and consist of command information and command steps. Summaries of both

parts of each command associated with the Object Type are visible in the **COMMANDS** tab.

Command steps are the shell script commands that make Object Types function. Double-click the Command Step to open the EDIT COMMAND window. The EDIT COMMAND window displays the shell script code in the STEPS window, as shown in *Figure 7-2*.

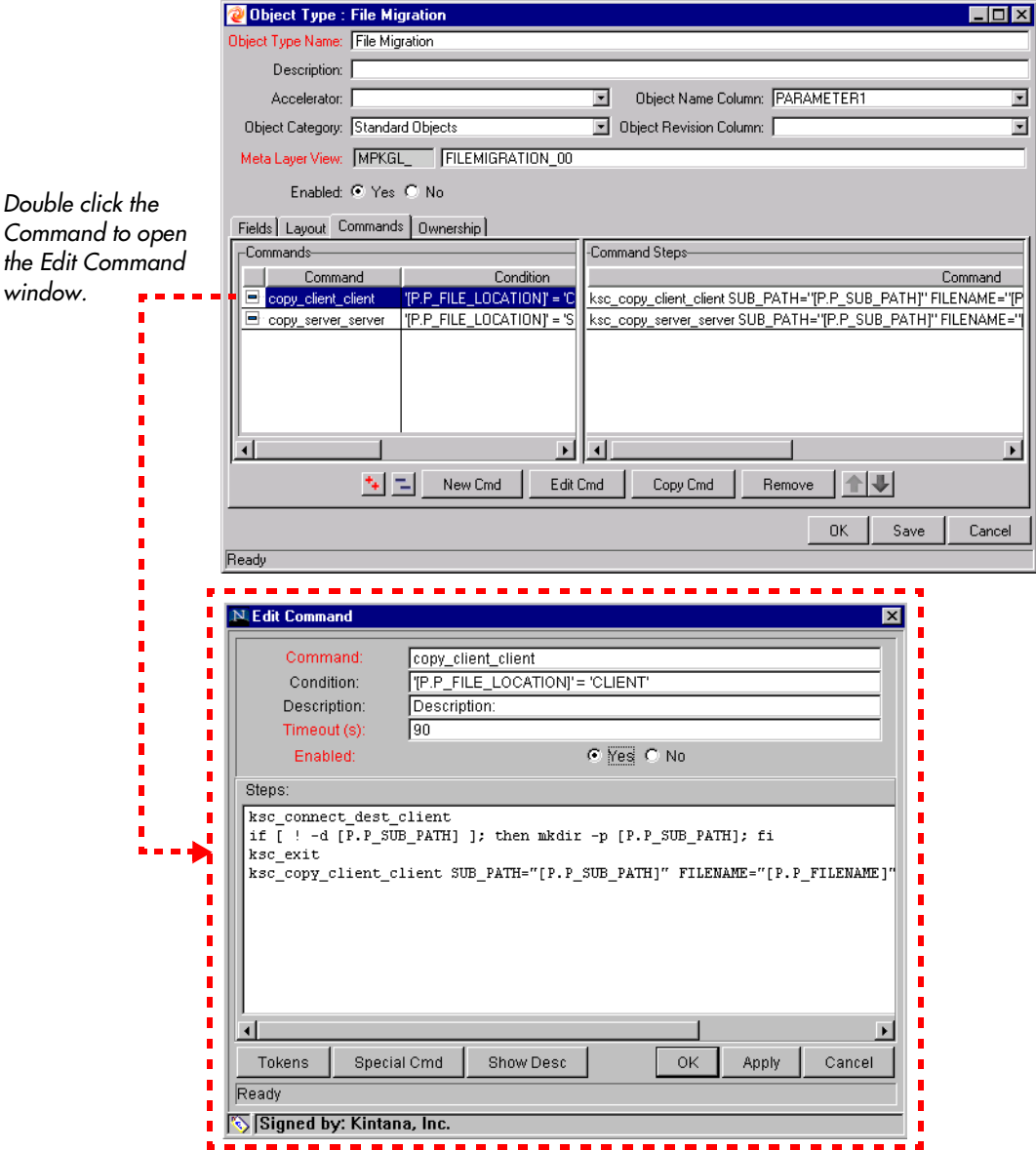


Figure 7-2 Object Type Commands Tab and Edit Command Window

To generate a new command, click **NEW CMD** in the **COMMANDS** tab. This opens the NEW COMMAND window shown in *Figure 7-3*. *Table 7-8* shows the fields included in this window.

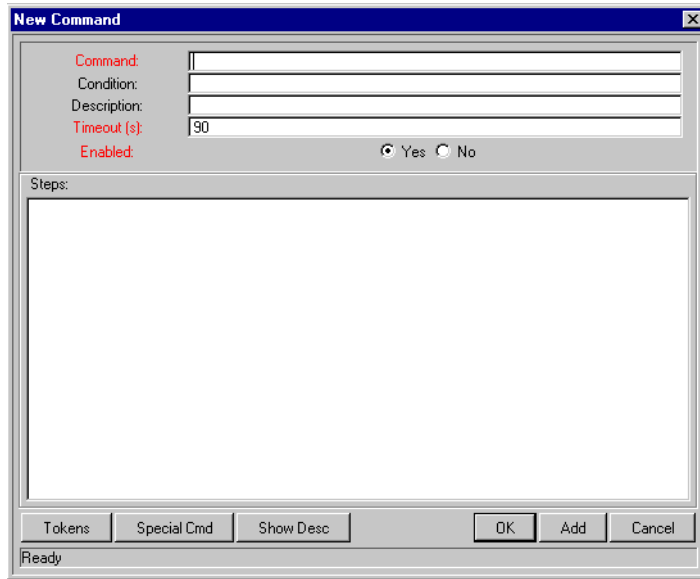


Figure 7-3 New Command Window

Table 7-8. New Command Window Fields

Field	Description
Command	A simple name for the command.
Condition	A condition that determines whether the command steps for the command are executed or not. (See <i>“Command Conditions”</i> on page 161 below for more information).
Description	A description of the command.
Timeout	The amount of time the command will be allowed to run before its process is terminated. This mechanism is used to abort commands that are hanging or taking an abnormal amount of time.
Enabled?	Determines whether the command is enabled for execution.

An Object Type may have many commands, and each command may have many command steps. A command may be viewed as a particular function for an object. Copying a file may be one command, and checking that file into

version control may be another. To perform these functions, a series of events needs to take place, and these events are defined in the command steps.

One additional level of flexibility is introduced when commands must be executed in certain cases. This is powered by the condition field of the object commands and is discussed in “*Command Conditions*” on page 161.

Object Type Commands and Workflow

Object Type Commands are tightly integrated with the Kintana Workflow engine. The commands contained in an Object Type are executed at EXECUTION Workflow steps.

It is important to note the following concepts regarding Command/Workflow interaction:

- To execute Object Type commands at a particular Workflow step, the Workflow step must be configured with the following parameters:
 - o Workflow step must be an Execution type step.
 - o WORKFLOW SCOPE = **PACKAGES**.
 - o EXECUTION TYPE = **BUILT-IN WORKFLOW EVENT**.
 - o WORKFLOW COMMAND = **EXECUTE_OBJECT_COMMANDS**.
- When the object reaches the Workflow step (with WORKFLOW COMMAND = **EXECUTE_OBJECT_COMMANDS**), all of the Object Type commands whose conditions are satisfied will be run in the order they are entered in the Object Type’s command panel.
- The Object Type can be configured to run only certain commands at a particular step. To do this, specify the “*Command Conditions*” on page 161.
- Each Object Type command can be configured so that only certain steps (within a command) are executed within a particular Workflow step. This is set using conditional statements within the command.

Kintana Special Commands

Kintana Object Types, Request Types, Report Types, Workflows and Validations all use commands to access the Kintana execution layer. In order

to simplify the use of command executions, Kintana contains a predefined set of Special Commands. Users can also create their own Special Commands.

Special Commands are commands with variable parameters and are used in Object Types, Request Types, Report Types, Workflows, Workflow steps, and Validation command steps. These command steps perform a variety of functions, such as copying files between environments and establishing connections to environments for remote command execution. Kintana features two types of Special Commands:

- **System Special Commands** - These commands are shipped with Kintana. System Special Commands are read-only and have the naming convention “ksc_command_name.” System Special Commands always begin with “ksc_.”
- **User Defined Special Commands** - These commands are user-defined and have the naming convention “sc_command_name.” User-defined Special Commands must begin with “sc_.”

Kintana Special Commands act as sub-programs that can be reused where ever needed. It is often more convenient to create a Special Command for a program that will be used in multiple places rather than placing the individual commands into every Object Type.

Command Steps

Command steps represent the actual directives that Kintana Deliver specifies to an environment's host as it tries to execute the commands for that instance of an object. [Table 7-9](#) describes the fields in the COMMAND STEPS region of the NEW/EDIT COMMANDS dialog.

Table 7-9. Command Steps

Field	Description
Steps	Defines the command-line directive or special command to be issued.
Description	Describes each of the command steps.

A command step can be an actual command-line directive that is sent to the target machine or can be one of Kintana Deliver's many “special commands.”

 Note

The Kintana Execution Engine will execute the commands and command steps in the order they are displayed in the **COMMANDS** tab. To change the order of the commands or the command steps, in the **COMMANDS** tab, select the given command or command step and use the arrow buttons to move the selected item.

Command Conditions

In many situations, it may be necessary to run a different set of commands depending on the context of execution. This flexibility is achieved through the use of conditional commands. The **CONDITION** field for an object command is used to define the situation under which the associated command steps execute.

Conditions are evaluated as boolean expressions. If the expression evaluates to true, the command is executed. If false, the command is skipped and the next command is evaluated. If no condition is specified, the command is always executed. The syntax of a condition is identical to the “where” clause of a SQL statement, which allows enormous flexibility when evaluating scenarios. Some example conditions are detailed in the following table:

Table 7-10. Example Conditions

Condition	Evaluates to
BLANK	Command will be executed in all situations.
'[P.P_VERSION_LABEL]' IS NOT NULL	Command will be executed if the parameter with the token P_VERSION_LABEL in the Package line is not null.
'[DEST_ENV.ENVIRONMENT_NAME]' = 'Archive'	Command will be executed when the destination environment is named “Archive”.
'[AS.SERVER_TYPE_CODE]' = 'UNIX'	Command will be executed if the application server is installed on a UNIX machine.

 Note

The single quotes are necessary for strings.

Example Object Type Command Uses

This section provides a number of operations that you can execute using Object Type commands. This section will use a combination of UNIX commands, third party commands, and Kintana Special Commands.

- Commands for connecting to machines.
 - o Connect to the destination environment and run system commands
 - o Connect to an alternate environment and run command (Environment override)
- Commands for manipulating data. (Kintana fields and other info stored in files or database)
 - o Set a value in a Package Line
 - o Create, run and delete a script
 - o Extract information from a file (version number)
- Commands for running operating system-specific commands. (NT and Unix)
 - o Starting a server
 - o Stopping a server
- Commands for running program-specific commands
 - o Checking files in and out of a Version control system
- Commands for copying files

Chapter
8**Defining your Environments**

This chapter provides an overview for defining the Environments that will be used in your deployment system. This activity requires some knowledge of the machines, filesystem, and databases that are serving as your deployment sources and destinations. Kintana's Environment definitions include system account information, passwords, and supported communication protocols for those machines. This chapter illustrates how the information gathered in "*Gathering Process Requirements and Specifications*" on page 49 can be used to define the Environments. See "*Environment Workbench Reference*" for details on Environment screens and fields.

This chapter discusses the following topics:

- *Environment Requirements*
- *Defining Environments in Kintana*
- *Testing the Environment Setup*
- *Creating Environment Groups*
- *Linking Environments and Environment Groups to Workflows*
- *Environment Maintenance and Utilities*

Environment Requirements

When migrating objects, Kintana Deliver logs onto remote computers in the same way any other user would (using FTP, SCP, SSH or Telnet). Kintana can logon using any existing username and password. However, it is recommended that a new user 'Kintana' be generated on each computer that Kintana Deliver will access. This will help clarify the setup and relieve some administrative

burden. The 'Kintana' user should have full access to the Kintana home directory on the Kintana server as well as the correct read and write permissions on other required directories. In addition, on Windows NT computers, the 'Administrators' group must have read access to Kintana's home directory. (Any Windows NT computer that Kintana Deliver will access should have been configured as directed in "*Kintana Installation Guide*", which is available on the Kintana Web Site).

Defining Environments in Kintana

To define a new environment, perform the following steps:

1. Open the Environment Workbench by clicking **ENVIRONMENTS** in the shortcut bar and clicking the **ENVIRONMENTS** icon.
2. Click **NEW ENVIRONMENT** on the ENVIRONMENT WORKBENCH WINDOW or select **FILE -> NEW -> ENVIRONMENT**. The ENVIRONMENT window appears.
3. Enter the Environment name in the ENVIRONMENT NAME field.
4. Enter a description of the Environment in the DESCRIPTION field.
5. Specify the physical location of the server in the LOCATION field.
6. Click **YES** or **NO** for the ENABLED radio button to indicate whether the Environment is available for use in a Workflow.
7. Click the **HOST** tab. Enter the following information about the server:
 - a. Enter the name for the server in the SERVER NAME field.
 - b. Select the type of server from the TYPE drop down list.
 - c. Enter the name of the user whom Kintana Deliver will logon to the server specified in the USERNAME field. This will usually be KINTANA if that user account has been generated on this computer.
 - d. Enter the corresponding password by clicking **C** and typing the password in the CHANGE PASSWORD window.
 - e. If applicable, enter the NT DOMAIN.

f. Specify the complete path that Kintana Deliver will go to upon logging in to the server in the **BASE PATH** field. (*Remember: use forward slashes to separate directories, even on Windows systems.*)

g. Select the **CONNECTION PROTOCOL**.

SSH is a secure connections protocol, whereas Telnet is not. In order to use SSH as your connection protocol, you must first setup SSH on the target machine.

h. Select the **TRANSFER PROTOCOL**.

SCP is a secure transfer protocol, whereas FTP is not. In order to use SCP as your transfer protocol, you must first setup SCP on the target machine.

8. Repeat steps “a” through “h” in step 7 to enter information about the client.

9. Specify the following information about the database.

a. Select the **SERVER TYPE**.

b. If **ORACLE SERVER** is selected:

i. Enter the name of the database in the **HOST NAME** field.

ii. Specify the database connection string to connect to the desired database in the **CONNECT STRING** field.

iii. Enter the username of the database schema in the **USERNAME** field.

iv. Enter the corresponding password in the **PASSWORD** field.

v. Enter the Oracle **SID** of the database instance in the **ORACLE SID** field.

vi. Enter the **PORT NUMBER**.

vii. Enter the **DB LINK**.

viii. Enter the database version number in the **DB VERSION** field.

c. If **SQL SERVER** is selected:

i. Enter the name of the server in the **SERVER NAME** field.

- ii. Enter the name of the database in the **DB NAME** field.
 - iii. Enter the username of the database schema in the **USER LOGIN** field.
 - iv. Enter the corresponding password in the **PASSWORD** field.
 - v. Enter the **PORT NUMBER**.
 - vi. Enter the database version number in the **DB VERSION** field.
10. Select the **APPLICATIONS** tab and click the **NEW APP, COPY APP, or COPY APPS FROM** button to select/generate an App Code. Enter the relevant information, if any, for the App Codes to be generated. Only enter data for each Application that should override the same data at the Environment Host level. See *“Using App Codes with Your Environment”* on page 170.
 11. Click the **OWNERSHIP** tab to specify the users that can edit, copy and delete the Environment.
 12. Click the **USER ACCESS** tab to specify the users that can use this Environment in Workflow Steps and Environment Groups.
 13. Enter any necessary information in the **USER DATA** tab.
 14. If enabled, move to the **ACCELERATOR DATA** tab. Select the desired **ACCELERATOR** tab at the bottom of the window and enter the information for correctly defining that environment.

Note

For detailed information regarding one or more of the purchased Accelerators, see the Kintana Accelerator information located on the Kintana Web Site at <http://www.kintana.com/products/accelerators/accelerators.html>.

15. Click **SAVE** in the **ENVIRONMENTS** window to register all the environment information entered so far.
16. Click **CHECK** to run the Environment checking functionality for this new Environment.

Note

The Enable Server, Enable Client and Enable Database buttons should generally remain checked, and the information below the appropriate headings should be entered. These check boxes can be used as flags in conjunction with Object Type commands to further define multi-tiered environments.

Copying Environments

To copy an Environment:

1. Open the ENVIRONMENT WORKBENCH by clicking **ENVIRONMENTS** in the shortcut bar and clicking the **ENVIRONMENTS** icon.
2. Enter the search criteria required to select the Environments in the **QUERY** tab of the ENVIRONMENT WORKBENCH and click **LIST**. The **RESULTS** tab opens, displaying the results of the search.
3. Select the Environment to be copied in the **RESULTS** tab.
4. Click **COPY**. The **COPY** window appears.
5. Enter the new Environment name as well as any additional information and click **OK**.

This generates the new Environment. The new Environment can then be opened and the desired changes can be made.

Note

You must be a member of one of the Environment's Ownership Groups in order to copy an Environment. See "[Setting Configuration Security](#)" on page 210 for details.

Selecting the Environment's Connection Protocol

You need to specify which communication protocol should be used to connect to the server or client specified in the Environment. This protocol will be used by commands in Kintana to connect to source and destination Environments in your deployment system. Work with your system administrator to determine which connection protocols are supported at your site for the machines housing your deployment environments.

Kintana supports the following connection protocols:

- Telnet
- SSH
- SSH2

Selecting the Environment's Transfer Protocol

You need to specify which transfer protocol should be used to transfer files to the server or client specified in the Environment.

Kintana supports the following transfer protocols:

- FTP
- FTP (active)
- FTP (passive)
- Secure Copy
- Secure Copy 2

Configuration notes:

Choose the transfer protocol best suited to your business and technology need. You should consider factors related to security and performance when selecting the transfer protocol. Work with your system administrator to determine which connection protocols are supported at your site for the machines housing your deployment environments. The following list provides some suggestions for when to use the above protocols.

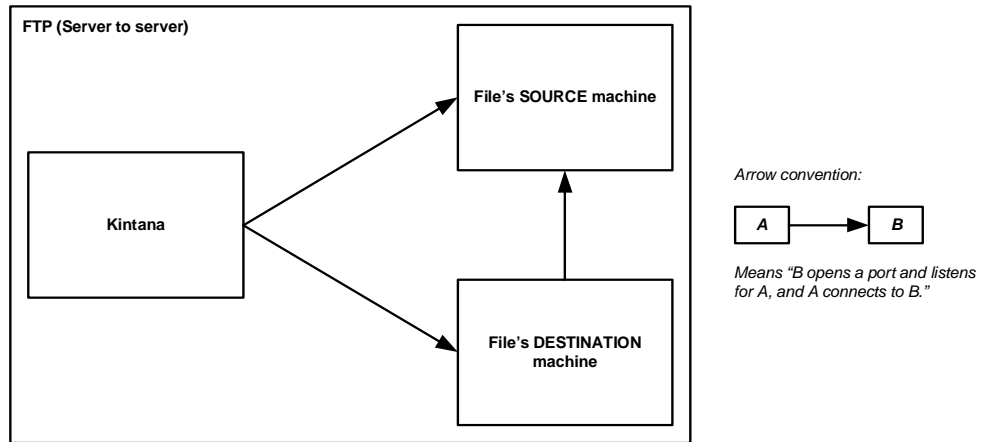
No additional Kintana configuration is required to enable one FTP mode over another (i.e. Kintana server parameters). Administrators do, however, need to consider their FTP server configuration (particularly as they relate to security and firewall settings) when selecting an FTP protocol for transferring data.

Selecting the FTP protocol:

The following capabilities must be enabled on the source and destination machines for the following FTP protocol selection to function properly:

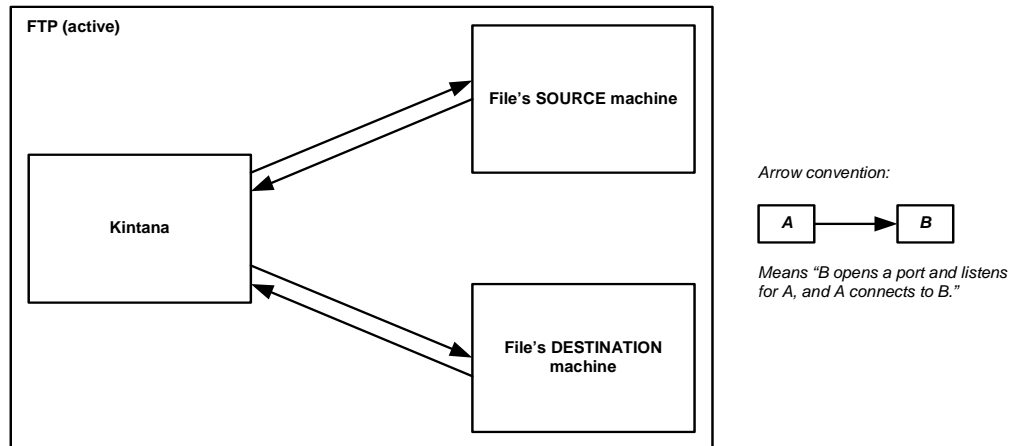
FTP:

- Either the source or the destination environment needs to allow outgoing connections to a third party.
- FTP PORT command must be enabled on one of the environments
- FTP PASV command must be enabled on the other environment



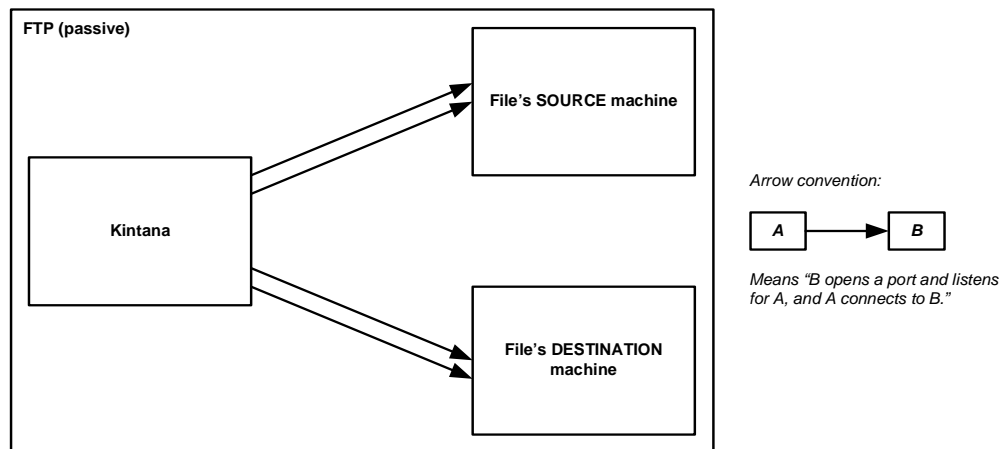
FTP (active):

- PORT command must be enabled on both the source and destination environments (allows outgoing back to the requestor)



FTP (passive):

- PASV must be enabled on both the source and destination environments. In this configuration, Kintana sends a command to the source or destination instructing that environment to open a port. Kintana then connects to that port.)



Using App Codes with Your Environment

Complex Environments are often segmented into subsections called Environment Applications. The Environment information consists of the default set of attributes for an Environment. It is rare, however, that an actual Environment could be described simply by this set of defaults. For example, files belonging to different applications may reside at different paths and may be owned by different users. SQL scripts may need to be run against a different schema than the one defined in the Host panel.

When adding a line to a Package there is the option of choosing the “App Code” to specify the application that the migration object belongs to. When that object is subsequently migrated, the application-specific Environment items are referenced in place of the default Environment items. As a general rule, any application-specific Environment item that has no value is substituted by the corresponding Environment value.



Note

Environment User Data fields will be inherited by each App Code and will appear in the App Codes' **USER DATA** tab. App Code User Data fields behave like other App Code fields (such as host name and base path), in that blank field values indicate that the App Code has the same value as its parent Environment. Therefore, required Environment User Data fields are not also required at the App Code level.

Every Kintana Deliver Environment can contain its own set of applications. The **APPLICATIONS** tab of the ENVIRONMENT window is shown in [Figure 8-1](#).

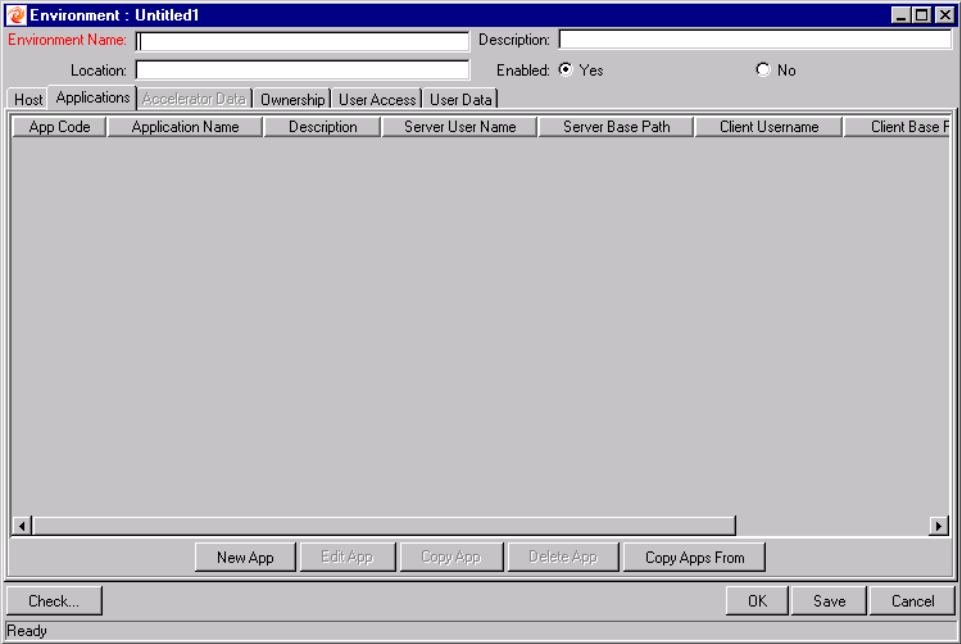


Figure 8-1 Environment Window - Applications Tab

The **APPLICATIONS** tab consists of the various fields and buttons shown in the [Table 8-1](#). Application fields do not always have to be populated. Click **NEW APP** to open the NEW APPLICATION CODE window, where you can define a new App Code.

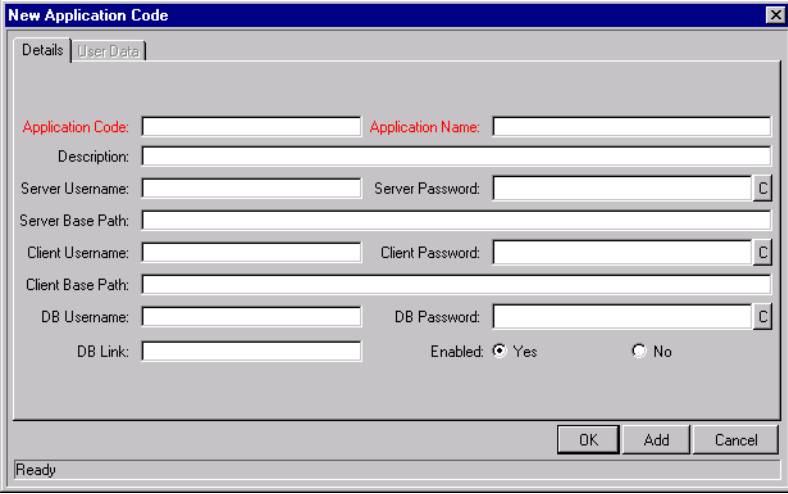


Figure 8-2 Application Code Window

Table 8-1. Environment Window - Applications Tab - Application Fields

Field Name	Description
App Code	The short name abbreviation for the application.
Application Name	The long name for the application.
Description	A description of the application.
Server User Name	The username that Kintana Deliver should logon as when transferring files or running commands on the Environment server for this application, if it is different from the default server username.
Server Password	The password for logging on to the server, if different from the default server password. This field is encrypted.
Server Base Path	The base path for the application on the server machine.
Client User Name	The username that Kintana Deliver should log in as when transferring files or running commands on the Environment client for this application, if different from the default client name.
Client Password	The password for logging on to the client, if different from the default client password. This field is encrypted.
Client Base Path	The base path for the application on the client machine.
DB Username	The DB username for the application. It is used when running database level commands (such as SQL scripts) for this application.
DB Password	The DB password for the application. It is used when running database level commands (such as SQL scripts) for this application. This field is encrypted.
DB Link	The name of the database link for this application, if different from the default Environment DB Link.
Enabled	Identifies if this application Environment is currently enabled.
New App	Brings up a dialog allowing the user to generate a new App Code.
Edit App	Allows the user to edit the selected App Code.
Copy App	Brings up a dialog box that allows the user to copy a selected App Code.
Delete App	Removes the selected App Code.

Table 8-1. Environment Window - Applications Tab - Application Fields

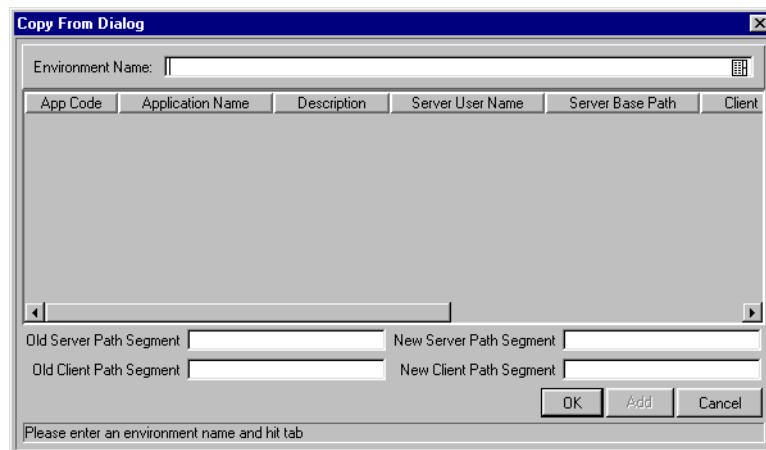
Field Name	Description
Copy Apps From	Brings up a dialog box that allows a user to copy App Codes from other Environments. For more information see “Copying App Codes from Other Environments” on page 173.

Copying App Codes from Other Environments

When generating a new Environment, all or some of the applications attached to an existing Environment can be copied to the new Environment to speed up the set-up process.

To copy the App Codes:

1. Open the ENVIRONMENT WORKBENCH by clicking **ENVIRONMENTS** in the shortcut bar and clicking the **ENVIRONMENTS** icon.
2. Click **NEW ENVIRONMENT** on the ENVIRONMENT WORKBENCH or select **FILE > NEW ENVIRONMENT**. A blank ENVIRONMENT window appears.
3. Click the **APPLICATIONS** tab.
4. Click **COPY APPS FROM**. The COPY FROM DIALOG opens.



5. Select the Environment from which the applications are to be copied from the ENVIRONMENT NAME auto-complete list.

The names of App Codes are displayed in the list.

6. To change the base path segment of all the app codes selected, enter the old server/client base path segment and the new server/client base path segment in the appropriate fields.
7. Select all the applications to be copied and click **ADD**.

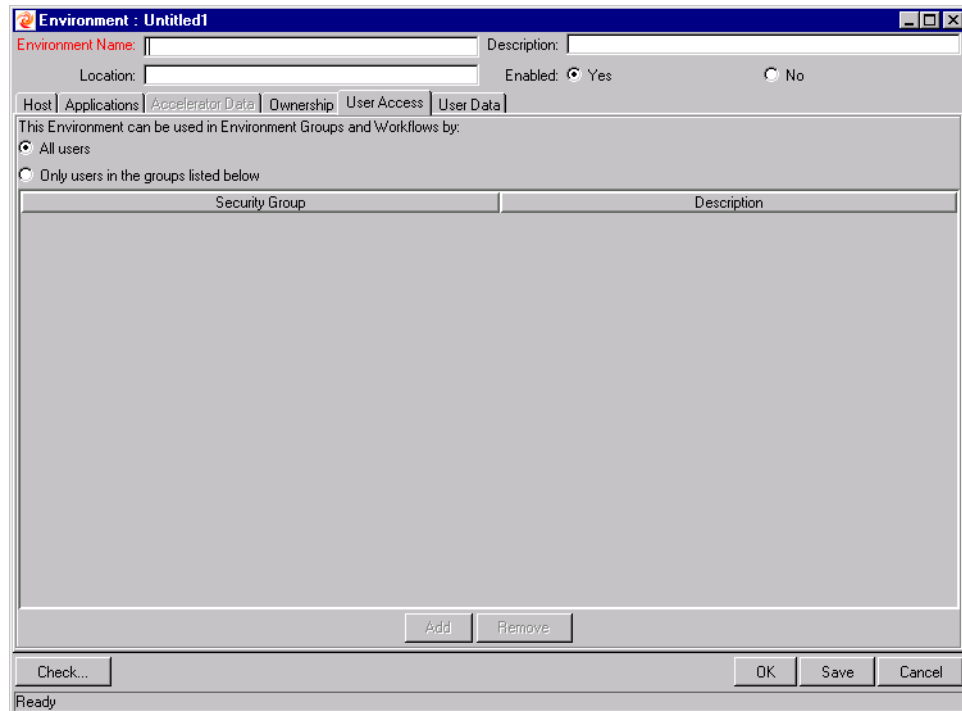
These fields have, by default, the server or client base path of the Environment from which the applications are being copied. For example, suppose that two applications have been selected. The server base paths for these two applications are “/u2/apps/isi” and “/u2/apps/demo_107”. To change u2 to u3, enter “u2” in 'OLD SERVER BASE PATH' and “u3” in 'NEW SERVER BASE PATH' before copying these applications. Every occurrence of the old server/client base path segment will be changed to the new base path segment in all the applications selected. The changes will be reflected in the applications in the Environment into which they were copied.

8. After copying the App Codes, any necessary modifications such as adding additional applications, deleting applications, or editing any of the applications can be made.

Setting the Access for Environments

You can control which Kintana users can access an Environment for use in Environment Groups and Workflows. To specify who can use an Environment in Environment Groups and Workflows:

1. Click the **ENVIRONMENTS** shortcut bar and click the **ENVIRONMENTS** icon. The **ENVIRONMENT WORKBENCH** opens.
2. Click **NEW ENVIRONMENT** to create a new Environment or click **LIST** to open an existing Environment. The **ENVIRONMENT** window opens.
3. Click the **USER ACCESS** tab.



4. Select the **ONLY USERS IN THE GROUPS LISTED BELOW** option.
5. Click **ADD**. The **ADD SECURITY GROUPS** window opens.
6. Select one or more Security Groups from the **SECURITY GROUP** auto-complete list.
7. Click **OK** to close the **ADD SECURITY GROUP** window. The Security Group you selected display in the **ACCESS** tab.
8. Click **ADD** to add more Security Groups. Click **OK** to save the changes and close the window. Click **SAVE** to save the changes and leave the **ENVIRONMENT** window open.

Now only members of the Security Group(s) specified in the **USER ACCESS** tab can use this Environment in Environment Groups and Workflows.

Creating Environment Groups

Environment Groups define a set of Kintana Environments which can be referenced as the Source or Destination for object migrations. Environment Groups are defined and edited using the ENVIRONMENT GROUP WORKBENCH.

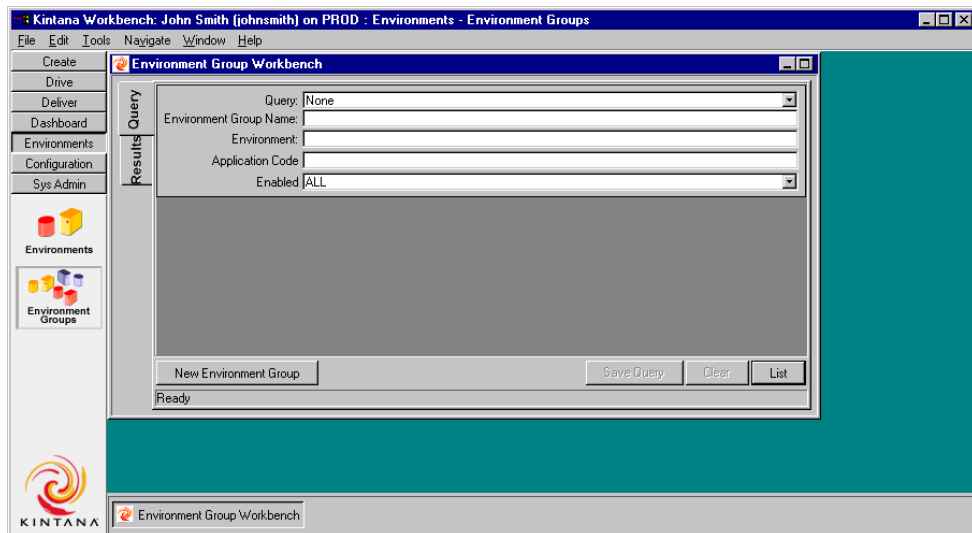
When to Use Environment Groups

Use Environment Groups where it is desirable to execute a Workflow Step on multiple Environments. For example, it may be necessary to migrate an object to multiple testing Environments for different targeted tests. These multiple Environments can be referenced together in one Environment Group.

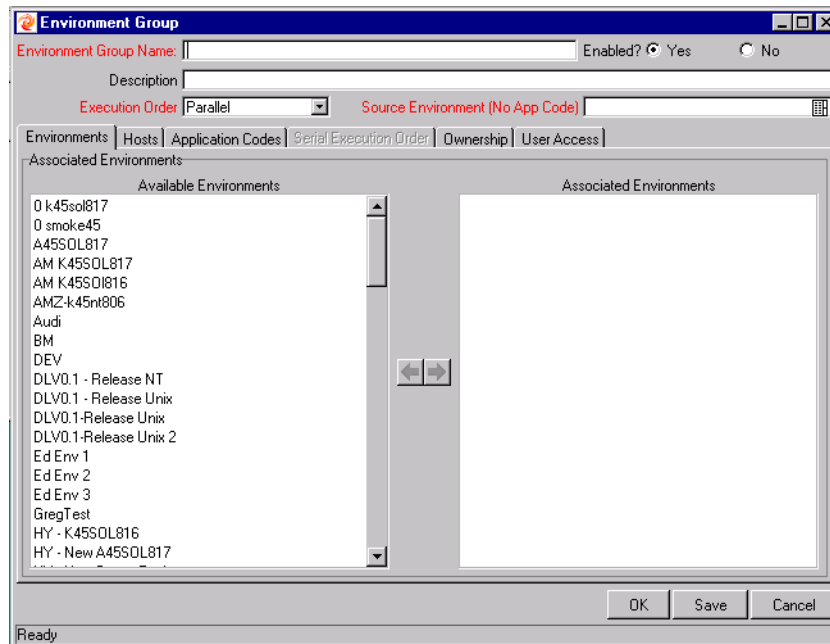
Defining an Environment Group

To define a new Environment Group:

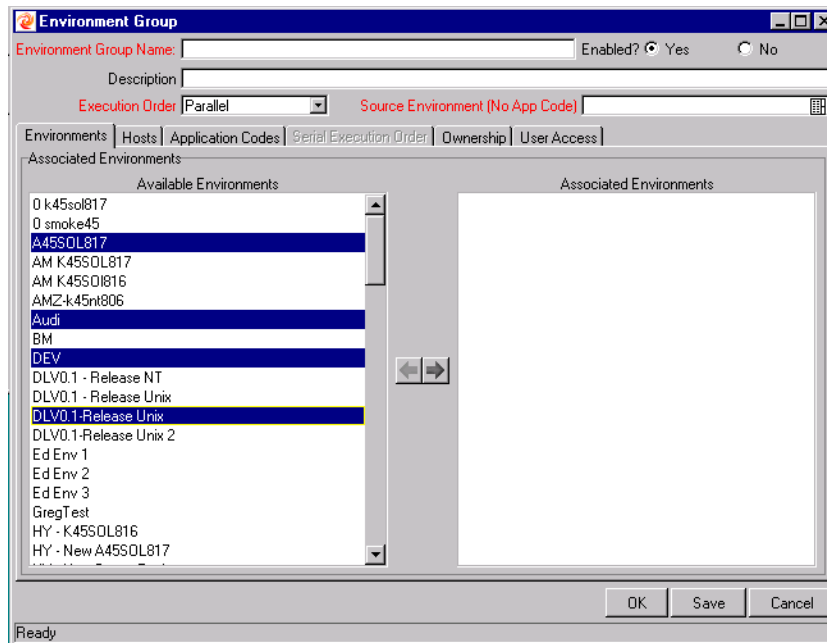
1. Click **ENVIRONMENTS** in the shortcut bar and click the **ENVIRONMENT GROUPS** icon.



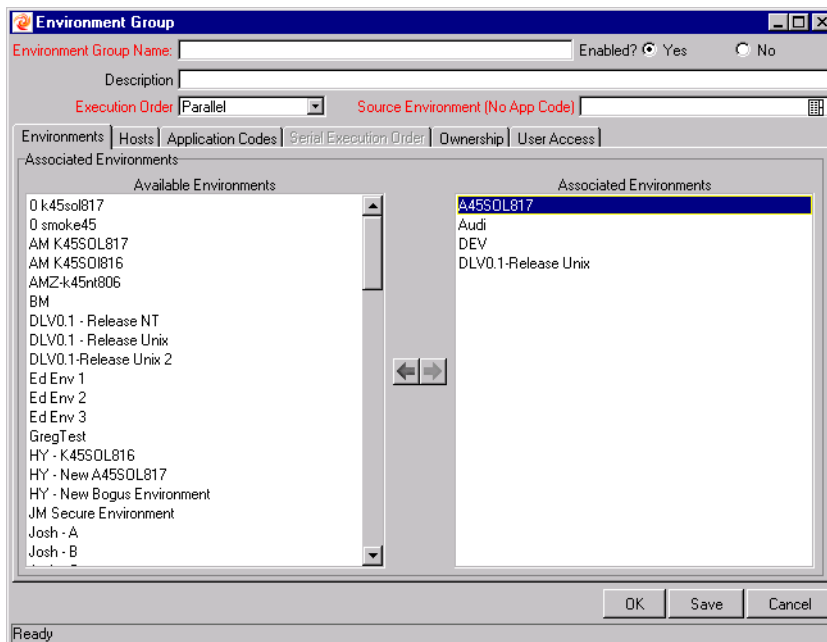
2. Click **NEW ENVIRONMENT GROUP**. The ENVIRONMENT GROUP window opens.



3. Enter an ENVIRONMENT GROUP NAME and DESCRIPTION.
4. Select an EXECUTION ORDER (either **PARALLEL** or **SERIAL**).
5. Select a SOURCE ENVIRONMENT. This is the single Environment that will be used as the source of the deployment when an Environment Group is specified as the Source Environment in a Workflow step.
6. Select the Environments to be included in the Environment Group from the AVAILABLE ENVIRONMENTS list. Press **CTRL** and mouse click to select nonadjacent items in the list. Press **SHIFT** and mouse click to select multiple adjacent Environments.



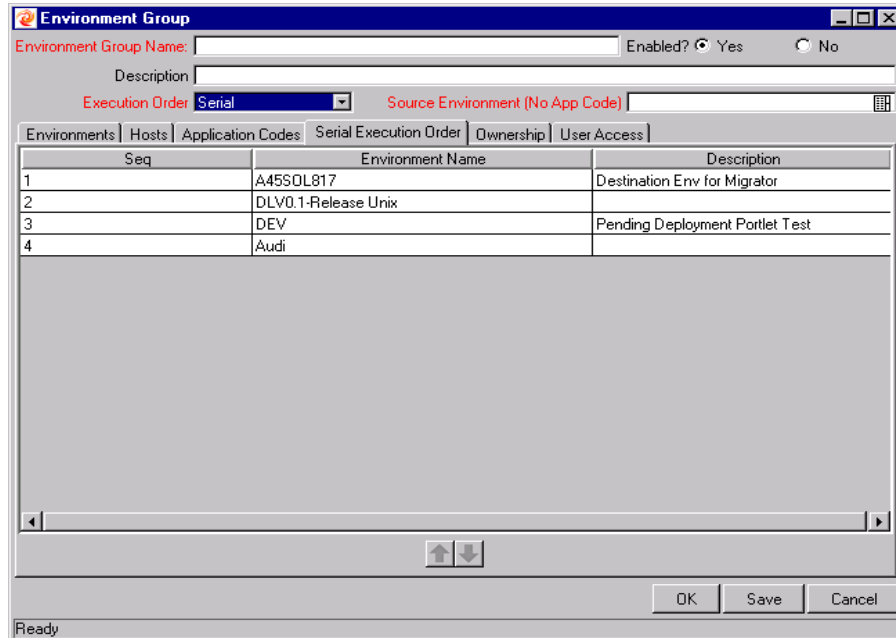
7. Click the right arrow to move the selected Environments to the ASSOCIATED ENVIRONMENTS list.



8. Click the **APPLICATION CODES** tab.
9. Select a PRIMARY SOURCE ENVIRONMENT for the listed Application Codes.
Note that this is useful when specifying a Source Group Environment for

an Execution step. See [“Choosing the Source Environment Based on Selected App Code”](#) on page 189 for more information.

- Click the **SERIAL EXECUTION ORDER** tab. This tab is only enabled if **SERIAL** is selected from the EXECUTION ORDER drop down list.



- Change the execution order by selecting an Environment and clicking the up and down arrows.
- Click the **OWNERSHIP** tab to specify the users that can edit, copy and delete the Environment Group.
- Click the **ACCESS** tab to specify the users that can use this Environment Group in Workflows.
- Verify that the Enabled radio button is set to **YES** to enable use of this Environment Group in Workflow Step configuration.
- Click **OK**.

This defines the new Environment and closes the ENVIRONMENT GROUP window.

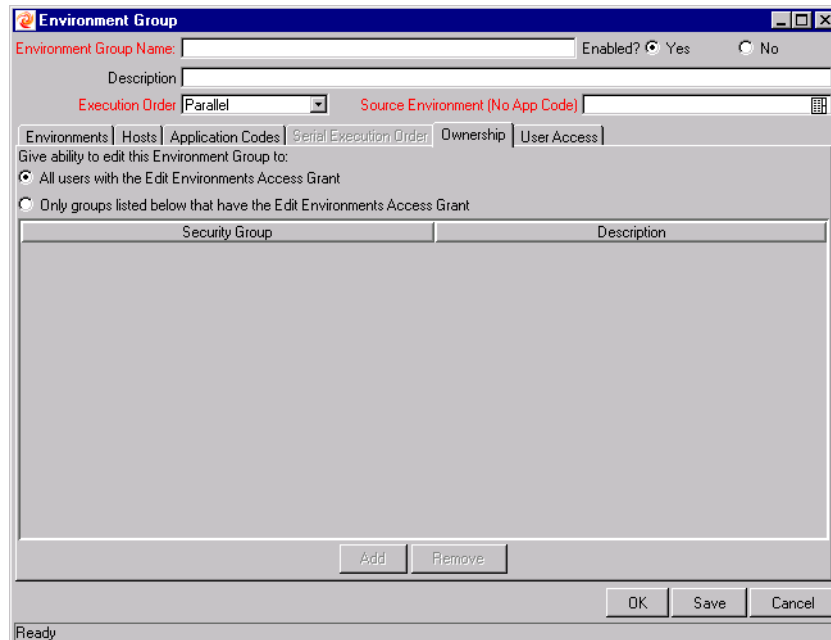
Setting Ownership for Environment Groups

Different groups of Kintana users can have exclusive control over the Environment Groups used by their group. These groups are referred to as Ownership Groups. Members of the ownership group are the only users who can edit, delete or copy the Environment Group. Each Environment Group can be assigned multiple ownership groups.

Ownership Groups are defined by adding Security Groups to the **OWNERSHIP** tab.

To set the Ownership for an Environment Group:

1. Open the ENVIRONMENT GROUP window.
2. Click the **OWNERSHIP** tab.



3. Select the **ONLY GROUPS LISTED BELOW THAT HAVE THE EDIT ENVIRONMENTS ACCESS GRANT** option.
4. Click **ADD**. The ADD SECURITY GROUPS window opens.
5. Select one or more SECURITY GROUPS.
6. Click **OK** to close the ADD SECURITY GROUP window. The Security Group you selected display in the **OWNERSHIP** tab under the Security Group column.

7. Click **ADD** to add more Security Groups. Click **OK** to save the changes and close the window. Click **SAVE** to save the changes and leave the ENVIRONMENT GROUP window open.

Only members of the Security Group(s) specified in the OWNERSHIP window can edit, delete or copy this Environment Group.



Note

If no Ownership groups are associated with the entity, the entity is considered global and any user with the Edit Access Grant for the entity can edit, copy or delete it. Refer to the "[Kintana Security Model](#)" for more information on Kintana Access Grants.

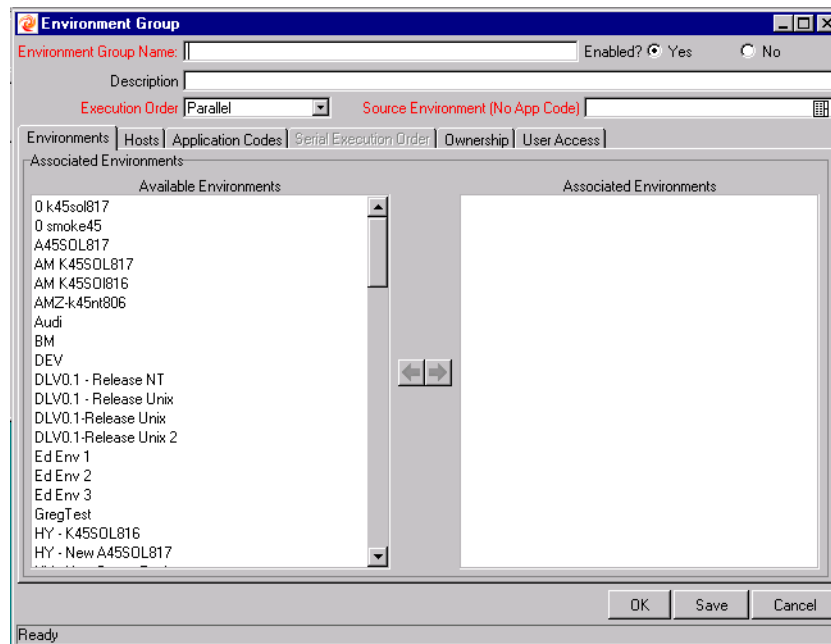
Kintana administrators have the 'Ownership Override' access grant and can access configuration entities even if the administrator is not a member of one of the Ownership Groups and does not have the Edit Access Grant.

If a Security Group is disabled or loses the Edit Access Grant, that group will no longer be able to edit the entity.

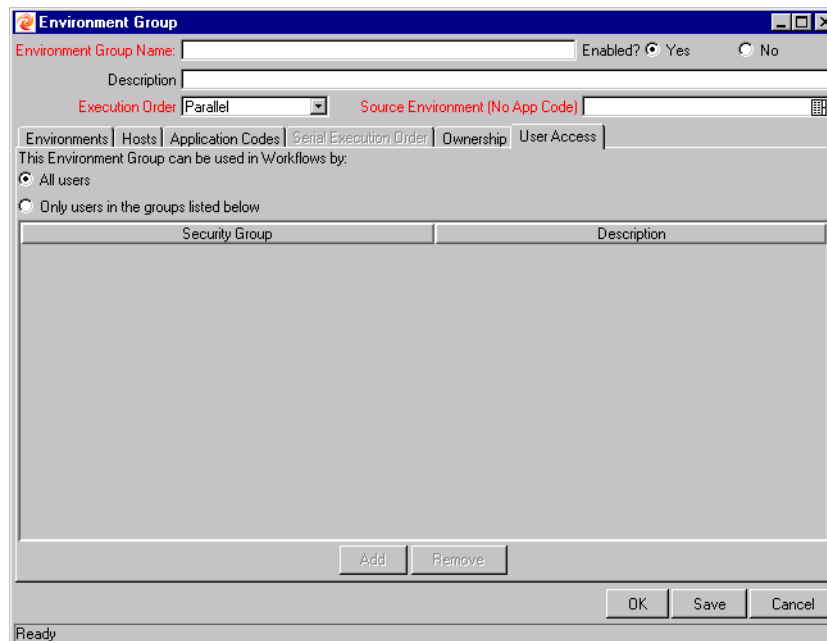
Setting the Access for Environments

You can control which Kintana users can access an Environment Group for use in Workflows. To specify who can use the Environment Group when defining a Workflow:

1. Open the ENVIRONMENT GROUP window.



2. Click the **ACCESS** tab.



3. Select the **ONLY USERS IN THE GROUPS LISTED BELOW** option.

4. Click **ADD**. The **ADD SECURITY GROUPS** window opens.

5. Select one or more **SECURITY GROUPS**.

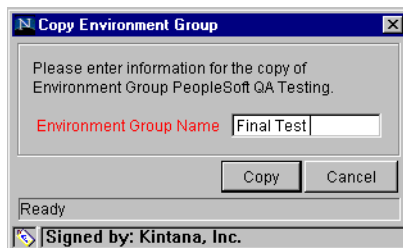
6. Click **OK** to close the **ADD SECURITY GROUP** window. The Security Group you selected display in the **USER ACCESS** tab.
7. Click **ADD** to add more Security Groups. Click **OK** to save the changes and close the window. Click **SAVE** to save the changes and leave the **ENVIRONMENT GROUP** window open.

Only members of the Security Group(s) specified in the **USER ACCESS** tab can use this Environment Group in Workflows.

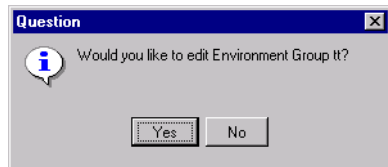
Copying an Environment Group

To generate a new Environment Group by copying an existing Environment Group:

1. Enter search criteria in the **QUERY** tab of the **ENVIRONMENT GROUP WORKBENCH** for the Environment Group to be copied.
2. Click **LIST** to find all Environment Groups that match the search criteria.
3. In the **RESULTS** tab of the **ENVIRONMENT GROUP WORKBENCH**, select the Environment Group to be copied.
4. Click **COPY**. The **COPY ENVIRONMENT GROUP** window opens.



5. Enter the new name for the copied Environment Group.
6. Click **OK** to copy the Environment Group. The following **QUESTION** dialog appears.



7. Click **YES** to edit the Environment Group or **No** to exit.



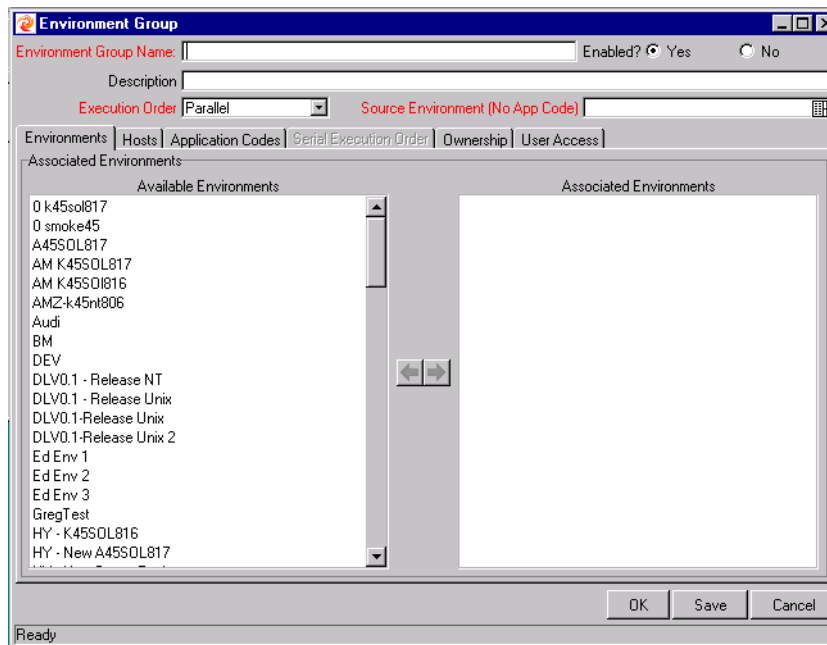
Note

You must be a member of one of the Environment Group's Ownership Groups in order to copy an Environment Group. See [“Setting Ownership for Environment Groups”](#) on page 180 for details.

Adding Environments to an Environment Group

To add an Environment to an existing Environment Group:

1. Enter search criteria in the **QUERY** tab of the ENVIRONMENT GROUP WORKBENCH for the Environment Group to be altered.
2. Click **LIST** to find all Environment Groups that match the search criteria.
3. Select the Environment Group in the **RESULTS** tab and click **OPEN**.



4. Select the Environment(s) to be added to the Environment Group in the AVAILABLE ENVIRONMENTS list.
5. Click the right arrow to move the selected Environment(s) into the ASSOCIATED ENVIRONMENTS list.
6. Click the **APPLICATION CODES** tab and modify the Primary Source specification as desired.
7. Click the **SERIAL APPLICATION ORDER** tab, if enabled, and modify the Environment sequence as desired.
8. Click **OK**.

This adds the new Environment to the existing Environment Group.

Note

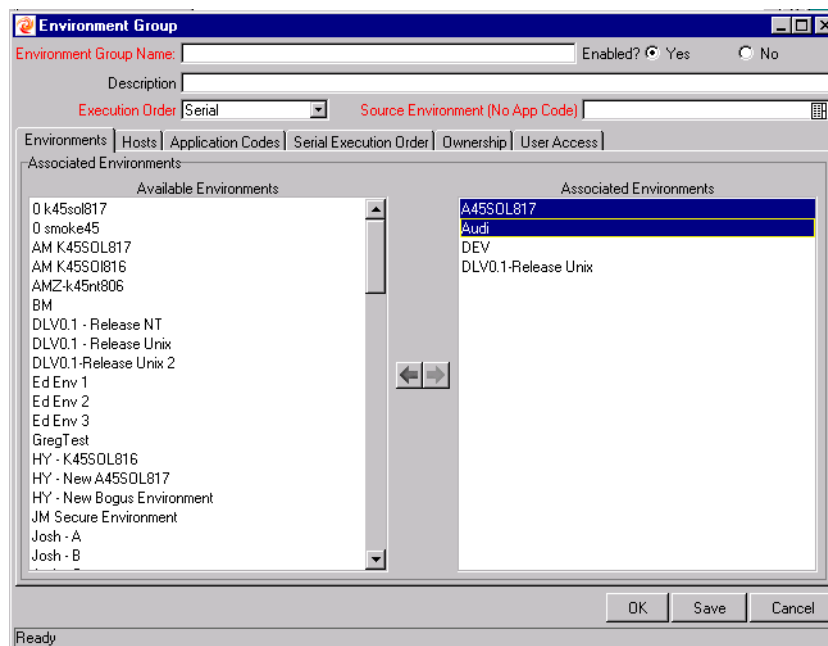
You must be a member of one of the Environment Group's Ownership Groups in order to modify an Environment Group. See "[Setting Ownership for Environment Groups](#)" on page 180 for details.

Only Environments that allow access (defined in the Access tab) will be included in Available Environments list. See "[Setting the Access for Environments](#)" on page 174 for more information.

Removing Environments from an Environment Group

To remove an Environment from an existing Environment Group:

1. Enter search criteria in the **QUERY** tab of the ENVIRONMENT GROUP WORKBENCH for the Environment Group to be altered.
2. Click **LIST** to find all Environment Groups that match the search criteria.
3. Select the Environment Group in the **RESULTS** tab and click **OPEN**.



4. Select the Environment(s) that to be removed from the Environment Group in the ASSOCIATED ENVIRONMENTS list.
5. Click the left arrow to move the selected Environment(s) into the AVAILABLE ENVIRONMENTS list.
6. Click on the **APPLICATION CODES** tab and modify the PRIMARY SOURCE specification as needed. If the Environment which was designated as the Primary Source is removed, then it is necessary to select a new Primary Source for the Application Code. See [“Choosing the Source Environment Based on Selected App Code”](#) on page 189 for more information.
7. Click the **SERIAL APPLICATION ORDER** tab, if enabled, and modify the Environment sequence as desired.

- Click **OK**.

This removes the Environment from the Environment Group and closes the window.



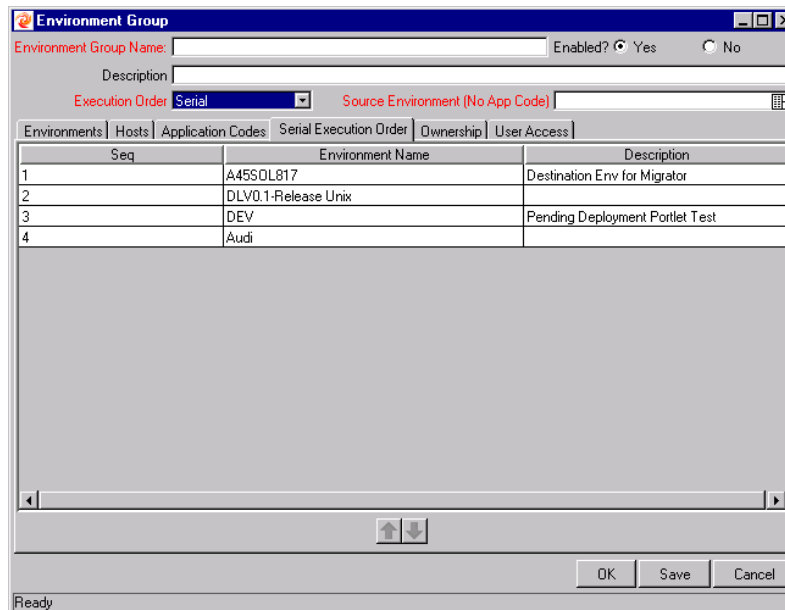
Note

You must be a member of one of the Environment Group's Ownership Groups in order to modify an Environment Group. See "[Setting Ownership for Environment Groups](#)" on page 180 for details.

Setting the Environment Execution Order

To set the Environment execution order:

- Click the **SERIAL APPLICATION ORDER** tab in the ENVIRONMENT GROUP window.



- Select a row containing the Environment to be moved.
- Click the up or down arrow to move the selected Environment to a new sequence position.

The Environments are executed in sequential order until all executions have completed. Each Environment execution waits for the previous Environment execution to complete (success or fail) before beginning.

- Click **OK**.

This saves the changes and closes the window.



Note

You must be a member of one of the Environment Group's Ownership Groups in order to modify an Environment Group. See "[Setting Ownership for Environment Groups](#)" on page 180 for details.

Linking Environments and Environment Groups to Workflows

Environments must be linked to Workflow execution steps that require connection, communication, or transfer between the clients, servers and databases used in your deployment system.

Environments are specified on the WORKFLOW STEP window, accessible from the WORKFLOW window's LAYOUT tab. Select the source and destination Environment or Environment Groups from the fields shown below.

Environments are specified for the execution Workflow steps.

The screenshot shows the 'Workflow Step' dialog box with the following fields and options:

- Step Number: 1
- Step Name: Migrate
- Action Summary: (empty)
- Description: (empty)
- Source Type: Execution
- Source Name: Migrate
- Enabled: Yes No
- Display: Always
- Workflow Parameter: NONE
- Source Environment: (selected, empty text box)
- Source Environment Group: (empty text box)
- Dest Environment: (selected, empty text box)
- Dest Environment Group: (empty text box)
- Avg Lead Time: (empty text box)
- Project Status: (empty text box)
- Parent Assigned To User: (empty text box)
- Parent Assigned To Group: (empty text box)
- Workflow Step Information: (empty text box)

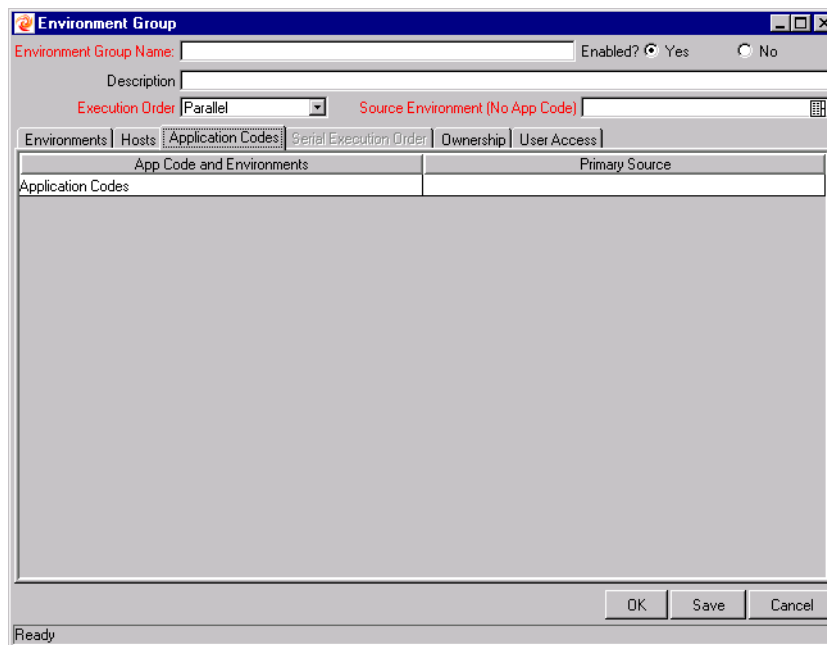
Buttons: OK, Apply, Cancel

Choosing the Source Environment Based on Selected App Code

Environment Groups can be used to dynamically determine the source Environment based on the Application Code for a Package Line. The Application Codes are picked based on the Environments associated with the Environment Groups. All Apps Codes associated with an Environment are inherited by the Environment Group.

To enable the dynamic selection of a Source Environment based on the Application Code:

1. Click the **APPLICATION CODES** tab in the ENVIRONMENT GROUP window.



2. Select the PRIMARY SOURCE ENVIRONMENT for each Application Code.

The Primary Source Environment will automatically be selected as the Source Environment in the Workflow step when the associated Application Code is used in a particular Package Line.



Note

You must be a member of one of the Environment Group's Ownership Groups in order to modify an Environment Group. See [“Setting Ownership for Environment Groups”](#) on page 180 for details.

Environment Maintenance and Utilities

Kintana provides the following functionality to help you validate and maintain your Environment configurations:

- *Testing the Environment Setup*
- *Mass Update of Base Paths*
- *Environment Password Management Utility*
- *Deleting Environments*

Testing the Environment Setup

To check the validity of the Environment:

1. Click **CHECK** at the bottom left of the ENVIRONMENT window.
2. The CHECK ENVIRONMENT window opens.

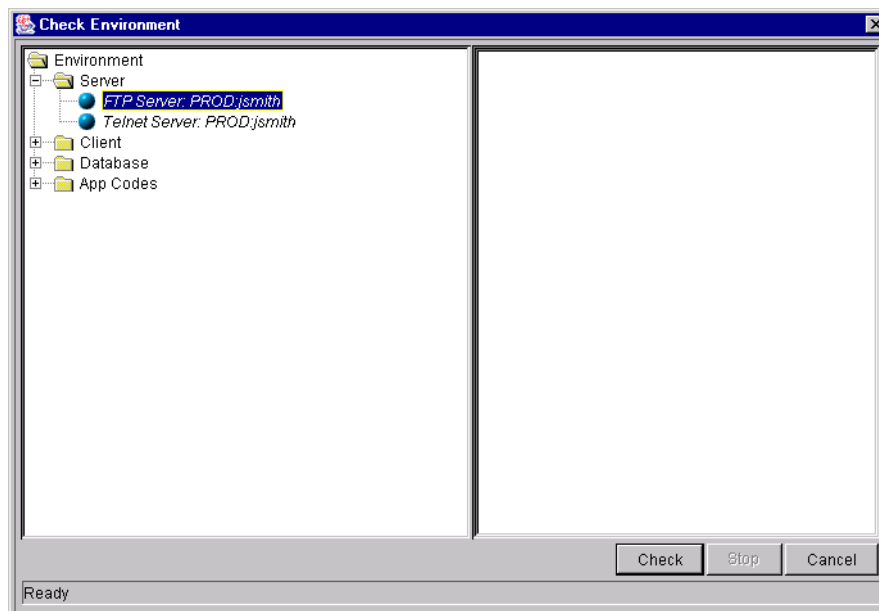


Figure 8-3 Sample Check Environment Window

3. Select the parts of the Environment to be checked by clicking on the directory (Server, Client, Database, etc.).

4. Click **CHECK**.
5. Kintana verifies the Environment definition.

Environment definition testing includes actions performed during regular code migration, such as opening a Telnet session to the server, opening an FTP session to the server, and connecting to the database. While the Environment Checker cannot guarantee that all migrations will be successful, it can help catch some of the most common set-up problems.

While the Check process can take a significant amount of time, it is recommended that any new Environment is checked once all the data for it is entered. Additionally, it is good practice to periodically check all Environments to catch any obvious problems, such as changed passwords or disabled accounts.

Mass Update of Base Paths

It is possible to update server/client base path segments in the Environment and all of its applications at the same time. This functionality is useful, for example, to relocate a particular Environment and all of its applications onto a new disk or partition. To do this:

1. Select **ENVIRONMENT->UPDATE BASE PATHS**. The UPDATE BASE PATH window opens.
2. Enter the old server/client base path segment and the new server/client base path segment in the fields provided. The default value in the old server/client base path field is the Environment's current server/client base path segment.
3. Click **APPLY** or **OK**. Every occurrence of the old server/client base path segment will be replaced by the new server/client base path segment in the Environment and all of its applications.

For example, suppose that two applications have been selected. The server base paths for these two applications are “/u2/apps/isi” and “/u2/apps/demo_107”. To change u2 to u3, enter “u2” in “OLD SERVER BASE PATH” and “u3” in “NEW SERVER BASE PATH” before copying these applications. Every occurrence of the old server/client base path segment will be changed to the new base path segment in all the applications selected. The changes will be reflected in the applications in the Environment into which they have been copied.

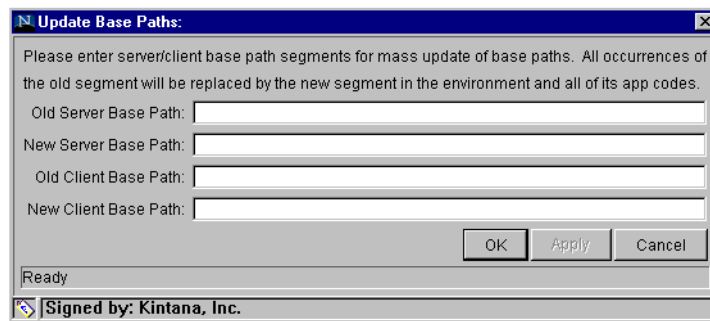


Figure 8-4 Update Base Path Window

Environment Password Management Utility

A single user can have access to multiple password-protected Environments. It is often convenient to use a single username and password for all of the Environments that a single user encounters. If the user decided to change their password or if that user's job functions were transferred to another user, it would take hours to update the Environment passwords in each Environment window.

The Environment Password Management Utility enables a user to update their password in all of the Environments located on a single host, simultaneously. These updates can be made using the Kintana Environment interface or directly at the command line.



Note

This utility will only mass update passwords with matching parameters (Hostname, Username, Old Password, Connect String, etc.).

Updating Passwords Using the Kintana Workbench Interface

To change a user's Environment password(s) using the Kintana interface:

1. Click the **ENVIRONMENTS** screen in the **ENVIRONMENTS** screen group.
2. From the **ENVIRONMENTS** menu, select **UPDATE PASSWORD**. The **UPDATE ENVIRONMENTS PASSWORD** window opens.



3. Select the **HOST TYPE**. The required fields will dynamically change to match requirements of the selected Host Type.
4. Fill in the all of the fields.
5. Click **OK** to implement the changes.

Updating Passwords Using the Command Prompt

To change a user's Environment password(s) using the UNIX Command Line:

1. On the Kintana Server, `cd` to the `bin` directory where the Kintana product suite is installed.
2. Type one of the following commands, depending on the environment:

- For an NT environment:

```
KNTA_Username KNTA_Password Hostname NTDomain Username
Old_PW New_PW |
```

- For a UNIX environment:

```
KNTA_Username KNTA_Password Hostname Username Old_PW New_PW
|
```

- For an Oracle environment:

```
KNTA_Username KNTA_Password Connect_String Username Old_PW
New_PW |
```

- For an MS SQL Server environment:

```
KNTA_Username KNTA_Password DB_Name User_Login Old_PW New_PW
|
```

3. Enter no arguments for user-interface.
4. Follow the remaining command prompts to update the password.

Deleting Environments

To delete an Environment:

1. Open the ENVIRONMENT WORKBENCH window.
2. Enter the search criteria required to select the Environments in the **QUERY** tab and click **LIST**. The **RESULTS** tab opens, displaying the results of the search.
3. Select the Environment to be deleted and click **DELETE**. A dialog box appears asking for confirmation that the Environment is to be deleted.
4. Click **OK** to delete the Environment or **CANCEL** to leave it unchanged.



Note

You must be a member of one of the Environment's Ownership Groups in order to delete an Environment. See *"Setting Ownership for Environments"* on page 659 for details.

Chapter 9

Integrating Participants into Your Deployment System

This chapter provides an overview for how to integrate Kintana users into your deployment process. This chapter illustrates how the information gathered in *“Gathering Process Requirements and Specifications”* on page 49 can be used to define the Security Groups and control users’ access to actions in Kintana.

This chapter discusses the following topics:

- *User Security and Participation - Overview*
- *Mapping the Worksheet to Participant Security*
- *Establishing Security Groups*
- *Setting Package Creation Security*
- *Setting Package Processing Security*
- *Setting Configuration Security*



Note

This chapter presents a number of configuration options available to you. It does not, however, provide detailed instructions on implementing each configuration. See *“Kintana Security Model”* for a comprehensive resource for configuring user access to Kintana screens and features.

User Security and Participation - Overview

Kintana allows you to exercise a great deal of control over your deployment process. You can restrict users’ actions around:

- **Package creation:**
 - Who can create Packages.
 - Who can use a specific Workflow.
 - Who can use specific Object Types.
- **Package processing:**
 - Who can approve / process each step in the Workflow. For this restriction, you can enable access by specifying specific users or security groups. You can also provide access dynamically by having a Kintana Token resolve to provide access.
 - Whether you only want “Participants” to process the Packages. Participants are defined as the Assigned User, the creator of the Package, members of the Assigned Group, or any users who have access to the Workflow step(s).
- **Managing your deployment process:**
 - Who can change the Workflow.
 - Who can change each Object Type.
 - Who can change the Environment definitions.
 - Who can change the Security Group definitions.



Kintana recommends using Security Groups or dynamic access (Tokens) to provide access to Kintana functionality whenever possible. You should avoid specifying a list of users to control an action; for example, specifying a list of users who can act on a Workflow step. If the list of users changes (due to any departmental reorganization), you would have to update that list in many places on the Workflow. By using a Security Group instead of a list of users, you can update the Security Group once, and the changes are propagated throughout the Workflow steps.

Mapping the Worksheet to Participant Security

Using the information gathered in “*Gathering Process Requirements and Specifications*” on page 49 you can begin to configure your Security Groups. Use the following as a guide to configuration instructions related to the Security worksheet shown below.

Participant and Security

Table D-8. Security Groups.

Security Group Name	Members	Act on Workflow Steps	Description

Establishing Security Groups

Security Groups are used in Kintana to control who can access certain screens and functionality in Kintana. See “*Security Groups*” on page 30 for an overview. The following sections provide instructions on defining Security Groups:

- [Creating a Security Group by Specifying a List of Users](#)
- [Using Kintana's Resource Management to Control User Security](#)

The general process for creating a Security Group is as follows:

1. Specify Security Group membership on the **USERS** tab. This can be accomplished by providing a list of users or by associating the group with an organization unit defined in Kintana.
2. Specify the screen and feature access by linking the appropriate Access Grants. See "[Kintana Security Model](#)" for details.
3. Specify which Workflows users in this Security Group can use when deploying changes. This is set in the **DELIVER SETTINGS** tab.
4. Restrict the Security Group from using certain Application Codes when creating a Package Line. This restricts which applications each user can process objects through.

Note

You should consider creating and maintaining two types of Security Groups:

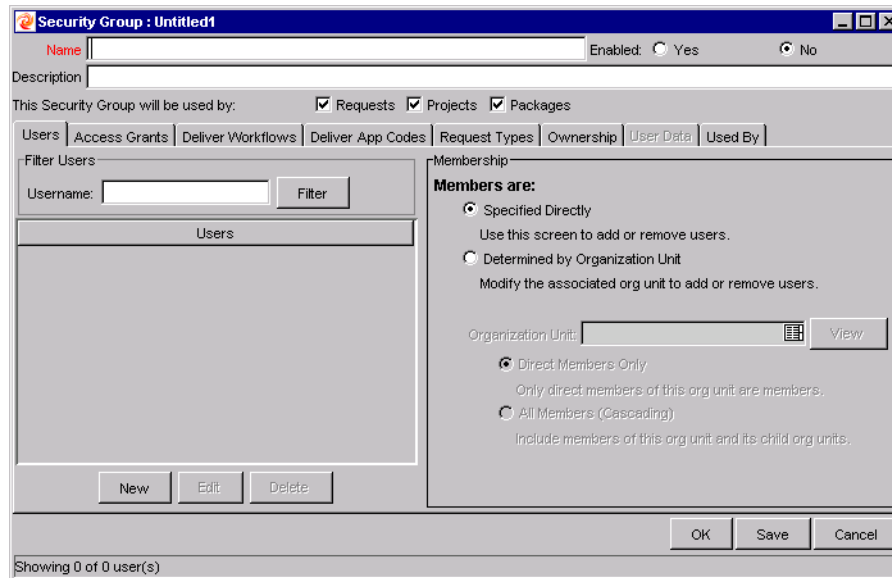
- Security Groups to control who can act on a specific Workflow step (list of users without any special access grants)
- Security Groups to control who can access a particular screen or function (list of users and appropriate access grants)

This can greatly simplify your maintenance of a security model around Kintana processes. As new users are added to the system, they can be granted to appropriate screen and function access and associated with specific Workflows.

Creating a Security Group by Specifying a List of Users

To generate and define a new Security Group:

1. Click **NEW SECURITY GROUP** in the SECURITY GROUP WORKBENCH or select **FILE > NEW > SECURITY GROUP** from the menu. The SECURITY GROUP window opens.



2. Enter the NAME and DESCRIPTION.
3. Select **YES** to enable this Security Group.

If the Security Group is not enabled, it does not appear as a choice when generating or updating users or Workflows.

4. Select which Kintana entities (Requests, Projects or Packages) will use the Security Group by clicking their respective check boxes in the THIS SECURITY GROUP WILL BE USED BY field.
5. Link the desired Users to the Security Group.
 - a. Click **NEW** in the **USERS** tab. The **USERS** window opens.
 - b. Enter the desired username into the **USERS** field and click **ADD** or click on the **USERS** auto-complete list to display all available users. The **VALIDATE** window opens.
 - c. Select the desired **USER NAME**.
 - d. Click **OK**. The **VALIDATE** window closes.
 - e. Click **OK** to add your selection to the **USERS** tab.
6. Link the desired Access Grants. Each Access Grant enables certain functions performed on a Kintana screen. See "*Kintana Security Model*" for a description of each available access grants.

- a. Select the desired Access Grants in the AVAILABLE ACCESS GRANTS list.
 - b. Click the right arrow button pointing to the LINKED ACCESS GRANTS list. The selected Access Grants are moved into the column.
7. Restrict the Security Group from using certain Workflows.
 - a. Click the **DELIVER WORKFLOWS** tab.
 - b. Select the Workflows in the ALLOWED DELIVER WORKFLOWS list.
 - c. Click the left arrow button pointing to the RESTRICTED DELIVER WORKFLOWS list. The selected Workflows are moved into the column.
 - d. If all future Workflows should also be excluded, select the ALWAYS RESTRICT NEW WORKFLOWS check box.
8. Restrict the Security Group from using certain Application Codes when creating a Package Line. This restricts which applications each user can process objects through.
 - a. Click the **DELIVER APP CODES** tab.
 - b. Select the App Codes in the ALLOWED DELIVER APP Codes list.
 - c. Click the left arrow button pointing to the RESTRICTED DELIVER APP CODES list. If all future App Codes are to be excluded, select the **ALWAYS RESTRICT NEW APP CODES** check box.
9. Optional: If you plan on using the same Security Group for processing Requests in Kintana Create, specify which Request Types the Security Group can use in the **REQUEST TYPES** tab.

See "[Kintana Security Model](#)" for more information about allowing/restricting Request Types in Security Groups.
10. Click the **OWNERSHIP** tab and select the Ownership Groups that have the right to edit, copy or delete the current Security Group. See "[Setting Ownership for Security Groups](#)" on page 92 for more information about setting Ownership for a new or existing Security Group.
11. (Optional) Enter any necessary information in the **USER DATA** tab's custom fields.

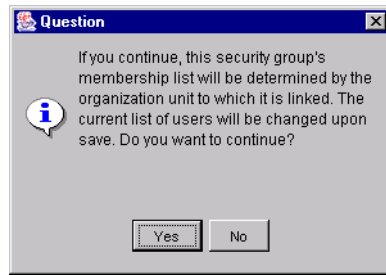
12. Click **OK** to register the current Security Group and close the SECURITY GROUP window. Click **SAVE** to save the information and leave the SECURITY GROUP window open.

Using Kintana's Resource Management to Control User Security

Users can also be associated to Security Groups through their inclusion in an organization model definition. Using Kintana's resource management capabilities, a user can be placed into a model that includes security and access information. See "[Managing Resources in Kintana](#)" for details.

To define a Security Group to use the members of an organization unit:

1. Open the SECURITY GROUP window.
2. Select **DETERMINED BY ORGANIZATION UNIT** in the **MEMBERSHIP** section of the **USERS** tab. The following question dialog opens.



3. Click **YES**.



Note

When you select an Organization Unit to control user access to the Security Group, any users specified in the Users list will be replaced with the members of the organization unit.

4. Select the ORGANIZATION UNIT.

5. Select whether you want to include:

- **Direct Members Only:**
Only direct members of the specified organization unit.
- **All Members (cascading)**
Members of this organization unit and its child units.

6. Click **SAVE**.

Related Topics:

- ["Managing Resources in Kintana"](#)
- ["Kintana Security Model"](#)

Setting Package Creation Security

You can control who can create certain Packages or use specific Object Types and Workflows. This provides a great deal of control over who can process changes of a certain type to specific environments. The following sections discuss how to control security related to Package creation:

- [Enabling Users to Create Packages](#)

- *Restricting Users from Selecting a Specific Workflow*
- *Restricting Users from Selecting a Specific Object Type*

Enabling Users to Create Packages

You can control which Kintana users have the ability to create and submit Packages. To enable a user to create and submit a Package, ensure that the following are set.

Table 9-1. Settings required to enable a user to create Packages in Kintana

Setting	Value	Description
License	Kintana Deliver: Power License	The Power License provides a Kintana user with access to the Kintana Workbench, where the Package is defined. This is set in the USER window on the KINTANA WORKBENCH.
Access Grants linked to the Security Group	Deliver: Edit Packages	This Access Grant allows the user to generate, edit and delete certain Packages. <ul style="list-style-type: none"> • User cannot delete a Package if it has been released or if user is not the owner. • To edit the Package, user must be its creator, the 'assigned to' user, a member of the assigned group or a member of the Workflow Steps security group. Access Grants are set in the SECURITY GROUP window.
	Deliver: Manage Packages	This Access Grant allows the user to create, edit and delete Packages at anytime. Access Grants are set in the SECURITY GROUP window.

Table 9-1. Settings required to enable a user to create Packages in Kintana

Setting	Value	Description
Allowed Deliver Workflows in the Security Group window	You must have at least one Workflow allowed.	When creating a Package, you are required to select a Workflow for the Package to proceed through. At least one Workflow must be enabled to be able to create and submit a Package. The user should select the Workflow intended to process the deploying objects. This is set on the SECURITY GROUP window - DELIVER WORKFLOWS tab.
Allowed Deliver Object Types in the Workflow window.	You must allow at least one Object Type in each Workflow used to deploy changes.	You can associate Object Types with Workflows such that only certain Object Types can be processed through the Workflow. At least one Object Type must be enabled so that the user can create a Package Line when using that Workflow. This is set on the WORKFLOW window - DELIVER SETTINGS tab, under the PACKAGE LINE selection.



Note

Screen and function access provided through Access Grants are cumulative. If a user belongs to three different Security Groups, he will have all access provided to each of the groups. Therefore, to restrict certain screen and feature access, you need to remove the user from any Security Group that grants that access.

You can use the **ACCESS GRANTS** tabs in the USER window to see all Security Groups where specific access grants are included. You can then:

- Remove the user from the Security Group (using the **SECURITY GROUP** tab on the USER window)
- Remove the Access Grants from the Security Group (in the Security Group window). Note: you should only do this if no one in that Security Group needs the access provided in that Access Grant.

Restricting Users from Selecting a Specific Workflow

You can restrict users from selecting specific Workflows when creating a new Package. To do this, ensure that the following conditions are met.

Table 9-2. Settings required to restrict Workflow selection

Setting	Value	Description
Restricted Deliver Workflows in the Security Group window	Include the Workflows that you would like to restrict.	<p>When creating a Package, you are required to select a Workflow for the Package to proceed through. Users (in the Security Group) will not be able to select any Workflows included in the Restricted Deliver Workflows list.</p> <p>Note: If a user belongs to another Security Group that allows the use of that Workflow, the user will be able to select it.</p> <p>This is set on the SECURITY GROUP window - DELIVER WORKFLOWS tab.</p>



Restricting the Workflow selection also controls who can deploy changes to specific environments, because the source and destination Environments are defined in the Workflow step.

Restricting Users from Selecting a Specific Object Type

You can restrict users from selecting specific Object Types when creating a new Package. To do this, ensure that the following conditions are met.

Table 9-3. Settings required to restrict Object Type selection

Setting	Value	Description
Restricted Deliver Object Types in the Workflow window.	Include the Object Type that you would like to restrict.	<p>You can associate Object Types with Workflows such that only certain Object Types can be processed through the Workflow. Users (in the Security Group) will not be able to select any Object Types included in the RESTRICTED DELIVER WORKFLOWS list.</p> <p>This is set on the WORKFLOW window - DELIVER SETTINGS tab, under the PACKAGE LINE selection.</p>

Setting Package Processing Security

You can control who can process Packages following a Package submission. You can also control who can act on certain steps (decisions and executions) in your process. The following sections discuss how to control security related to Package processing:

- [Providing Users with General Access to Update Packages](#)
- [Enabling Users to Act on a Specific Workflow Step](#)
- [Restricting Package Processing to Participants](#)

Providing Users with General Access to Update Packages

All users who will be processing Packages must meet the following conditions:

Table 9-4. Settings required to enable a user to process Packages in Kintana

Setting	Value	Description
License (at least one is required)	Kintana Deliver: Power License	The Power License provides a Kintana user with access to the Kintana Workbench. Users can act on all Workflow steps (decisions and executions) in the Workbench. This is set in the USER window on the KINTANA WORKBENCH.
	Kintana Deliver: Standard	The Standard License provides a Kintana user with access to the Kintana standard interface. Users can act on all decision Workflow steps. Note: you must have a Power Licence to process execution steps. This is set in the USER window on the KINTANA WORKBENCH.

Table 9-4. Settings required to enable a user to process Packages in Kintana

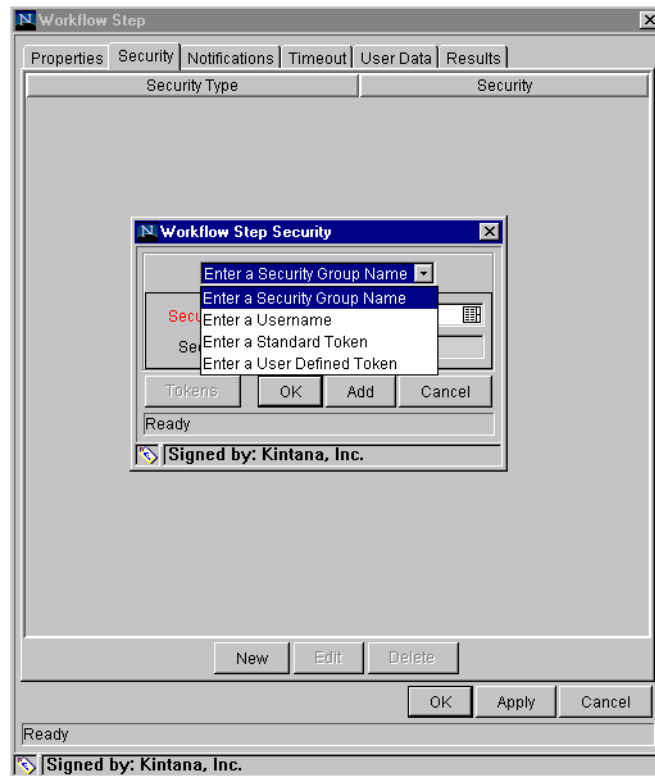
Setting	Value	Description
Access Grants linked to the Security Group	Deliver: Edit Packages	<p>This Access Grant allows the user to generate, edit and Packages.</p> <ul style="list-style-type: none"> • User cannot delete a Package if it has been released or if user is not the owner. • To edit the Package, user must be its creator, the 'assigned to' user, a member of the assigned group or a member of the Workflow Steps security group. <p>Access Grants are set in the SECURITY GROUP window.</p>
	Deliver: Manage Packages	<p>This Access Grant allows the user to edit or delete Packages at anytime.</p> <p>Access Grants are set in the SECURITY GROUP window.</p>

Enabling Users to Act on a Specific Workflow Step

You need to specify who can act on each step in the deployment Workflow. Only people who are specified on the **SECURITY** tab in the **WORKFLOW STEP** window will be able to process Packages and Package Lines at that step.

To specify the users who can act on a specific Workflow step:

1. Open the Workflow.
2. Click the **LAYOUT** tab.
3. Double click on the step that you would like to configure. The **WORKFLOW STEP** window opens. Note: the **WORKFLOW STEP** window also opens when first adding a step to the **LAYOUT** tab.
4. Click the **SECURITY** tab.
5. Click **NEW**. The **WORKFLOW STEP SECURITY** window opens.



6. Select the method for specifying the step security from the drop down list: Security Group Name, Username, Standard Token, User Defined Token. Selecting a value from this field automatically updates the other fields on this window. For example, selecting **ENTER A USERNAME** will change the SECURITY GROUP field to USERNAME.
7. Specify the Security Groups, Usernames, or Tokens that will control the access to this step.
8. Click **OK**. The security specification is added to the **SECURITY** tab. You can add additional specifications to the step by clicking **NEW** and repeating the above process. You can therefore select to control the step's security using a combination of multiple Security Groups, Usernames and Tokens.
9. Click **OK** to save and close the window.



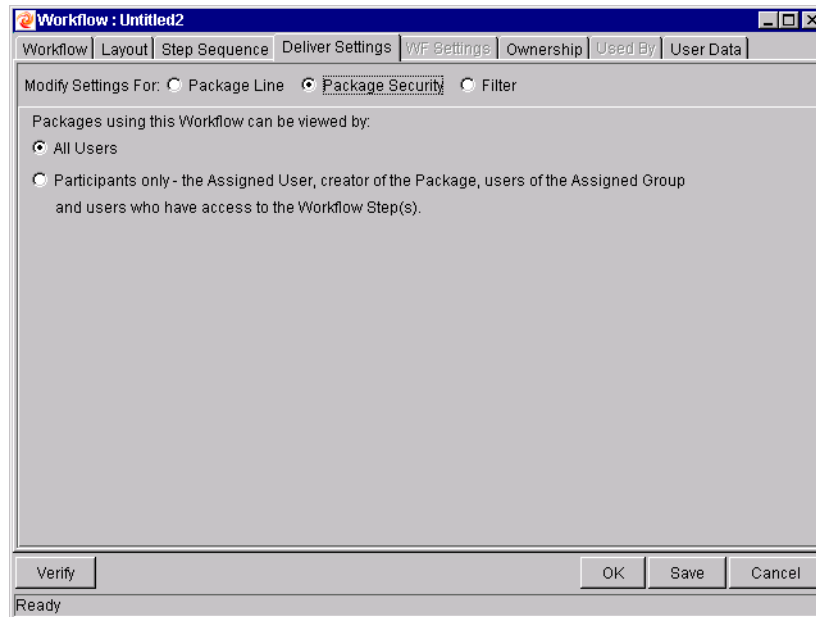
1. Consider assigning a Security Group to each decision, execution and condition step, even though many of these steps will proceed automatically. If a command fails, or a condition is not met, you may need to manually override the step.
2. You may also want to consider assigning a “Deployment Manager” Security Group to each step. That group could be configured with global access to act on every step in the process. Again, this could help avoid bottlenecks by providing a small group with permission to process stalled Packages.
3. You may want to avoid specifying a single user as the only person who can act on a Workflow step. This would require a process update (re-configuration) when that user changes roles or leaves the company. Better to grant access dynamically using a Token or Security Group.

Restricting Package Processing to Participants

The **PACKAGE SECURITY** tab in the **WORKFLOW** window lets you determine who can have access to Packages that use the current Workflow. Restricting access to Participants means that when non-Participant users search for Packages, they will not see a Package that uses the current Workflow. In this instance, Participants are defined as:

- The Assigned User
- The creator of the Package
- Members of the Assigned Group
- Any users who have access to the Workflow Step(s)

To let all Kintana users access Packages using the current Workflow, select **ALL USERS**.



To restrict the number of Kintana users who can access Packages using the current Workflow to Participants of the Packages, select **PARTICIPANTS ONLY - THE ASSIGNED USER, CREATOR OF THE PACKAGE, USERS OF THIS ASSIGNED GROUP, AND USERS WHO HAVE ACCESS TO THE WORKFLOW STEP(S)**.

Setting Configuration Security

A critical part of ensuring successful deployments is ensuring that your deployment process is altered only by the correct people. Kintana allows you to set security around the Kintana configuration. You can establish configuration security around all of the Kintana configuration entities. This includes such activities as controlling:

- Who can change the Workflow.
- Who can change each Object Type.
- Who can change the Environment definitions.
- Who can change the Security Group definitions.

The following sections discuss some options for securing your Kintana configurations:

- [Setting Ownership for Kintana Configuration Entities](#)
- [Removing Access Grants](#)

Setting Ownership for Kintana Configuration Entities

Different groups of Kintana users have ownership and control over Kintana entities. These groups are referred to as Ownership Groups. Unless a 'global' permission has been designated to all users for an entity, members of Ownership Groups are the only users who have the right to edit, delete or copy that entity. The Ownership Groups must also have the proper access grant for the entity in order to complete those tasks. For example, the `EDIT WORKFLOWS` Access Grant is needed to edit Workflows and Workflow Steps.

You can assign multiple Ownership Groups to the various entities. Ownership Groups are defined in the `SECURITY GROUP` window. Security Groups become Ownership Groups when used in the Ownership capacity.

You can select to specify Ownership Groups for the following entities involved in your deployment process:

- Workflows
- Workflow Steps
- Environments
- Environment Groups
- Object Types
- Security Groups
- User Definitions
- Report Types
- Validations
- Special Commands

The Ownership setting is accessed through the individual entity windows in the Kintana Workbench. For example, to set the Ownership for Workflows:

1. Open the Workflow.
2. Click the **OWNERSHIP** tab.

3. Click the **ONLY GROUPS LISTED BELOW THAT HAVE THE EDIT WORKFLOWS ACCESS GRANT** radio button.
4. Click **ADD**. The **ADD SECURITY GROUP** window opens.
5. Select the **SECURITY GROUP**.
6. Click **ADD** to add the current Security Group and continue adding more Security Groups. Click **OK** to add the current Security Group and close the **ADD SECURITY GROUP** window.

The Security Group(s) you selected displays in the **OWNERSHIP** tab under the **SECURITY GROUP** column.

Workflow : Untitled5

Workflow | Layout | Step Sequence | Deliver Settings

Package Workflows | Request Types | Ownership | Used By | User Data

Give ability to edit this Workflow to:

All users with the Edit Workflows Access Grant

Only groups listed below that have the Edit Workflows Access Grant

Security Group	Description
Kintana Deliver Config Manager	Configuration Manager for Kintana Deliver

Add Remove

Verify OK Save Cancel

Ready

7. Click **OK** to save the selection and close the **WORKFLOW** window. Click **SAVE** to save the selection and leave the **Workflow** window open.



Note

The **SYS ADMIN: OWNERSHIP OVERRIDE** access grant allows the user to access and edit configuration entities even if he is not a member of one of the entity's Ownership Groups.

Removing Access Grants

You can also restrict the ability to modify Kintana configuration entities by removing the user from any Security Group that grants that access.

You can use the **ACCESS GRANTS** tabs in the USER window to see all Security Groups where specific access grants are included. You can then either:

- Remove the user from the Security Group (using the **SECURITY GROUP** tab on the USER window)
- Remove the Access Grants from the Security Group (in the **SECURITY GROUP** window). Note: You should only do this if no one in that Security Group needs the access provided in that Access Grant.

The following table lists the access grants that provide edit access to different Kintana configuration entities.

Table 9-5. Access Grants for editing Kintana configuration entities

Access Grant	Description
Config: Edit Workflows	Allows the user to generate, update and delete Workflows.
Config: Edit Report Types	Allows the user to generate, update and delete Report Types.
Config: Edit Special Commands	Allows the user to generate, update and delete Special Commands.
Config: Edit User Data	Allows the user to generate, update and delete User Data.
Config: Edit Validation Values	Allows the user to generate, update and delete Validation Values.
Config: Edit Validations	Allows the user to generate, update and delete Validations.
Config: Edit Workflows	Allows the user to generate, update and delete Workflows.
Deliver: Edit Object Types	Allows the user to generate, update and delete Object Types.
Environments: Edit Environments	Allows the user to generate, update and delete Environments.
Sys Admin: Edit Security Groups	Allows the user to generate, update and delete Security Groups.

Table 9-5. Access Grants for editing Kintana configuration entities

Access Grant	Description
Sys Admin: Edit Users	Users Allows the user to generate, update and delete Users.

Chapter 10

Setting Up Communication Paths

This chapter provides an overview for different modes of communication that you can use in your deployment system. Kintana features three main devices for communicating status related to your deployment process:

- **Email Notifications:**
Each Workflow step can be configured to send an email to specified users when the step becomes eligible, has a specific result, or encounters an error. Using notifications at key points in your process ensures a speedy deployment by notifying appropriate parties of actions required by them or complications during deployment.
- **Kintana Dashboard:**
The Dashboard provides an interface through which you can quickly assess the current state of the deployments. Personalize your Dashboard to display status information that is most meaningful to your role. For example, software engineers may only want to see the Packages that they submitted, whereas the IT manager may want to have visibility into each critical Package currently in progress.
- **Reports:**
Kintana includes a number of reports that can be used to assess deployment status. Kintana also publishes a reporting meta layer that you can use to build your own custom reports. Kintana Reports can be scheduled to run periodically.

This chapter illustrates how the information gathered in *“Gathering Process Requirements and Specifications”* on page 49 can be used to define the notifications, Dashboard components, and reports used to monitor and control the deployment process. This chapter discusses the following topics:

- *Adding Notifications to Workflow Steps*
- *Configuring Your Dashboard*
- *Configuring Reports*

Adding Notifications to Workflow Steps

When configuring a Notification for a Workflow step, you need to consider the following:

- When to send it.
- Who should receive it.
- What the message should say.

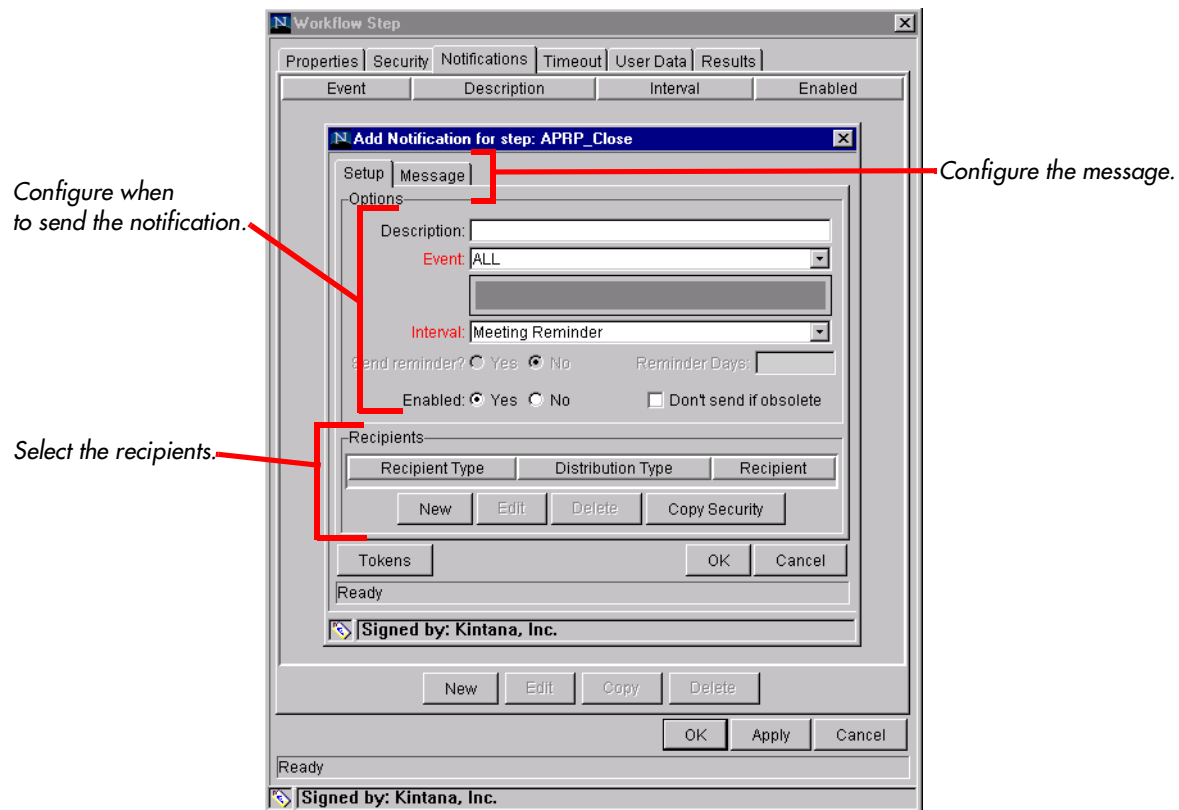
The following sections discuss different notifications techniques. See "[Configuration Workbench Reference](#)" for details on each field in the Notification interface.

- [Adding a Notification to a Workflow step - Overview](#)
- [Configuring When to Send a Notification](#)
- [Configuring the Notification Recipients](#)
- [Configuring the Notification Message](#)

Adding a Notification to a Workflow step - Overview

To add a notification to a Workflow step:

1. Open the Workflow.
2. Click the **LAYOUT** tab.
3. Double click on the step that you would like to configure. The **WORKFLOW STEP** window opens. Note: the **WORKFLOW STEP** window also opens when first adding a step to the **LAYOUT** tab.
4. Click the **NOTIFICATIONS** tab.
5. Click **NEW**. The **ADD NOTIFICATION FOR STEP** window opens.



6. Configure the following:

- When the notification is sent (EVENT and INTERVAL)
- Who receives the notification (RECIPIENTS)
- The body of the notification (MESSAGE)

7. Click **OK**. The notification specification is added to the **NOTIFICATIONS** tab. You can add additional specifications to the step by clicking **NEW** and repeating the above process. You can therefore select to send a different notification to different recipients for different events.

8. Click **OK** to save and close the window.

Configuring When to Send a Notification

Each Workflow step can be configured to send an email when the step becomes eligible, has a specific result, or encounters an error. The following topics are discussed:

- *Sending a notification when a step becomes eligible*
- *Sending a notification when a step has a specific result*
- *Sending a notification when the step has a specific error*
- *Configuring multiple notifications for a single step*
- *Specifying the Time the Notification is Sent*

Sending a notification when a step becomes eligible

To send a notification when a Workflow step becomes eligible, configure the notification as indicated below.

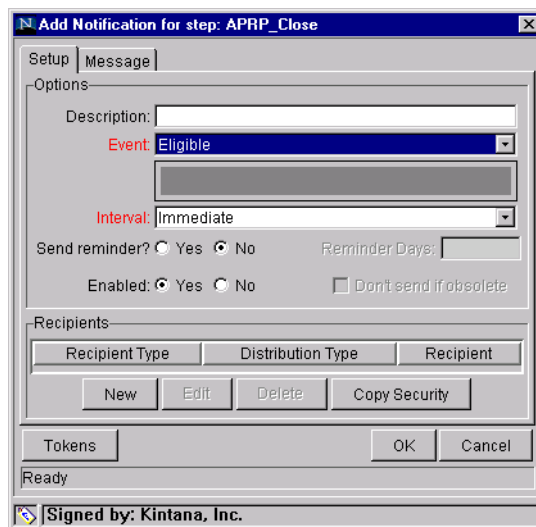


Table 10-1. Workflow step notification configuration - send on eligible

Field	Value	Notes
Event	Eligible	

Table 10-1. Workflow step notification configuration - send on eligible

Field	Value	Notes
Interval	Immediate	<p>You can select to send the notification at different intervals. For example, you might choose to send a notification of a final approval step at midnight so that it's ready for approval in the morning.</p> <p>Note also that multiple notifications to a single recipient can be batched and sent together. Selecting an interval other than "Immediate" will allow this batching to occur.</p> <p>See "Configuring the Notification Intervals" on page 225 for instructions on configuring this.</p>
Send Reminder	Yes/No	<p>This field is optional. A reminder notification can be sent if the notification event is still true after a period of time. For example, a reminder can be sent if a step is still Eligible after a number of days. A reminder cannot be sent if the notification event is 'All.'</p>
Enabled	Yes	

Sending a notification when a step has a specific result

You can configure the notification to be sent when a Workflow step results in a specific decision or execution result. The value for these events is determined by the Workflow step source's validation.

To send a notification when a Workflow has a specific result, configure the notification as indicated below.

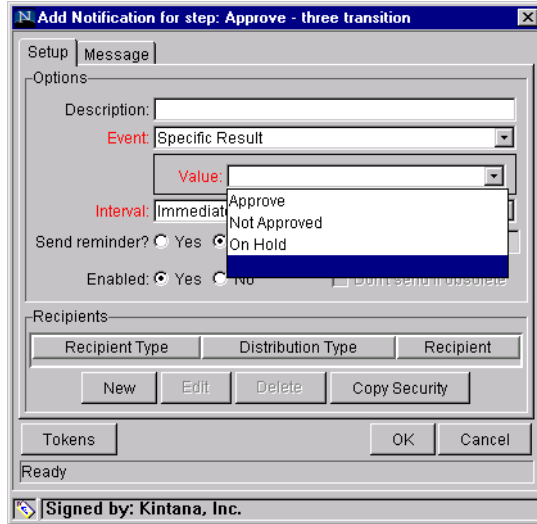


Table 10-2. Workflow step notification configuration - send on step result

Field	Value	Notes
Event	Specific Result	
Value	Select the value to trigger the notification.	The list of values is determined by the Workflow step source's validation. Therefore, this selection will always be limited to the possible results of the step.
Interval	Immediate	<p>You can select to send the notification at different intervals. For example, you might choose to send a notification of a final approval step at midnight so that it's ready for approval in the morning.</p> <p>Note also that multiple notifications to a single recipient can be batched and sent together. Selecting an interval other than "Immediate" will allow this batching to occur.</p> <p>See <i>"Configuring the Notification Intervals"</i> on page 225 for instructions on configuring this.</p>

Table 10-2. Workflow step notification configuration - send on step result

Field	Value	Notes
Send Reminder	Yes/No	This field is optional. A reminder notification can be sent if the notification event is still true after a period of time. For example, a reminder can be sent if a step is still Eligible after a number of days. A reminder cannot be sent if the notification event is 'All.'
Enabled	Yes	

Sending a notification when the step has a specific error

To send a notification when a Workflow has a specific error, configure the notification as indicated below.

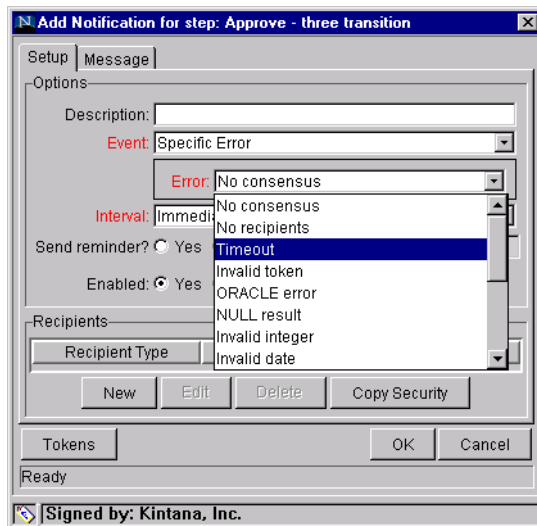


Table 10-3. Workflow step notification configuration - send on error

Field	Value	Notes
Event	Specific Error	
Error	Select the value to trigger the notification.	This is a standard set of errors. See “Specific Errors for Workflow Steps” on page 222.

Table 10-3. Workflow step notification configuration - send on error

Field	Value	Notes
Interval	Immediate	<p>You can select to send the notification at different intervals. For example, you might choose to send a notification of a final approval step at midnight so that it's ready for approval in the morning.</p> <p>Note also that multiple notifications to a single recipient can be batched and sent together. Selecting an interval other than "Immediate" will allow this batching to occur.</p> <p>See "Configuring the Notification Intervals" on page 225 for instructions on configuring this.</p>
Send Reminder	Yes/No	<p>This field is optional. A reminder notification can be sent if the notification event is still true after a period of time. For example, a reminder can be sent if a step is still Eligible after a number of days. A reminder cannot be sent if the notification event is 'All.'</p>
Enabled	Yes	

Specific Errors for Workflow Steps

The following errors can cause a notification to be sent.

Table 10-4. Specific Errors for Workflow Steps

Specific Error	Meaning
NO CONSENSUS	When all users of all Security Groups, or users linked to the Workflow Step need to vote, and there is no consensus.
NO RECIPIENTS	When none of the Security Groups linked to the Workflow Step has users linked to it, no user can act on the Workflow Step.
TIMEOUT	When the Workflow Step times out. Used for Executions and Decisions.
INVALID TOKEN	Invalid Token used in the execution.

Table 10-4. Specific Errors for Workflow Steps

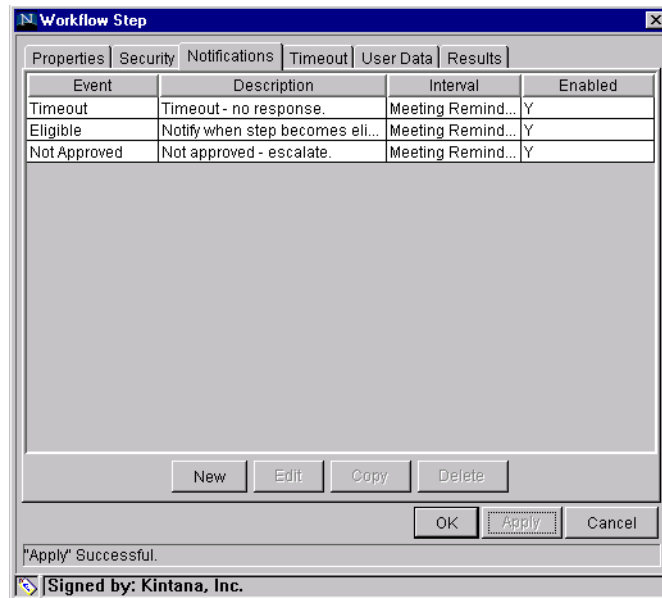
Specific Error	Meaning
ORACLE ERROR	Failed PL/SQL Execution.
NULL RESULT	No result is returned from the execution.
INVALID INTEGER	Validation includes an invalid value in the Integer field.
INVALID DATE	Validation includes an invalid value in the Date field.
COMMAND EXECUTION ERROR	Execution engine has failed or has a problem.
INVALID RESULT	Execution or Subworkflow has returned a result not included in the Validation.
PARENT CLOSED	For wf_receive or wf_jump steps, a Package Line is expecting a message from a Request that is cancelled or closed.
CHILD CLOSED	For wf_receive or wf_jump steps, a Request is expecting a message from a Package Line that is cancelled or closed.
NO PARENT	For wf_receive or wf_jump steps, a Package Line is expecting a message from a Request that has been deleted.
NO CHILD	For wf_receive or wf_jump steps, a Request is expecting a message from a Package Line that has been deleted.
MULTIPLE JUMP RESULTS	For wf_jump steps in a Package Line, different result values were used to transition to the step.
MULTIPLE RETURN RESULTS	When the Package Level subworkflow receives multiple results from Package Lines that traversed through it.

Configuring multiple notifications for a single step

You can configure multiple notifications for each Workflow step. This can be useful in the following sample situations:

- Sending a different message depending on the result of the step
- Sending a different message depending on the type error
- Sending the notification to a different set of users depending on the step's result or error
- Specifying different intervals or reminders based on the type of step error

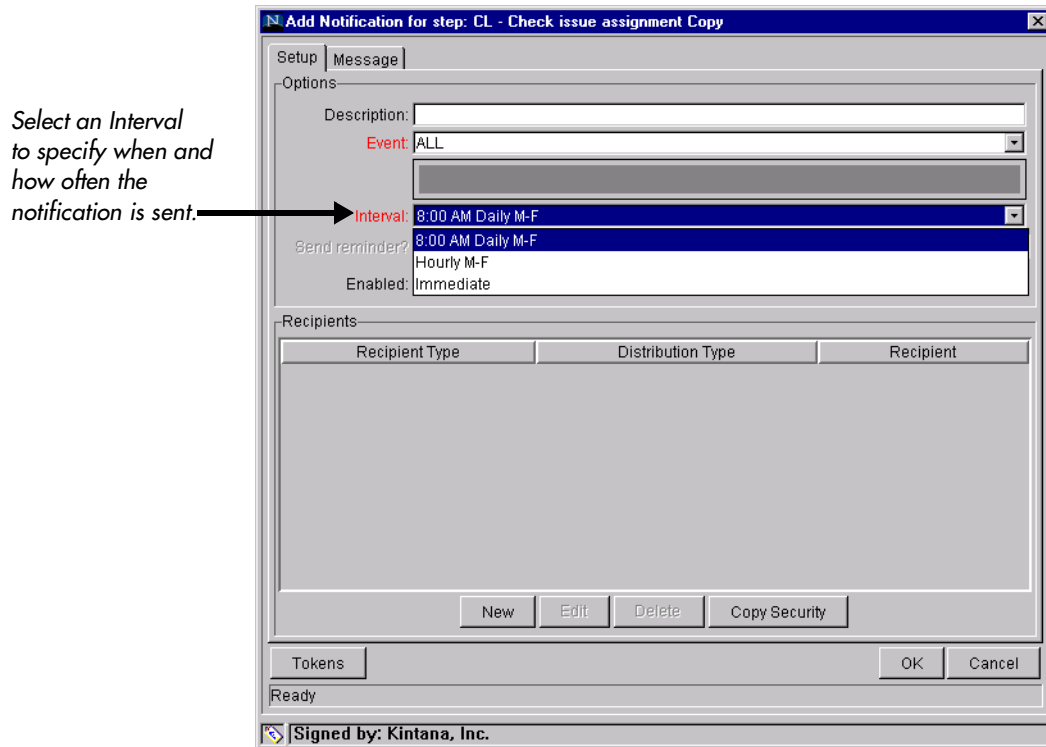
To configure multiple notifications for a Workflow step, simply add multiple notifications to the same Workflow step window.



In this example, one set of users is notified when the step becomes eligible. Then, depending on the outcome of the step, different groups are notified. If the step experiences a “TIMEOUT” error event, then the user responsible for acting on the step is notified. If the step results in the specific result of “NOT APPROVED,” then a notification is sent to the deployment manager.

Specifying the Time the Notification is Sent

Use the INTERVAL field on the Workflow step to specify when the notification will be sent.



The interval determines when the notification will be sent. Kintana provides the following pre-configured intervals:

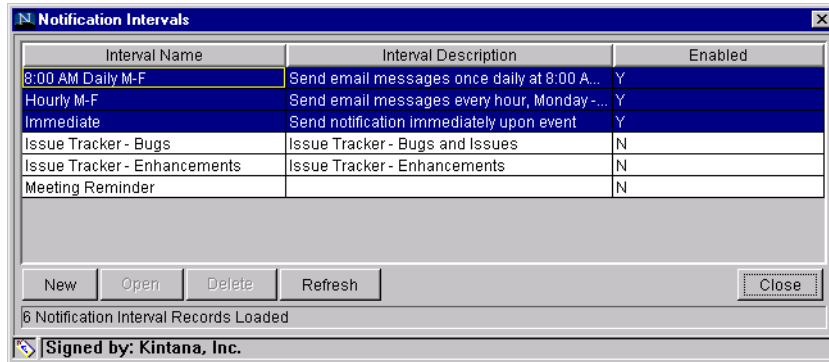
- **8:00 AM DAILY M-F:** This notification is sent at 8:00 AM on the next available work day after the notification event occurs.
- **HOURLY M-F:** This notification is sent on the hour, starting on the next available work day after the notification event occurs.
- **IMMEDIATE:** This notification is sent immediately.

Configuring the Notification Intervals

Notifications are configured on the NOTIFICATION TEMPLATES WORKBENCH. To configure the Notification Intervals:

1. Click the **CONFIGURATION** screen group and click the **NOTIFICATION TEMPLATES** icon.

2. Select **NOTIFICATION TEMPLATES ->INTERVALS** from the menu. The NOTIFICATION INTERVALS window opens.



3. Click **NEW** or **OPEN** to access the NOTIFICATION INTERVAL: NEW window. Enter the required information on the **INTERVAL** tab. These fields are defined in [Table 10-5](#).

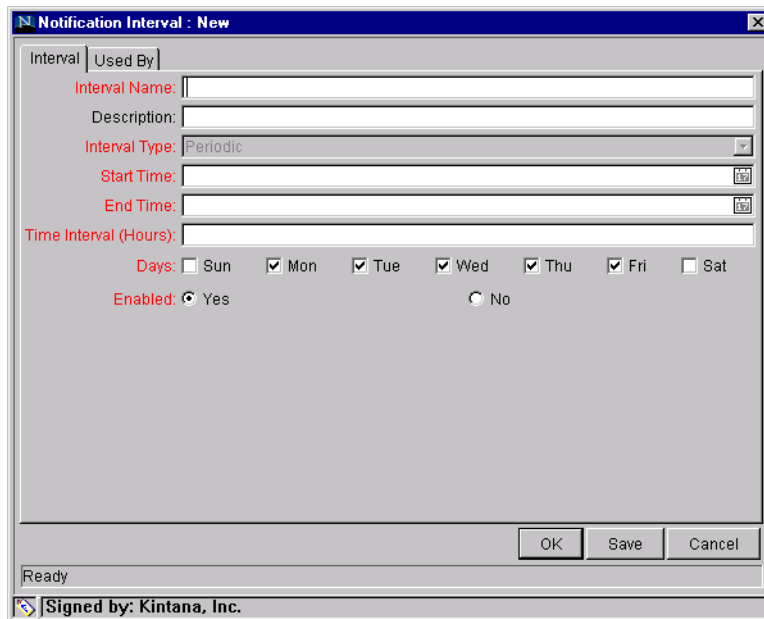


Table 10-5. Notification Intervals

Field Name	Description
Interval Name	This is the name assigned to the interval.
Description	Free form description of this interval.

Table 10-5. Notification Intervals

Field Name	Description
Interval Type	For internal use. This is always set to Periodic, unless Immediate Interval is used.
Start Time	Time to start sending out notifications and to start counting down the time interval until the next batch.
End Time	Time to stop sending out notifications.
Time Interval	Number of hours to wait after the Start Time or the last batch sent, before sending out the next batch of notifications.
Days	Used to select which days this interval should execute on.
Enabled	If Yes is set, this interval is selectable. If No is set, this interval is unavailable.

4. Click **OK**. The new interval is added to the NOTIFICATION INTERVALS window.
5. Click **CLOSE** to close the window.

The new Notification Interval can now be used in any Workflow step notification.

When notifications are sent with an hourly or daily interval, there are sometimes several notifications pending for a particular user. In this case, all of the notifications are grouped together in one email message. The Subject of each individual notification appears at the top of the email message in a Summary section.

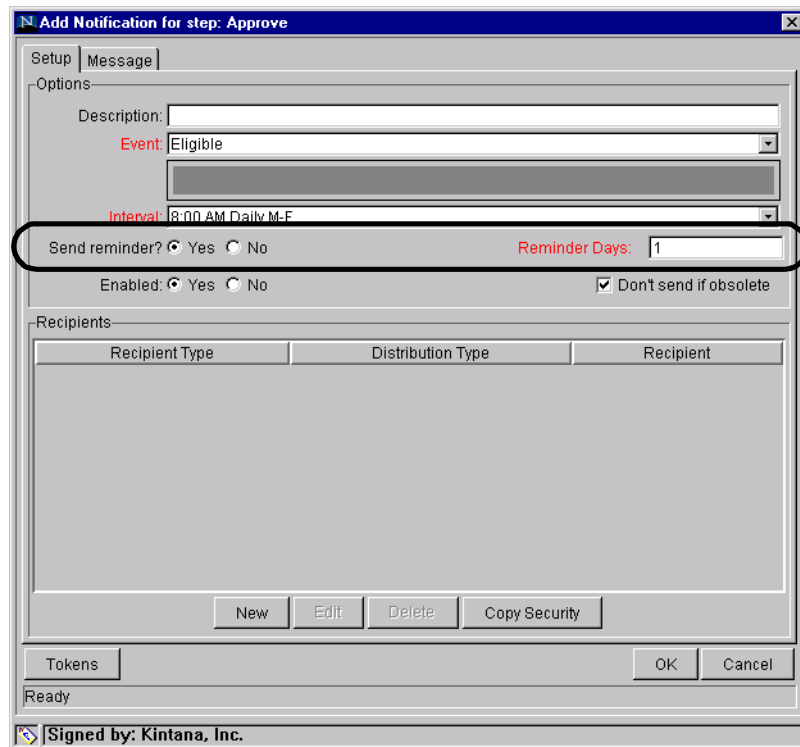
Sending a follow up notification (reminder)

A reminder notification can be sent if the notification event is still true after a period of time. For example, a reminder can be sent if a step is still Eligible after a number of days. A reminder cannot be sent if the notification event is **ALL**.

To configure a notification to re-send after a period of time, configure the notification as indicated below.

Table 10-6. Workflow step notification configuration - send on error

Field	Value	Notes
Event		You can select any event except for ALL .
Send Reminder	Yes	Selecting Yes enables the REMINDER DAYS field.
Reminder Days	Enter the number of days.	The number of days to wait before sending a reminder notification.

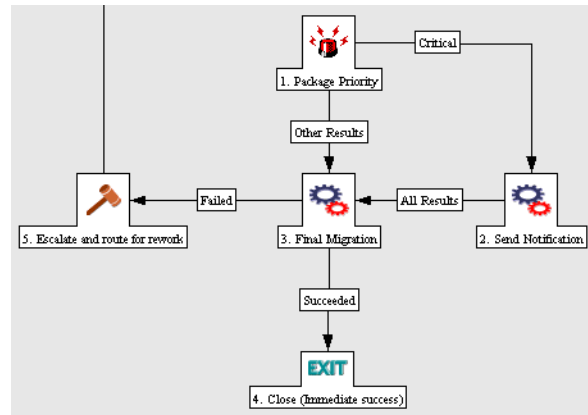


Configuration Tip: Sending a Notification Based on a Field Value.

The following configuration allows you to send a notification at a point in your process based on the value in a field. In this example, you want to send a notification to the Deployment manager at the “Final Migration” step for all critical Packages. This will alert him to other actions/communications that he may want to prepare for. Your configuration consists of the following:

- Token evaluation step for the priority field.

- Transition to another step (immediate execution) if the Priority = “Critical.”
- Notification upon entering the “Send Notification” step.

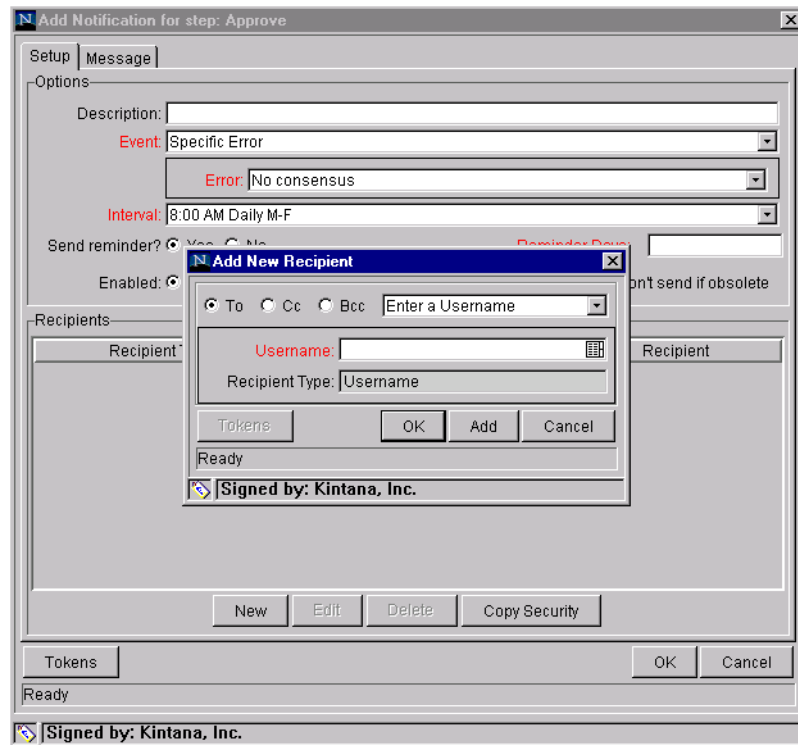


Configuring the Notification Recipients

When creating a notification, at least one recipient must be added for the message. The recipient can be a specific Kintana user, all members of a Security Group, or any email address.

To add a recipient to a notification:

1. Click **NEW** in the ADD NOTIFICATION FOR STEP window. The ADD NEW RECIPIENT window opens.



2. Select how you would like to specify the recipient from the drop down list. You can select to:
 - **ENTER A SECURITY GROUP** – select a specific Security Group, and all enabled users in the group with email addresses will receive the notification.
 - **ENTER A USERNAME** – select a specific User to receive the notification. The User must have an email address.
 - **ENTER AN EMAIL ADDRESS** – enter any email address to send the notification to.
 - **ENTER A STANDARD TOKEN** – select from a list of system tokens that corresponds to a User, Security Group, or Email Address.
 - **ENTER A USER DEFINED TOKEN** – enter any field token that corresponds to a User, Security Group, or Email Address.

Selecting a value will automatically update the field below. For example, selecting **ENTER A SECURITY GROUP** will change the field below to SECURITY GROUP.

3. Enter the specific value that corresponds to the recipient type selected above. This can be a Username, Email Address, Security Group, or a Token.

Recipient Configuration Tips

Tip 1:

Kintana recommends using Security Groups or dynamic access (Tokens) to define the notification recipients whenever possible. You should avoid specifying a list of users or an individual user's email address. If the list of users changes (due to a departmental or company reorganization), you would have to manually update that list. By using a Security Group instead of a list of users, you can update the Security Group once, and the changes are propagated throughout the Workflow steps.

Tip 2:

Use Tokens when sending a notification to an undetermined party. For example, you can configure the notification to be sent to the Assigned to User by specifying **[PKG.ASSIGNED_TO_USERNAME]** in the ADD NEW RECIPIENT window.

Configuring the Notification Message

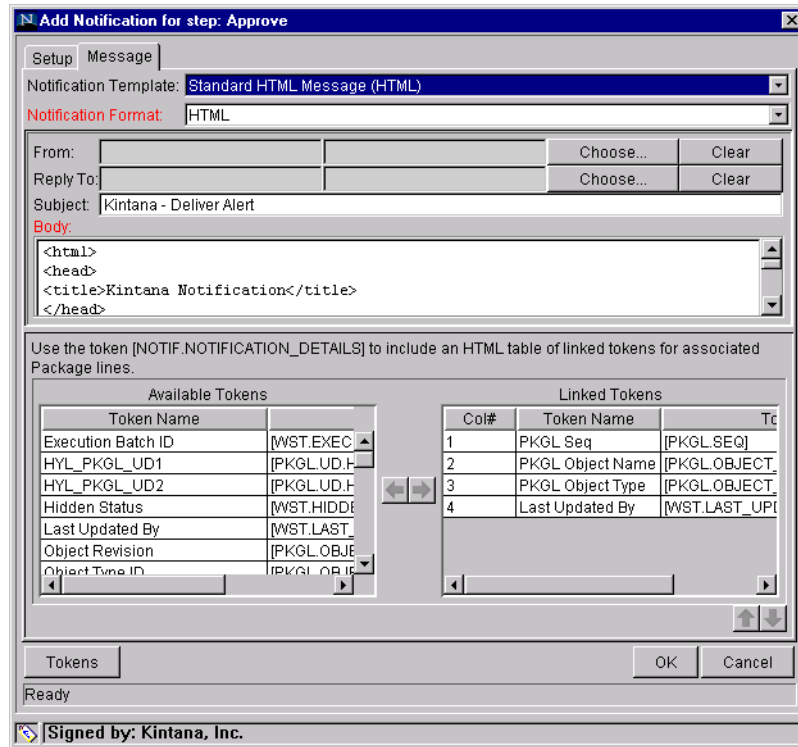
You can construct the notification's message to ensure that it contains the correct information or instructions for the recipient. For example, if the notification was sent to instruct the user that a Package deployment requires his approval, the message should instruct him to log onto Kintana and update the Package's status. Additionally, the notification should include a link (URL) to the referenced Package.

Kintana provides a number of features to make the notifications easier to configure and use:

- You can select from pre-configured notification templates to more quickly construct the body of your message.
- The body of the notification can be plain text or HTML.
- You can include multiple Tokens in the notification. These Tokens will resolve to information relevant to the recipient. For example, you can include Tokens for the URL to the Package approval page, information on Package status and priority, and emergency contacts.

To configure the message in a Notification:

1. Click the **MESSAGE** tab on the ADD NOTIFICATION FOR STEP window.



2. Select a NOTIFICATION TEMPLATE. This updates the contents in the BODY section with the information defined for the selected template.
3. Select **HTML** or **PLAIN TEXT** from the NOTIFICATION FORMAT field.

Selecting **HTML** allows you more flexibility when formatting the look and feel of your notification. You can write and test the HTML code in any HTML editor and then paste the code into the BODY window.

4. Select values for the From and Reply to fields.
5. Construct the BODY of the message. When constructing the body, consider utilizing the following:
 - Token for the URL to the Package (Workbench or HTML interface). See [Table 10-7](#) for a list of these tokens.

- Tokens in the Body of the message:
Click the **TOKENS** button to access the Token Builder window where you can select Tokens to add to the message body.
 - Tokens related to specific Package Lines:
Add Tokens to the LINKED TOKEN list to include tokens that resolve information related to the individual Package Line.
6. Click **OK** to save the notification specification.

Using Tokens in the Message Body

You can select any of the available tokens accessed through the Token builder to include in the body of your message. You should note, however, that not all Tokens will resolve in all situations. As a general rule, Tokens associated with the Package, Package Line, or Workflow will resolve.

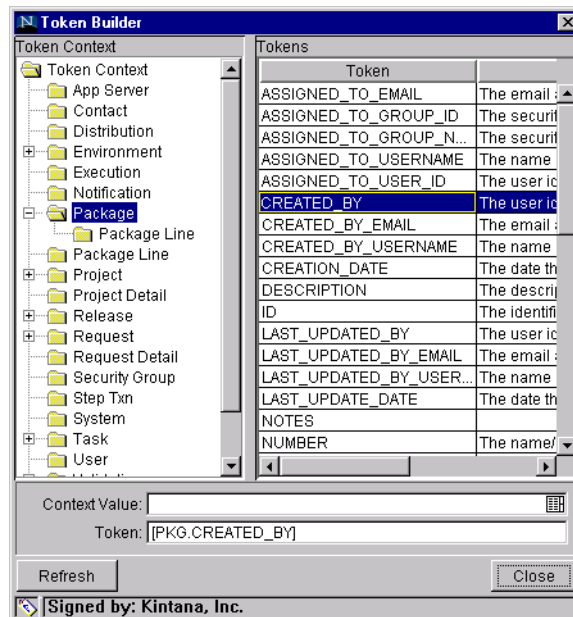
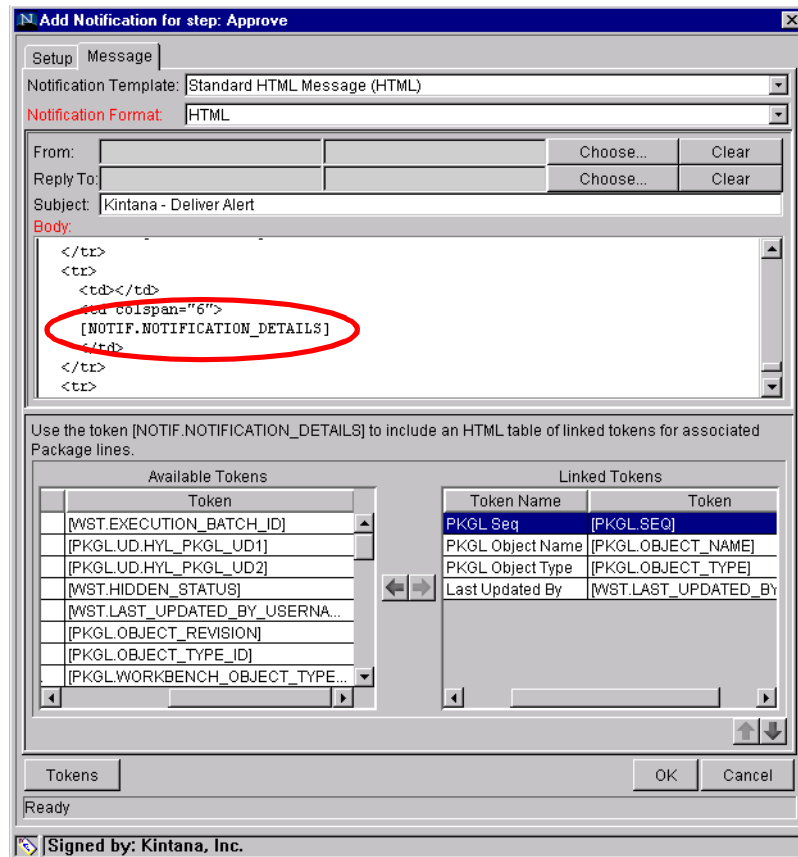


Figure 10-1 Token Builder Window

Special Case: Tokens in HTML Message

If you select to use an HTML formatted notification in your Package deployment process, you need to include an additional Token in the body to include all of the linked Tokens. Each Deliver (Package) HTML notification

should include the token “[NOTIF.NOTIFICATION_DETAILS]” within the <body> tags to incorporate linked tokens.



Including URLs to Open the Package (Smart URLs)

When a user receives a Notification, it is often helpful to include a link to the item that needs their attention. For example, John Smith receives a notification stating that his Package is ready for final approval. He clicks the URL included in the body of the notification and proceeds directly to the Package approval page.

Notifications can be configured in the body of the email notification to include the Web address (URL) for the following entities:

- Packages
- Requests
- Request Types

- Projects
- Tasks
- Workflows
- Validations
- Object Types
- Environments

If end-users are viewing their mail with a Web-based mail reader (such as Microsoft Outlook or Netscape Messenger), they can then click the URL in the notification and be taken directly to the referenced entity.

For Workflows, Request Types, Validations, Object Types and Environments the Notification can use the entity ID or the entity name as the parameter in the URL. This will bring the user to the correct window in the Workbench and open the detail window for the specified entity.

The most commonly used Smart URL Tokens for Packages and Requests are described in [Table 10-7](#).

Table 10-7. Smart URL Tokens

Smart URL Token	Description
PACKAGE_URL	Provides a URL that loads the Package Details page in the Kintana interface.
WORKBENCH_PACKAGE_URL	Provides a URL that loads the Package window in the Kintana Workbench.
REQUEST_URL	Provides a URL that loads the Request Details page in the Kintana interface.

Smart URLs in an HTML Formatted Messages

If you are using an HTML formatted message, you need to use an alternate Token to provide a link to the Package.

Table 10-8. Smart URL Tokens in HTML Format

Smart URL Token	Description
PACKAGE_NO_LINK	Provides a link that loads the Package window in the Kintana Workbench.

The Token will resolve to the following format:

```
<a href="http://URL">Package Name</a>
```

In the Notification, the link would appear as:

Package Name



Note

These Tokens can also be used in plain-text formatted Notifications. They will appear with the HTML tags showing.

Configuring Your Dashboard

The Dashboard provides an interface through which you can quickly assess the current state of the deployments. Personalize your Dashboard to display status information that is most meaningful to your role. For example, software engineers may only want to see the Packages that they submitted, whereas the IT manager may want to have visibility into each critical Package currently in progress.

Each user can personalize their own Dashboard to display only information relevant to their role. When configuring your deployment system, you need to consider the following configuration topics:

- *Controlling User Access to Portlets*
- *Creating and Distributing a Default Dashboard*
- *Creating Custom Portlets*

Related Topics:

- *"Using the Kintana Dashboard"*
- *"Configuring the Kintana Dashboard"*



Users are required to have a Dashboard license to add portlets to their Dashboard. See "[Kintana Security Model](#)" for details.

Controlling User Access to Portlets

You can control portlet user access at two levels:

- [Disabling Portlets](#)
- [Restricting User Access](#)

Disabling Portlets

You can disable custom-built portlets at your site. To disable a portlet:

1. Click the **DASHBOARD** screen group and click the **PORTLETS** icon.
2. Search for and open the custom Portlet that you would like to disable.

Note that you can not disable Kintana system portlets. To control access to these portlets, you can restrict user access. See [Restricting User Access](#) for details.

Portlet: Finance - My Expenses for Last 2 Periods

Portlet Name: Finance - My Expenses for Last 2 Periods Product Scope: Kintana Deliver

Default Title: Finance - My Expenses for Last 2 Periods Portlet Category:

Default Max Rows Displayed: 10 Portlet Width: Wide

Description:

Enabled: Yes No

Time-Out: Use Default 20 Seconds

Currently Used By 47 User(s)

User Access

Allow all users to add this portlet to their dashboard

Security Type	Security

Remove

Security Group: Add Security Group(s)

User: Add User(s)

Verify OK Save Cancel

System portlets only allow editing of user access, and cannot be copied or deleted.

3. Click **ENABLED = No**.



Note

If there are any users currently using the portlet on their Dashboard, disabling the portlet will delete it from their Dashboards.

4. Click **SAVE**.

Related Topics:

- ["Using the Kintana Dashboard"](#)
- ["Configuring the Kintana Dashboard"](#)

Restricting User Access

You can control which users can add a portlet to their Dashboard. For example, you may want to restrict the Package-related portlets to only members involved in the deployments. Enabling only the portlets that a specific user needs will make it easier for that user to personalize their Dashboard, because there are less (non-relevant) portlets to choose from.

To specify which users can use the portlet on their Dashboard:

1. Click the **DASHBOARD** screen group and click the **PORTLETS** icon.
2. Search for and open the Portlet that you would like to configure.
3. Click the **USER ACCESS** tab. For system portlets (such as My Packages), the **USER ACCESS** tab is the only displayed tab.

Portlet: My Packages

Portlet Name: My Packages Product Scope: Kintana Deliver

Default Title: My Packages Portlet Category: Packages

Default Max Rows Displayed: 5 Portlet Width: Wide

Description: Displays all Packages created by or assigned to the current user. Users can drill down

Enabled: Yes No Time-Out: Use Default 20 Seconds

Currently Used By 32 User(s)

User Access

Allow all users to add this portlet to their dashboard

Security Type	Security

Remove

Security Group: Add Security Group(s)

User: Add User(s)

Verify OK Save Cancel

System portlets only allow editing of user access, and cannot be copied or deleted.

4. Un-check the ALLOW ALL USERS TO ADD THIS PORTLET TO THEIR DASHBOARD field. The SECURITY GROUP and USER fields are enabled.
5. Select the desired Security Groups or Users and click the respective ADD button. They are added to the USER ACCESS tab.

Portlet: My Packages

Portlet Name: My Packages Product Scope: Kintana Deliver

Default Title: My Packages Portlet Category: Packages

Default Max Rows Displayed: 5 Portlet Width: Wide

Description: Displays all Packages created by or assigned to the current user. Users can drill down

Enabled: Yes No Time-Out: Use Default 20 Seconds

Currently Used By 32 User(s)

User Access

Allow all users to add this portlet to their dashboard

Security Type	Security
Security Group Name	Alliance Manager
Security Group Name	Alliance Partner Portal Approv
Security Group Name	Alliances Group
Security Group Name	Alliances VP & Strat Dir

Remove

Security Group: Add Security Group(s)

User: Add User(s)

Verify OK Save Cancel

System portlets only allow editing of user access, and cannot be copied or deleted.

6. Click SAVE.

You can restrict access by specifying multiple Security Groups and Users for each portlet. Only members of the specified Security Group or the specified users can add this portlet to their Dashboard.



Note

You can restrict user access for both custom and system portlets.

Creating and Distributing a Default Dashboard

You can configure a Default Home page that all Kintana users will see when they log into Kintana for the first time. This saves time and allows first-time users to more quickly and easily integrate the Dashboard into their business processes. The Default Dashboard is configured using the Kintana HTML interface. See "[Configuring the Kintana Dashboard](#)" for detailed instructions.



Note

Users must have the **EDIT DEFAULT USER HOMEPAGE** Access Grant to configure the default Dashboard.

Creating Custom Portlets

Portlets are visual displays that act as windows into different aspects of Kintana data. While Kintana's system portlets (provided at the time of installation) are personalizable by end-users and provide wide access to your Kintana data, you can also create custom portlets to access additional information in Kintana or in other databases. These custom portlets behave the same as the system portlets, using filter fields to limit the displayed data. You can create textual or graphical portlets.

Because custom portlets are data-driven entities and require extracting information stored in the database, knowledge of SQL is required for users who wish to create or configure portlets.

See "[Configuring the Kintana Dashboard](#)" for detailed instructions on creating custom portlets.

Kintana Portlets to Enable for Use on the Dashboard

Before creating custom portlets, consider using one of Kintana's system portlets. Review the business and data presentation portlet requirements and compare against Kintana's system portlets. See "[Configuring the Kintana Dashboard](#)" for a complete list of Kintana's portlets.

Configuring Reports

Kintana features a pre-defined set of HTML-based reports that are accessed through a Web browser. The reports allow users to view the current detailed status of their Kintana data at any point in time. Kintana's Decision Support System (DSS) reports provide users with a high level overview of their initiatives through graphical summary reports.

Kintana also provides a Reporting Meta Layer, which allows users to build their own custom reports using third-party reporting tools.

See "[Kintana Reports](#)" for a full list of available reports and information on configuring them.

Chapter
11**Rolling Out Your Deployment Process**

After configuring your deployment process in Kintana you need to roll-out the system and processes to your user-base. This chapter discusses a number of topics to consider before and during the roll-out phase:

- [*Test the Deployment System - Checklists*](#)
- [*Migrate Your Configuration Data into Production*](#)
- [*Enable Entities and User Access*](#)
- [*Educating Your Users*](#)

Related Topics:

- [*“Developing Your Kintana Configurations \(Using Migrators\)”*](#) on page 37
- [*“Kintana Migrators”*](#)

Test the Deployment System - Checklists

Ensure a successful roll-out by thoroughly testing your processes and Kintana configurations. This section provides a series of high-level checklists to help you validate your system.

- [*General Deployment System Configuration Checklist*](#)
- [*Workflow Checklist*](#)
- [*Object Type Checklist*](#)

- [Environments Checklist](#)
- [Security / User Access Checklist](#)
- [Dashboard / Portlet Checklist](#)
- [Cross-Entity Checklist](#)

General Deployment System Configuration Checklist

The following items have to be configured to enable your deployment system. See the referenced sections in the Notes column for additional details/instructions on configuring each of the entities.

Table 11-1. General Configuration Checklist

	Entity Defined?	Notes
	Workflow	<p>You must have one or more Workflows that will be used to process the Packages (deploy your objects). See the following sections for details on Workflow construction:</p> <ul style="list-style-type: none"> • “Mapping your Process into a Kintana Workflow” on page 79 • “Advanced Workflow Topics” on page 257 • “Configuration Workbench Reference” document
	Object Types	<p>You need to define an Object Type for each type of object to be deployed. This includes creating fields that describe the object and commands required to process it during deployment. See the following sections for details on Object Type and Command construction:</p> <ul style="list-style-type: none"> • “Constructing the Object Type” on page 135 • “Validations” on page 287 • “Using Commands and Tokens” document
	Environments	<p>You must define the source and destination Environments for the objects being deployed. See the following sections for details on Environment definition:</p> <ul style="list-style-type: none"> • “Defining your Environments” on page 163 • “Environment Workbench Reference” document

Table 11-1. General Configuration Checklist

Entity Defined?	Notes
Security Groups / User Access	<p>You need to define the Security Groups used to control different aspects of the deployment process: Package creation, Package processing, and deployment system configuration. See the following sections for details on Security Group and User Participant definition:</p> <ul style="list-style-type: none">• <i>“Integrating Participants into Your Deployment System”</i> on page 195• <i>“Kintana Security Model”</i> document
Dashboard / Portlets	<p>You need to decide which portlets can be added to the Dashboard. If none of the default system portlets suit your business needs, you can construct your own custom portlets. See the following sections for details on Portlet construction and default Dashboard creation:</p> <ul style="list-style-type: none">• <i>“Setting Up Communication Paths”</i> on page 215• <i>“Configuring the Kintana Dashboard”</i> document

Workflow Checklist

Table 11-2. Workflow configuration checklist

	Workflow Check Item	Notes
	Business process is modeled on the Workflow	<p>You have added execution, decision and condition steps to the LAYOUT tab on the WORKFLOW window. See the following sections for details:</p> <ul style="list-style-type: none"> • “Building the Workflow Skeleton - Overview” on page 79
	Command execution points are set	<p>You have decided at which points commands will run. If they are object-specific commands, or commands that need to run for each package line, then you should execute the Object Type commands. If they are other, non-object-specific commands, consider setting them in the Workflow step source.</p>
	Decision steps set	<p>See the following sections for details:</p> <ul style="list-style-type: none"> • “Create the Required Step Source” on page 81
	Timeouts are set	<p>You have placed timeout values on how long Workflow steps can remain in a single state, and have added timeouts to command executions. This ensures that the deployment process is not delayed from lack of user action or complications during executions. See the following sections for details:</p> <ul style="list-style-type: none"> • “Creating a Decision Type Step” on page 85 • “Create an Execution Type Step” on page 90
	Automatic transitions are properly set	<p>Ensure that the Package will not become “stuck” in a step. This can happen when the results of an execution or query yield a result that is not linked to a transition out of the step. See the following sections for details:</p> <ul style="list-style-type: none"> • “Adding Transitions Between Steps” on page 122
	Manual transitions are set	<p>Ensure that the step has a transition path for each available decision result. See the following sections for details:</p> <ul style="list-style-type: none"> • “Adding Transitions Between Steps” on page 122

Table 11-2. Workflow configuration checklist

	Workflow Check Item	Notes
	Deployment steps specify a source and destination environment	Execution steps (and the included commands) need to know which environments to connect to for machine connections and object transfers. See the following sections for details: <ul style="list-style-type: none"> • “Create an Execution Type Step” on page 90
	Notifications are set on appropriate Workflow steps	You need to configure notifications to be sent at specific points in the process. See the following sections for details: <ul style="list-style-type: none"> • “Adding Notifications to Workflow Steps” on page 216
	Includes a Close step.	The process should conclude with a “Closed” Package at all exit points.
	Verify the Workflow	Use the Workflow’s VERIFY tool to ensure that you haven’t made any serious configuration errors. The Workflow verification tool checks for the possible configuration errors described in Table 11-3 .

Table 11-3. Workflow Logical Guidelines

Guideline	Returns	Reason
Workflow should have at least one step.	Error	No processing can be done if the Workflow has no steps.
Workflow should have at least one Close step.	Error	The Package Line cannot be closed without a Close step in the Workflow.
Each enabled Workflow step should have at least one incoming transition	Error	It is not possible to flow to a Workflow Step without an incoming transition.
Each decision step should have at least one security group, user or token defined in the Security tab.	Error	No one is authorized to act on the step without a Security Group.
Each manual execution step should have at least one Security Group, user or token defined in the Security tab.	Error	No one is authorized to act on the step without a Security Group.
First Workflow step should not be a condition.	Error	Workflow processing may not be correct if the first step is a condition.
A condition step should not have a transition to itself.	Error	A condition with a transition to itself could cause the Workflow to run indefinitely.

Table 11-3. Workflow Logical Guidelines

Guideline	Returns	Reason
Transition value is not a valid validation value (error).	Error	The Validation value has changed since the transition has been made.
Close steps should not have a transition on 'Success' or 'Failure.' Return steps should have no outgoing transitions.	Error	The Package or Request will not close if a transition exists on 'Success.'
An immediate execution step should not have a transition to itself on 'Success' or 'Failure.'	Error	The Workflow could loop indefinitely.
'Other Values' and 'All Values' transitions should not exist at the same step.	Warning	'Other Values' transition is always ignored if an 'All Values' transition exists.
Each Workflow Step should have at least one outbound transition.	Warning	The branch of the Workflow stops indefinitely without closing the Package Line or Request.
Each value from a list-validated validation should have an outbound transition.	Warning	There are validation values that do not have transitions defined.
Step with text or numeric validation should have an 'Other Values' or 'All Values' transition.	Warning	Since text and numeric validations are not limited, an 'Other Values' or 'All Values' transition should be defined.
All steps should be enabled.	Warning	Disabled steps cannot be used by a Package Line or Request.
AND or OR condition step should have at least two incoming transitions.	Warning	An AND or OR condition with only one incoming transition will always immediately be true and have no effect.
Subworkflow should have at least one Return step.	Error	Should include a Return step.
Notifications with reminders should not be set on results that have transitions.	Error	Transition into the Return Step does not match the validation.
Close step in Subworkflow will close entire Package Line or Request.	Warning	Has a Close step.
Top-level Workflow should not have a Return step.	Error	Only Subworkflows have a Return step.

Object Type Checklist

Table 11-4. Object Type configuration checklist

	Object Type Check Item	Notes
	Fields defined	<p>Fields are required to define the object. Ensure that you have the correct parameters to describe the object to be deployed. See the following sections for details:</p> <ul style="list-style-type: none">• <i>“Creating Object Type Fields”</i> on page 137• <i>“Validations”</i> on page 287
	Commands defined	<p>You have constructed all commands needed to process and deploy the object. See the following sections for details:</p> <ul style="list-style-type: none">• <i>“Creating Object Type Commands”</i> on page 156• <i>“Using Commands and Tokens”</i> document
	Conditions set in commands	<p>You have added conditions to steps within your command that dictate when the specific command steps run. See the following sections for details:</p> <ul style="list-style-type: none">• <i>“Using Commands and Tokens”</i> document

Environments Checklist

Table 11-5. Environment definition checklist

	Environment Check Item	Notes
	Define the “source” Environment	See the following sections for details: <ul style="list-style-type: none"> • <i>“Defining Environments in Kintana”</i> on page 164
	Define the “destination” Environment	See the following sections for details: <ul style="list-style-type: none"> • <i>“Defining Environments in Kintana”</i> on page 164
	Select the appropriate connection protocol	See the following sections for details: <ul style="list-style-type: none"> • <i>“Selecting the Environment’s Connection Protocol”</i> on page 167
	Select the appropriate transfer protocols	See the following sections for details: <ul style="list-style-type: none"> • <i>“Selecting the Environment’s Transfer Protocol”</i> on page 168
	Define Environment Groups	See the following sections for details: <ul style="list-style-type: none"> • <i>“Creating Environment Groups”</i> on page 176
	Verify the Environment Definitions	See the following sections for details: <ul style="list-style-type: none"> • <i>“Environment Maintenance and Utilities”</i> on page 190

Security / User Access Checklist

Table 11-6. Security / User Access Configuration Checklist

Security / User Access Check Item	Notes
Created Security Groups (for access to screens and functions)	<p>You have created security groups to be used to grant access to certain screens and functions. See the following sections for details:</p> <ul style="list-style-type: none"> • “Establishing Security Groups” on page 197 • “Kintana Security Model” document
Created Security Groups (for association with Workflow steps)	<p>You have created Security Groups to allow users to act on a specific Workflow step. See the following sections for details:</p> <ul style="list-style-type: none"> • “Establishing Security Groups” on page 197 • “Kintana Security Model” document
Set security on Package Creation	<p>You have set all available options for restricting who can create and submit Packages. See the following sections for details:</p> <ul style="list-style-type: none"> • “Setting Package Creation Security” on page 202
Set security on Package processing	<p>You have set all available options for restricting who can process Packages. See the following sections for details:</p> <ul style="list-style-type: none"> • “Setting Package Processing Security” on page 206
Set security on deployment system configuration	<p>You have specified who can modify the deployment process. This includes editing the Workflow, Object Type, Environment, Security Groups, etc. See the following sections for details:</p> <ul style="list-style-type: none"> • “Setting Configuration Security” on page 210

Dashboard / Portlet Checklist

Table 11-7. Dashboard / Portlet Configuration Checklist

	Dashboard Check Item	Notes
	Created custom portlets to display desired data	<p>Advanced Kintana users with a knowledge of SQL programming can create their own Dashboard. See the following sections for details:</p> <ul style="list-style-type: none"> • <i>"Configuring Your Dashboard"</i> on page 236
	Enable portlets for use on the Dashboard	<p>See the following sections for details:</p> <ul style="list-style-type: none"> • <i>"Configuring the Kintana Dashboard"</i> document • <i>"Kintana Security Model"</i> document
	Specify which users can add use certain portlets	<p>See the following sections for details:</p> <ul style="list-style-type: none"> • <i>"Configuring the Kintana Dashboard"</i> document • <i>"Kintana Security Model"</i> document
	Create a default Dashboard	<p>See the following sections for details:</p> <ul style="list-style-type: none"> • <i>"Configuring the Kintana Dashboard"</i> document

Cross-Entity Checklist

Table 11-8. Cross Entity Configuration Checklist

	Entities	Configuration Considerations
	Workflow and Object Type	<p>The following items should be coordinated between the Workflow and Object Type:</p> <ul style="list-style-type: none"> • Decide which Workflow steps will execute the Object Type commands. • Decide which Object Type commands will run at specific Workflow steps (using command conditions) • Workflow step source validations and Object Type field validations are in agreement. This is required when transitioning based on a field value (using Token, SQL or PL/SQL execution types) • Allow the Object Type use for the Workflow (set in the WORKFLOW window - DELIVER SETTINGS tab).
	Workflow and Environments	<p>The following items should be coordinated between the Workflow and Environments:</p> <ul style="list-style-type: none"> • Specify the source and destination Environments (or Environment Groups) on the appropriate Workflow execution steps.
	Workflow and Security Groups	<p>The following items should be coordinated between the Workflow and Security Groups:</p> <ul style="list-style-type: none"> • Associate Security Groups with Workflow steps. Users in the included groups can act on the step. • Set Workflow and Workflow step ownership.
	Object Types and Environments	<p>The following items should be coordinated between the Object Types and Environments:</p> <ul style="list-style-type: none"> • Specify any Environment overrides in the Object Type commands.
	Security Groups and other entities (Object Types, Environments, etc.)	<p>Set Ownership Groups for these entities. Members of the Ownership Group (determined by associating Security Groups) are the only users who can edit the entities.</p>

Migrate Your Configuration Data into Production

“Developing Your Kintana Configurations (Using Migrators)” on page 37 discusses using multiple Kintana instances to configure and deploy your configuration data. After all entities have been validated in a Development or Testing environment, you can perform the final migration into production. The following Kintana entities can be transferred using the Kintana migrators:

- Workflows
- Object Types
- Validations
- User Data Context
- Special Commands
- Report Types
- Request Types
- Request Header Types
- Project Template
- Portlet Migrator

Note

When you migrate the above entities, related configuration information is also migrated. For example, when migrating the Object Type the following information is also migrated: Validations referenced by the Object Type fields, Environments referenced by the Validations, and Special Commands referenced by Commands or Validations.

You can process all of the migrations in a single Package. Each individual entity (Workflow A, Workflow B, Object Type 1, Object Type 2, etc.) is added as a separate Package Line. See the *“Kintana Migrators”* user guide for detailed instructions on using the Kintana migrators.

The entity will inherit the Enabled or Disabled status. For example, if the Object Type is disabled in the Test instance, then it will be disabled in the Production instance upon migration. Ensure that each entity has the desired Enabled or Disabled status in the production instance.

Enable Entities and User Access

Each entity used in your deployment process includes an `ENABLED` parameter. This parameter needs to be set to **YES** in order to provide general access. Ensure that the following entities are enabled in the system:

- Workflows (including subworkflows)
- Object Types
- Environments
- Environment Groups
- Security Groups
- Users
- Portlets

Educating Your Users

The final step in rolling out your deployment system is training your users. This includes the following activities:

- Basic Kintana use: creating, processing, and reporting on Packages
- Process-specific training:
Ensure that each of your users understands the deployment process. You may want to have a formal roll-out meeting or publish documents on the configurations and processes at your site.
- User Responsibilities:
Ensure that each of your users understands their individual role in the deployment process. For example, the QA team may be restricted to only approve the testing phase of the deployment. Also, take advantage of Kintana's email notification functionality. The notifications can be very specific, instructing individual users with their required deployment tasks.

Additional Resources for Education and Roll-Out

Consider using the following resources to assist in your education and roll-out activities:

Kintana Education and Online courses

Kintana has created a complete product training curriculum to help you achieve optimal results from your Kintana applications. Learn more about our Education offering from our Web site:

<http://www.kintana.com/services/services.htm>

Site Help

The Kintana Site Help can be used to document your company's specific Kintana configurations. The Site Help is typically filled out by a Kintana product consultant during the initial Kintana implementation. The Site Help can then be maintained by your company and referenced for archival or training purposes. The Site Help is accessed from the Help menus in both the Kintana interface and Workbench.

Online Help

The Kintana Online Help provides details on creating and processing Kintana Requests, Packages, Projects and Tasks. This Help system features information on using Kintana's main HTML interface and the Kintana Dashboard.

Kintana Power Users can access other Kintana product documentation from the Kintana Workbench. From the **HELP** menu, select **KINTANA LIBRARY**. A single page opens and provides access to all of Kintana's user, configuration and administration documentation.

Appendix



Advanced Workflow Topics

Workflows are discussed in the Kintana Business Application Guides as they relate to business processes. This chapter provides additional instructions for advanced Workflow configurations. It is organized into the following sections:

- *Using Subworkflows*
- *Package - Request Workflow Integration*
- *Using Condition Steps*
- *Setting the Reopen Step for Request Workflows*
- *Modifying Workflows in Use*
- *Using Workflow Parameters*

Using Subworkflows

A Kintana Subworkflow is any Workflow that is referenced from within another Workflow. Subworkflows allow you to model complex business processes into logical, more manageable and reusable sub-processes.

Subworkflows are defined in the same manner as typical Kintana Workflows. Two things to keep in mind when working with Subworkflows:

- The WORKFLOW window contains a SUBWORKFLOW radio button which should be set to **YES**.
- The Validation for the step leaving the Subworkflow layout should match the Subworkflow step in the parent Workflow.



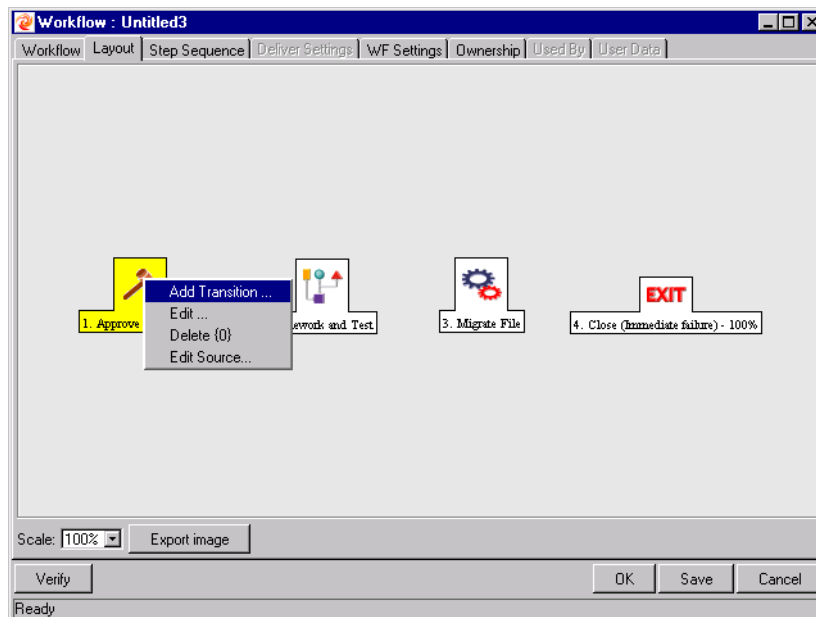
Subworkflows can also be generated by copying and renaming an existing Subworkflow.

A Subworkflow can be selected from the **WORKFLOW STEP SOURCES** window and dragged onto the **LAYOUT** tab. When the Package, Request, or Release Distribution reaches the Subworkflow step, it follows the path defined in that Subworkflow. The Subworkflow will either close within that Workflow or return to the parent Workflow.

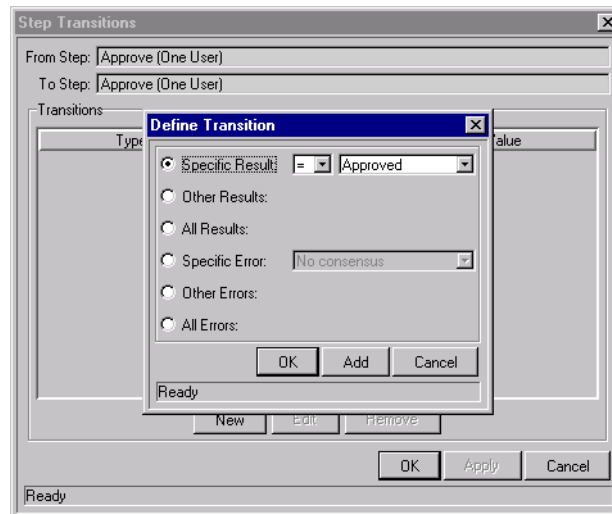
Transitioning to a Subworkflow

A transition to a Subworkflow Step is made in the same way as a transition to any other Workflow Step (Execution, Decision or Condition):

1. Open a Workflow and click the **LAYOUT** tab.
2. Right-click the source Workflow Step and select **ADD TRANSITION** from the pop-up menu. This creates an arrow from the source Workflow Step.



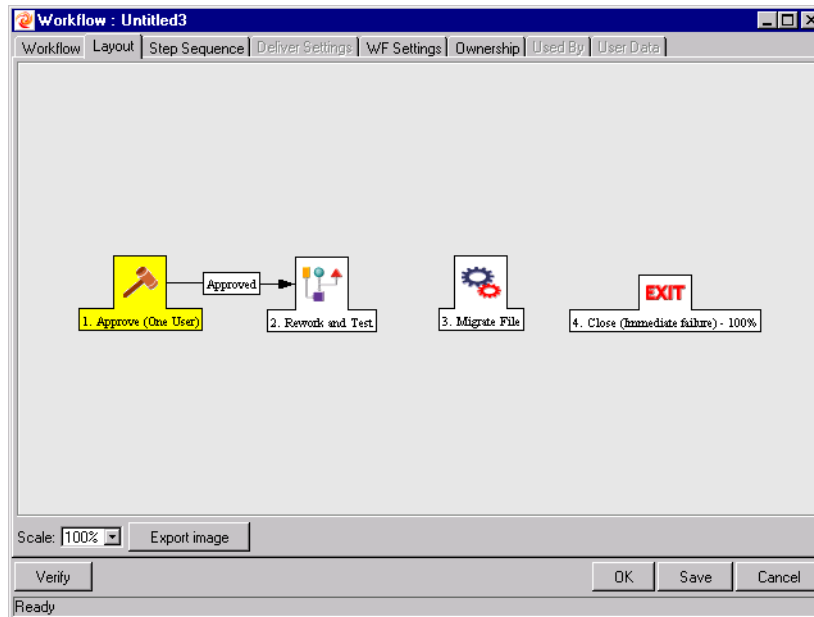
3. Drag the arrow to the destination Step and left-click. The **STEP TRANSITIONS** window opens.
4. Click **NEW**. The **DEFINE TRANSITION** window opens.



5. Define the desired transaction result by:
 - a. Clicking one of the radio buttons for results (SPECIFIC RESULT, OTHER RESULTS, etc.).
 - b. Selecting = or != (does not equal) from the drop down list.
 - c. Selecting **YES** or **No** from the drop down list.

If the Workflow Step results in this value, the Request or Package proceeds along this path.

6. Click **OK** to close the DEFINE TRANSITION window.
7. Click **OK** to finalize the step transition definition.

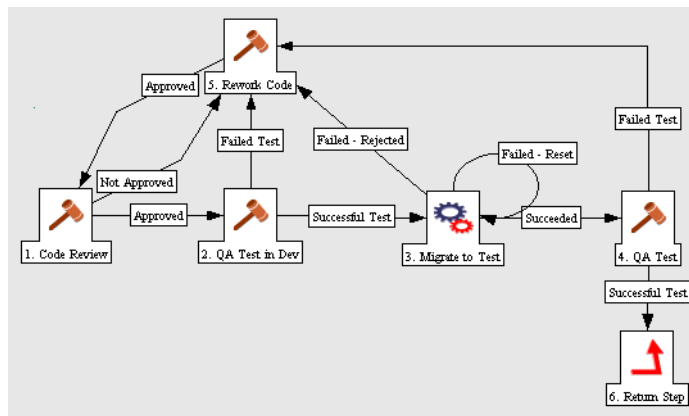
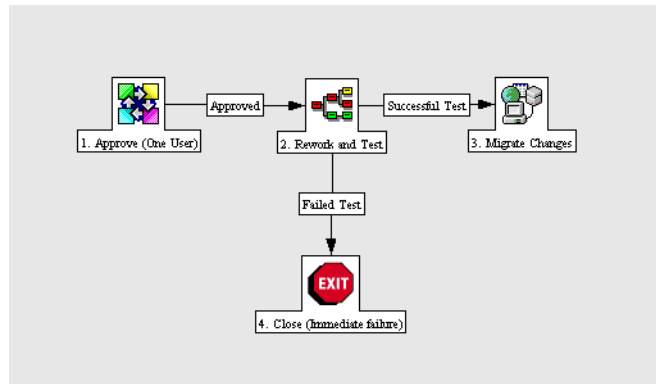


The transition is graphically represented by an arrow between the two steps. The Package Line or Request proceeds to the First Step designated in the Subworkflow definition.

Transitioning From a Subworkflow

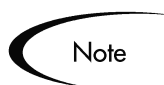
When the Package or Request reaches the Subworkflow Step, it follows the path defined in that Subworkflow. It either closes within that Workflow (at a Close step) or returns to the parent Workflow.

For a Package Line or Request to transition back to the parent Workflow, the Subworkflow must contain a Return step. The transitions leading into the Return step must match the Validation established for the Subworkflow Step. In the following example, the transitions exiting the REWORK AND TEST step (**SUCCESSFUL TEST** and **FAILED TEST**) match the possible transitions entering the Subworkflow's Return Step.



Users must verify that the Validation defined for the Subworkflow Step is synchronized with the transitions entering the Return Step. The Subworkflow Validation is defined in the WORKFLOW window.

Users typically define the possible transitions from the Subworkflow Step during the Subworkflow definition.



The Subworkflow Step validation cannot be edited if the Subworkflow is used in another Workflow definition.

The Subworkflow field cannot be edited if the Subworkflow is used in another Workflow definition.

Package - Request Workflow Integration

Kintana Request and Package Workflows can be configured to work together, communicating at key points in the Request and Package processes. A Request Workflow Step can actually jump to a preselected Package Workflow Step. The Package Workflow step receives the Request Workflow Step and acts on it to go to the next step in the process.



Note

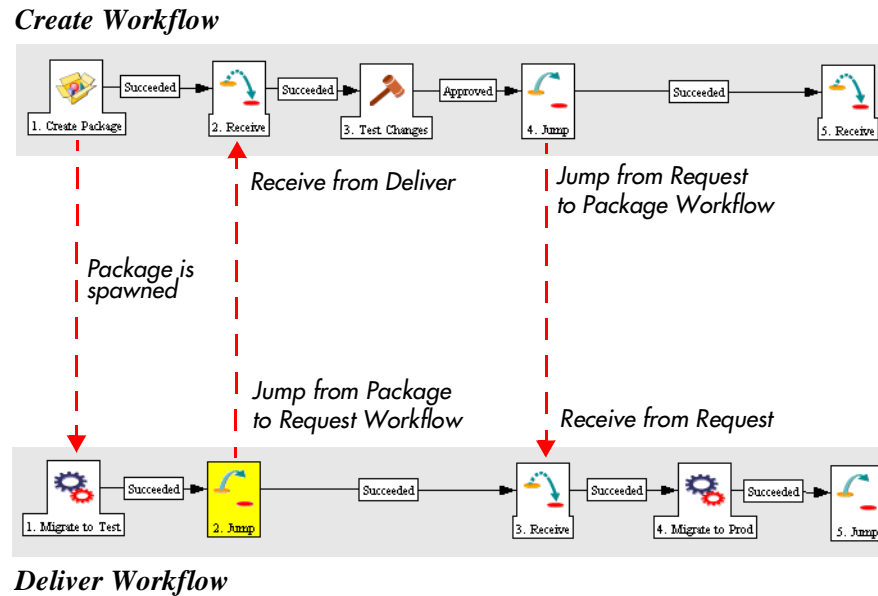
Kintana also supports another level of Request - Package integration that does not rely on the Workflow configuration. You can attach Packages and Requests to each entity as References. You can then set dependencies on these reference to control the behavior of the Request or Package. For example, you can specify that a Request is a “Predecessor” to the Package. This means that the Package will not continue until the Request closes.

Kintana has provided two built-in Workflow Events available in the Workflow Workbench to facilitate the cross-product Workflow integration. These steps are `WF_JUMP` and `WF_RECEIVE`. Jump (`WF_JUMP`) and Receive (`WF_RECEIVE`) steps can be created that define the points of interaction between Workflows. Each Jump step must be coupled with a Receive step. Workflows can communicate through these Jump and Receive pairs.

As an example of when this kind of communication is useful:

1. A Request spawns a Package for migrating new code to the Production environment.
2. The newly spawned Package must go through an `APPROVAL` step in Deliver.
3. When the `APPROVAL` step is successful, the process jumps back to and is received by the Request. The Request then undergoes more testing and changes in the QA Environment.
4. After successfully completing the QA Test, the process jumps from the Request and is received by the Package.
5. Because the step has succeeded, the process can now migrate the code changes to the Production Environment.

This process is graphically represented in [Figure A-1](#).



Deliver Workflow
 Figure A-1 Jump/Receive Workflow Steps

The Jump and Receive pair must be carefully coordinated. Each JUMP step must have an associated RECEIVE step, linked together by a common JUMP/RECEIVE STEP LABEL defined in the WORKFLOW STEP window. The transition values for entering into and exiting the JUMP and RECEIVE steps must also be coordinated.

This section details the process for setting up a successful Request - Package Workflow integration. To establish communication between Requests and Packages:

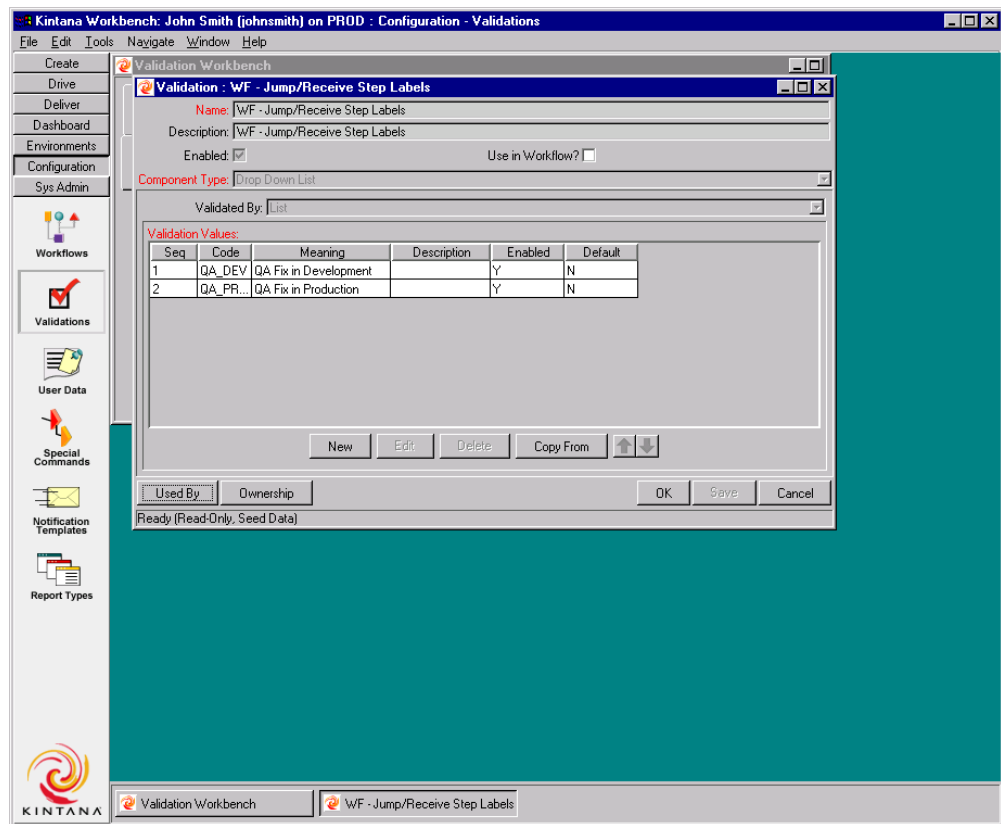
1. Set up the 'WF - JUMP/RECEIVE STEP LABELS' Validation for use in the WORKFLOW STEP window. This validation is used to group a JUMP and RECEIVE step. The selected JUMP/RECEIVE STEP LABEL must match in the paired Jump and Receive Workflow Step windows. See ["Setting Up the 'WF - Jump/Receive Step Labels' Validation"](#) on page 264.
2. Create a JUMP step using the **WF_JUMP** BUILT-IN WORKFLOW EVENT. See ["Generating a Jump Step Source"](#) on page 266.
3. Create a RECEIVE step using the **WF_RECEIVE** BUILT IN WORKFLOW EVENT. See ["Generating a Receive Step Source"](#) on page 267.
4. Verify that both the JUMP and RECEIVE steps specify the same JUMP/RECEIVE STEP LABEL. See ["Including the Jump/Receive pair in Workflows"](#) on page 269.

5. Verify that the transitions exiting the JUMP and RECEIVE steps match the possible values entering the JUMP step.

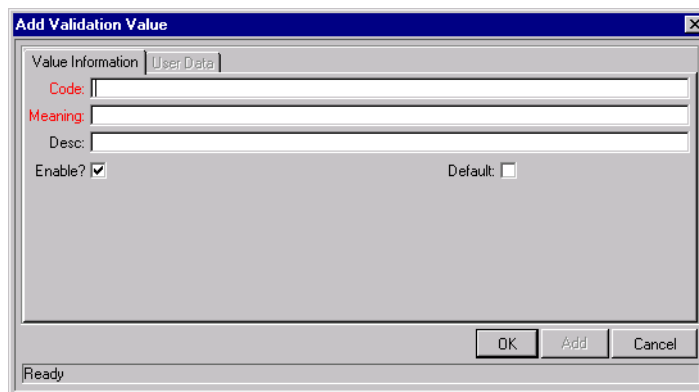
Setting Up the 'WF - Jump/Receive Step Labels' Validation

To set up the WF - JUMP/RECEIVE STEP LABELS Validation:

1. Click the **CONFIGURATION** screen group and click the **VALIDATIONS** icon. The **VALIDATION WORKBENCH** opens.
2. On the **QUERY** tab, enter **WF - JUMP/RECEIVE STEP LABELS** in the **VALIDATION NAME** field.
3. Click **LIST**.
4. Click the **RESULTS** tab. The WF - JUMP/RECEIVE STEP LABELS is listed in the **RESULTS** tab.
5. Click **OPEN**. The **VALIDATION** window opens.



6. Click **NEW** to define a new Validation Value that is used to link two Workflows together. The ADD VALIDATION VALUE window opens.



7. Enter the desired CODE, MEANING and DESCRIPTION in the appropriate fields.
8. Click **OK** to close the ADD VALIDATION VALUE window.
9. Click **OWNERSHIP** to select which Ownership Groups will have the ability to edit this Validation.

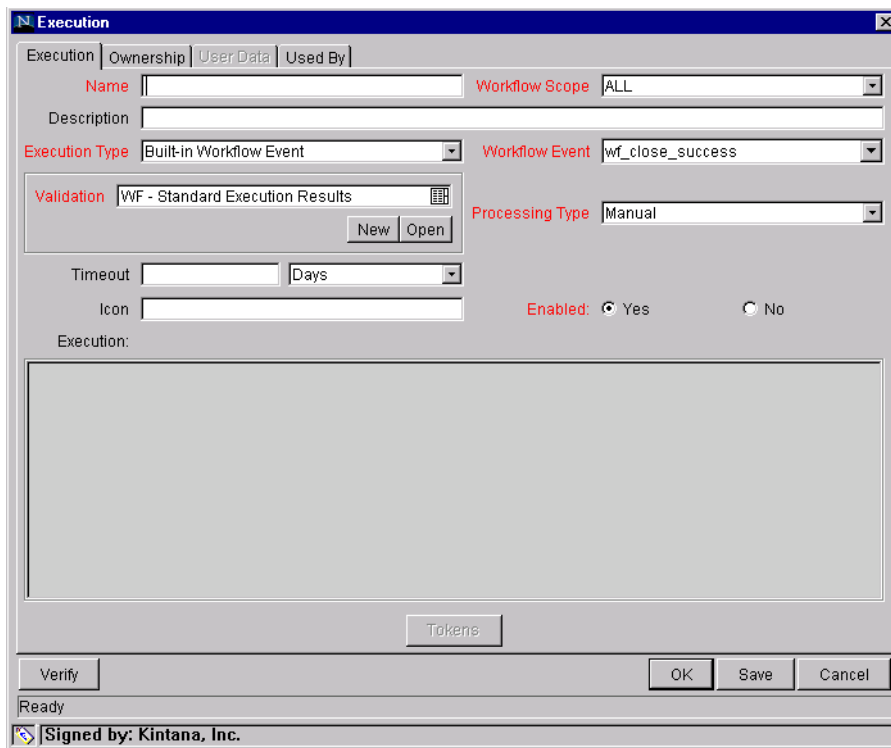
10. Click **OK** to close the VALIDATION window.

The new Validation Value is now included in the JUMP/RECEIVE STEP LABEL drop down list in the WORKFLOW STEP window.

Generating a Jump Step Source

To create a Jump step using the **WF_JUMP** BUILT-IN WORKFLOW EVENT:

1. Click the **CONFIGURATION** screen group and click the **WORKFLOWS** icon. The WORKFLOW WORKBENCH and WORKFLOW STEP SOURCES window open.
2. Select the WORKFLOW STEP SOURCES window.
3. Select the EXECUTIONS folder.
4. Click **NEW**. The EXECUTION window opens.



5. Select either **PACKAGES** or **REQUESTS** from the WORKFLOW SCOPE drop down list, depending on the desired application of the Workflow. Package Level Subworkflows can not include jump and receive steps.
6. Select **BUILT-IN WORKFLOW EVENT** from the EXECUTION TYPE drop down list.

7. Select **WF_JUMP** from the **WORKFLOW EVENT** drop down list.
8. Select or create a **Validation** from the **VALIDATION** drop down list which will be used to transition out of this **Workflow Step**.

Note

The **Validation** values exiting the **Jump** step must match the possible **Validation** values entering the **Jump** step.

9. Fill in any other required or optional information in the **EXECUTION** window (such as **NAME**, **DESCRIPTION** or **PROCESSING TYPE**).
10. Click the **OWNERSHIP** tab to select which **Ownership Groups** will have the ability to edit this **Execution** step.
11. Click **OK**. The **Workflow Step** is added to the **WORKFLOW STEP SOURCES** window.

This **Workflow Step** can now be used in any new or existing **Workflow** within the step's defined **Workflow Scope**. Remember that every **JUMP** step must have a paired **RECEIVE** step in another **Workflow**.

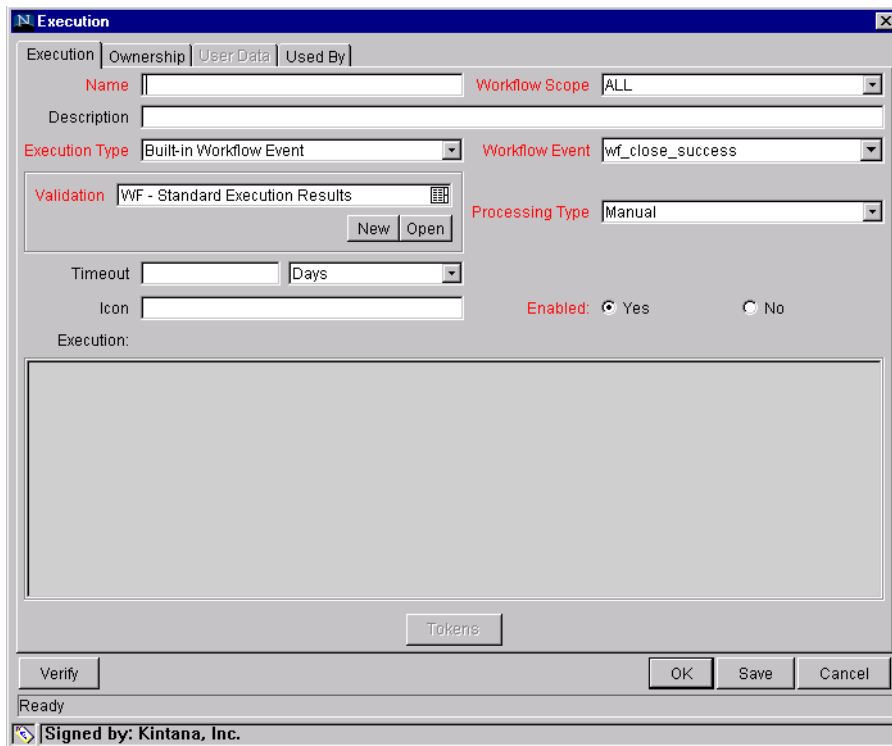
Generating a Receive Step Source

To create a **RECEIVE** step using the **WF_RECEIVE BUILT-IN WORKFLOW EVENT**:

1. Click the **CONFIGURATION** screen group and click the **WORKFLOWS** icon. The **WORKFLOW WORKBENCH** and **WORKFLOW STEP SOURCES** window open.
2. Select the **WORKFLOW STEP SOURCES** window.



3. Select the EXECUTIONS folder.
4. Click **NEW**. The EXECUTION window opens.



5. Select either **PACKAGES** or **REQUESTS** from the WORKFLOW SCOPE drop down list, depending on the desired application of the Workflow.

6. Select **BUILT-IN WORKFLOW EVENT** from the EXECUTION TYPE drop down list.
7. Select **WF_RECEIVE** from the WORKFLOW EVENT drop down list.
8. Select or create a Validation which will be used to transition out of this Workflow Step.



Note

The Validation values exiting the RECEIVE step must match the possible Validation values entering and exiting the JUMP step.

9. Fill in any other required or optional information (such as NAME, DESCRIPTION OR PROCESSING TYPE).
10. Click the **OWNERSHIP** tab to select which Ownership Groups will have the ability to edit this Execution step.
11. Click **OK**. The Workflow Step is added to the WORKFLOW STEP SOURCES window.

This Workflow Step can now be used in any new or existing Workflow within the step's defined Workflow Scope. Remember that every RECEIVE step must have a paired JUMP step in another Workflow.

Including the Jump/Receive pair in Workflows

1. Drag either the JUMP or RECEIVE step from the WORKFLOW STEP SOURCES window into the Workflow's **LAYOUT** tab. The WORKFLOW STEP window opens.

The screenshot shows the 'Workflow Step' dialog box with the following fields and values:

- Step Number: 1
- Step Name: Create Package Immediate
- Action Button Label: (empty)
- Description: (empty)
- Source Type: Execution
- Source Name: Create Package Immediate
- Enabled: Yes No
- Display: Always
- Jump/Receive Step Label: QA Fix in Development
- Workflow Parameter: NONE
- Source Environment: (empty)
- Source Environment Group: (empty)
- Dest Environment: (empty)
- Dest Environment Group: (empty)
- Save to O*M/GL*M Archive? Yes No
- Avg Lead Time: (empty)
- Request Status: (empty)
- Current % Complete: (empty)
- Parent Assigned To User: (empty)
- Parent Assigned To Group: (empty)
- Workflow Step Information: (empty)

Buttons: OK, Apply, Cancel

Status: Ready

2. Select an item from the JUMP/RECEIVE STEP LABEL drop down list. This item must be the same for a paired Jump/Receive Step.



Note

The JUMP/RECEIVE STEP LABEL is the key communication link between separate Workflows. The communicating Jump and Receive Workflow Steps must have a matching JUMP/RECEIVE STEP LABEL. It is also important that the JUMP/RECEIVE STEP LABEL is unique for any Jump and Receive pair.

3. Enter any additional Workflow Step information.
4. Click **OK**.
5. Repeat the above process for the other paired Workflow Step (Jump or Receive), depending on which one was configured first

Using Condition Steps

Kintana can perform complex routing based on the status of multiple Workflow Steps using Condition steps. There are five Condition steps available:

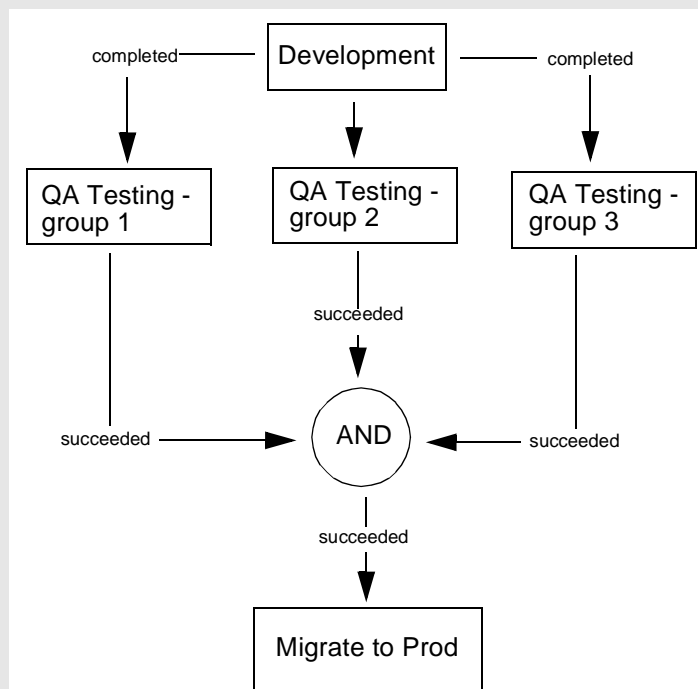
- *AND*
- *OR*
- *SYNC*
- *FIRST LINE*
- *LAST LINE*

AND

An AND condition is satisfied only if all steps leading to it reach the status they are supposed to attain.



The AND step becomes successful only if 'QA Testing - group 1,' 'QA Testing - group 2' and 'QA Testing - group 3' are successful. At that point, the following step 'Migrate to Prod' becomes eligible.

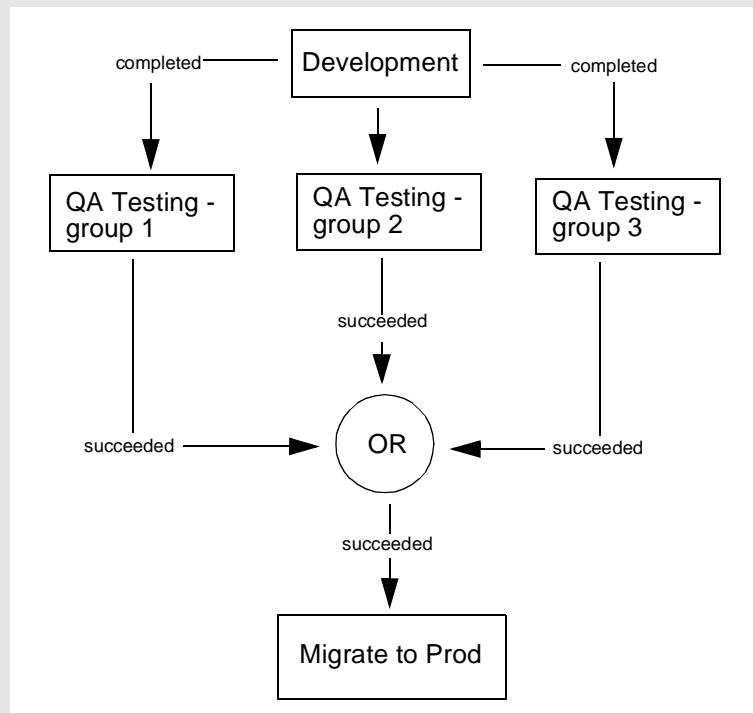


OR

An OR condition is successful when at least one of the steps leading to it reaches the status it is supposed to attain.



The OR step becomes successful if any one of 'QA Testing - group 1,' 'QA Testing - group 2' and 'QA Testing - group 3' is successful. At that point, the following step 'Migrate to Prod' becomes eligible.



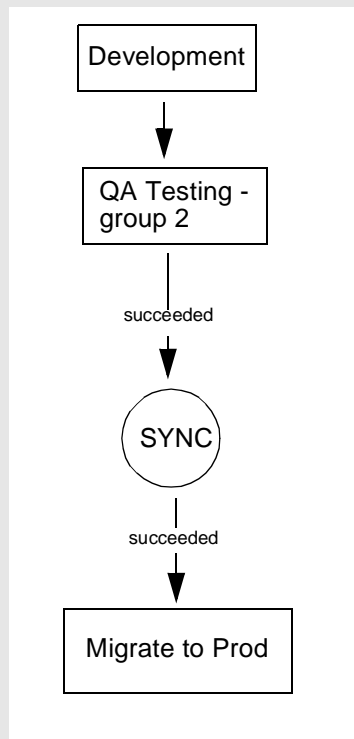
SYNC

A SYNC step is valid only for Kintana Packages. A SYNC step is successful only if all the Package Lines of that Package reach the status expected for the Workflow Step right before the SYNC step.



Consider the business process outlined in the following flow chart. According to the flow chart, when 'QA Testing' is successful for all Package Lines, SYNC becomes successful and the next step, 'Migrate to Prod' becomes eligible.

This business process could be part of a software development life cycle. Consider a case where three Java files are being processed on three respective Package Lines in a single Package. By including a SYNC step, even if the first two Java files pass 'QA Testing,' they must wait for the third Java file to succeed 'QA Testing' before 'Migrate to Prod' becomes eligible for any of these Package Lines.



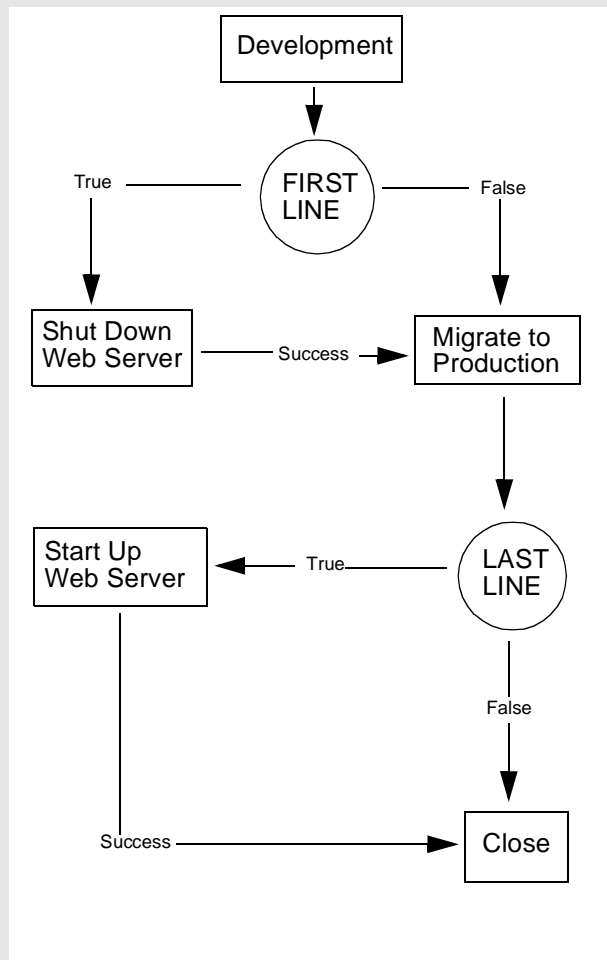
FIRST LINE

A FIRST LINE step is valid only for Kintana Packages. Only the first line to reach the Condition step takes the 'True' transition. All successive lines take the 'False' transition.

Example

Consider the business process outlined by the following flow chart. This business process could be part of a Website maintenance life cycle. As part of this life cycle, three HTML files are being processed on three respective Package Lines in a single Package. The Website updates are large enough to warrant shutting down the Web server while migrating the changes.

By including a FIRST LINE step, only the first line causes the server to shut down. The server remains down while the rest of the changes are migrated to production. By including a LAST LINE step, the server remains down until the last active line reaches the condition step. The last active line takes the True transition and the Web server starts up and the maintenance is complete.



LAST LINE

A LAST LINE step is valid only for Kintana Deliver. Only the last active line to reach the Condition step takes the 'True' transition. All previous lines take the 'False' transition. See the example of a LAST LINE step shown in the previous flow chart.

Setting the Reopen Step for Request Workflows

Closed Requests can be re-opened by users with the proper access grants. A re-opened Request begins at the pre-defined Reopen Step in its Workflow, and begins processing normally.

The Reopen Step is defined from the WORKFLOW window.

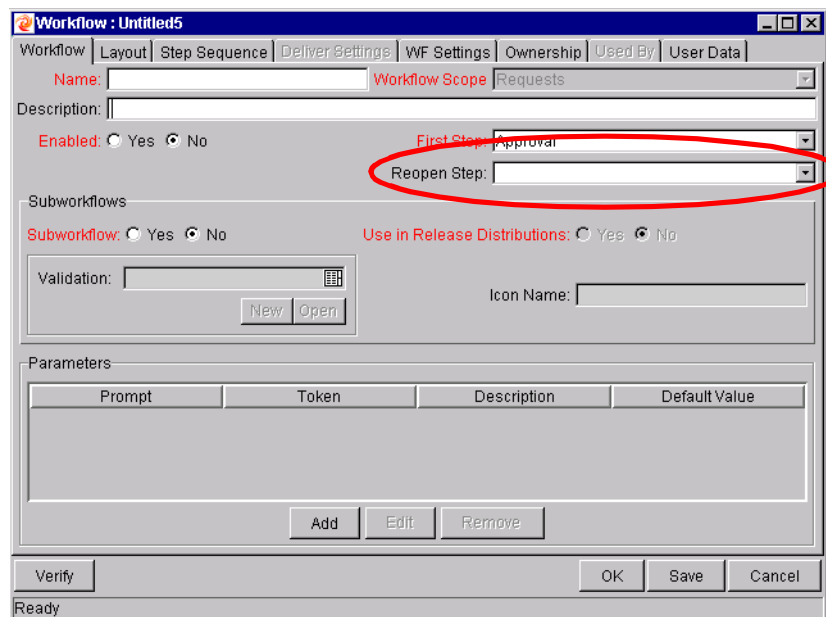


Figure 11-1 Workflow Window Reopen Step Drop Down

To specify the Reopen Step for the Workflow, select the desired step from the REOPEN STEP drop down list.

Modifying Workflows in Use

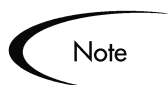
Kintana Workflows can be modified while they are going through their Workflow steps in a Package Line or Request that has been initiated. These modifications include adding new Workflow Steps, as well as changing the transitions, security assignments and notifications from within the Workflow.

You make changes to Workflows currently in use with the same procedures and windows that you used to define the Workflows. All of these procedures are performed in the Workflow Workbench.

When modifying Workflows that are being used, rules exist for which entities can be added, changed, deleted or renamed. These rules are described in [Table 11-9](#).

Table 11-9. Rules for Modifying Production Workflows

Entity	Procedure
Transitions Security Notifications Workflow Steps Workflow Parameters	All of these entities can be modified or added to a Workflow in use.
Transitions Security Notifications Workflow Parameters	All of these entities can be deleted from a Workflow in use.
Workflow Steps	This entity cannot be deleted from a Workflow in use, but can be renamed. Transitions coming into or going out of a Workflow Step can be deleted, effectively removing it from the Workflow.



Note

When a Workflow that is in use is modified and saved, the changes take effect in Kintana immediately. Any changes made to Workflow Steps are applied to all open Package Lines, Requests, and Distributions.

Changes to a Workflow can have undesirable effects on Requests or Packages currently in progress and are using that Workflow.



The information included here also applies when migrating Workflows between installations (instances) of Kintana.

When you modify a Workflow that is in use, this can disrupt the normal flow in and out of the Workflow and prevent it from reaching completion. For example, you might remove a transition from a Workflow Step and find that the Requests or Package Lines are stuck in that Step. While no one solution covers all situations, the following sections describe possible solutions for common problems when modifying Workflows:

- [*Copying and Testing the Workflow*](#)
- [*Moving Requests Out of a Step*](#)
- [*Disabling a Workflow Step*](#)
- [*Setting Up Execution Steps*](#)
- [*Verifying Workflow Logic*](#)

Copying and Testing the Workflow

To modify a Workflow that is being used, make a copy of the original Workflow in a Development environment. Then modify the copied version of the Workflow. Test the copied version of the Workflow to make sure it works correctly.

After verifying that the modified Workflow functions as it is supposed to, make the same changes to the original Workflow and move it through the same cycle of DEVELOPMENT -> TEST -> PRODUCTION environments.

Moving Requests Out of a Step

If your Requests are stuck in a step after you remove a transition from a Workflow in use, add the deleted transition back to the Workflow. After the Requests have flowed out of the step, delete the transition again.

To determine when the Requests have flowed out of the step, run the WORKFLOW DETAIL REPORT. This report indicates if the step you want to delete is eligible for user action or has been completed.



To determine if any Package Lines are Eligible for user action in a Workflow, run the Packages Pending Report.

Disabling a Workflow Step

As mentioned in [Table 11-9](#), you cannot delete a step from a Workflow that is in use; you can only disable it. However, you may want to change the process that is routed through the Workflow. Any changes to the process must be reflected in the Workflow. This will require disabling existing steps and adding new steps.

You can effectively disable a step that you no longer want to use and add a new step by following this process:

1. Remove transitions to the existing Workflow step you no longer want to use.
2. Add a new Workflow step to the Workflow.
3. Redirect the transitions to the new Workflow step.

Redirecting the Workflow

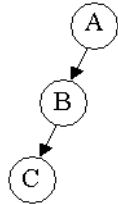
If you disable a Workflow step that is currently ‘Eligible’ for user action, the Requests or Package Lines in that step will become “stuck”. Since the step is now disabled, the user cannot take action on it and will not be able to progress any further through the Workflow.

To determine which steps are currently Eligible, remove the incoming transition to the step you want to delete and then run the Packages Pending Report in Deliver or the Workflow Detail Report in Create. The reports will indicate if the step you want to delete is ‘Eligible’ for action by Package Lines or Requests.

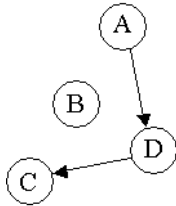
The outgoing transition to be deleted is still intact, so the eligible Package Lines and Requests will eventually be acted upon and flow out of the Workflow step.

Add a new Workflow step to the Workflow and redirect the transitions to that new Workflow step so that the movement of Package Lines and Requests avoids the disabled step and is not interrupted.

For example, consider a Workflow where you wanted to disable Workflow step B in the sequence shown below.



After removing the incoming and outgoing transitions to B, you would add a new Workflow step D which would connect steps A and C and let the Workflow continue to process Requests or Package Lines. See the sequence shown below.



Run the appropriate report(s) again to be sure there are no entities Eligible for action by the user in the step that was disabled.

Setting Up Execution Steps

When setting up Execution steps in a Workflow Step, be sure to include Workflow Events for both **SUCCESS** and **FAILURE**. If a Workflow Step has failed and users cannot select **FAILURE** as one of the Workflow Events, the Workflow will not be able to proceed.

Modifying Workflow Step Security – Performance Consideration

Updating an existing Workflow's step security with a specific configuration can impact system performance. If you add dynamic security to a step (i.e. based on a Standard or User Defined Token) in the **WORKFLOW STEP** window on the **LAYOUT** tab, tables in the Kintana database are updated to handle this new configuration. Because of the scope of database changes, you should re-run the Database Statistics on your Kintana Database. Instructions for this are included in the "Kintana System Administration Guide." Contact your System Administrator for help with this procedure.



This also applies if you migrate a Workflow with these types of changes into an instance of Kintana.

Verifying Workflow Logic

A Workflow can also become stuck if the logic behind it is faulty. Plan the steps of your Workflow process carefully before actually defining it. After configuring your Workflow, click the **VERIFY** button in the Workflow window to ensure that the logic of your Workflow is correct. Any mistakes in the Workflow's logic will be highlighted.

Using Workflow Parameters

You can use Workflow parameters to store the result of a workflow step. This value can then be used later to define a transition.



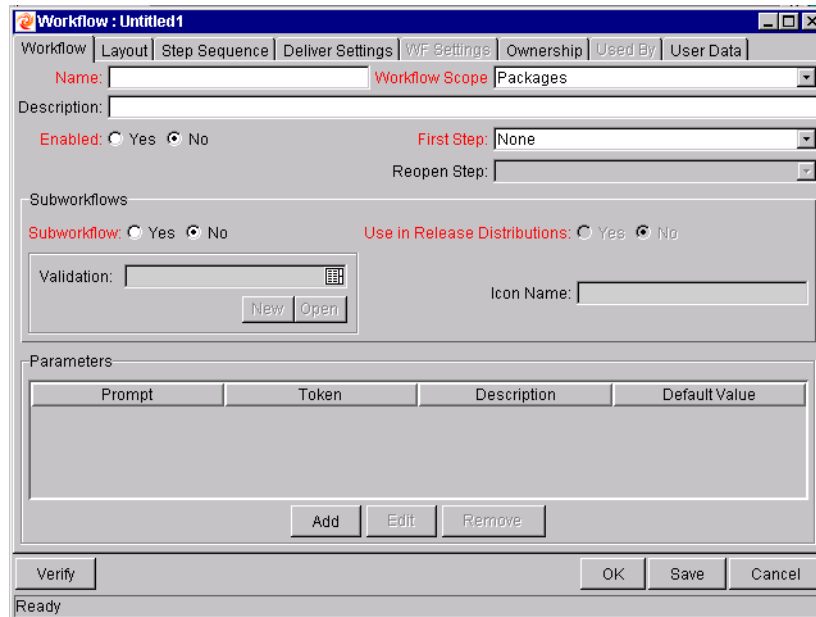
Workflow Parameters:

- Can be referenced using the WFI.P Token prefix.
- Can be used in PL/SQL and SQL Workflow Step executions.

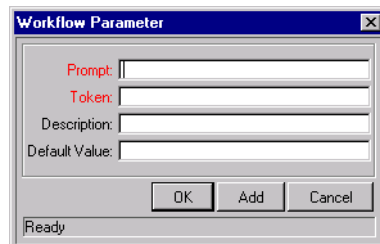
Creating a Workflow Parameter

To create a Workflow parameter:

1. From the WORKFLOW WORKBENCH, query and open the Workflow to be modified.



2. In the **WORKFLOW** tab, click **ADD**. The **WORKFLOW PARAMETER** window opens.

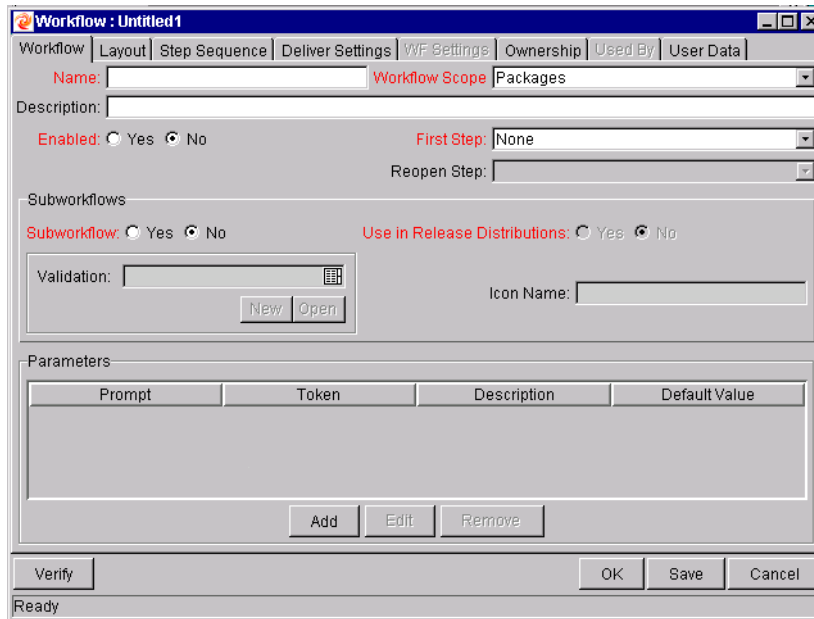


3. Enter information in the required fields.
4. Click **OK**.

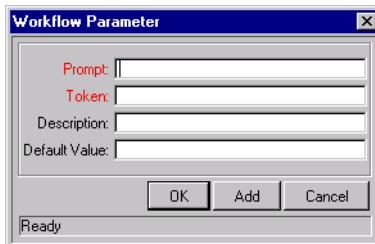
Example: Building a Loop Counter

Workflow parameters can be used to generate a counter for the number of times a Workflow Step is in a certain state. To build a loop counter using Workflow parameters:

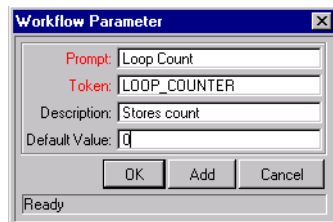
1. From the **WORKFLOW WORKBENCH**, open the Workflow to which the loop counter is to be added.



2. In the **WORKFLOW** tab, click **ADD**. The **WORKFLOW PARAMETER** window opens.



3. Generate the Workflow parameter by entering information in the fields of the **WORKFLOW PARAMETER** window. In this example, the parameter is named **LOOP_COUNTER**.



4. Click **OK**. The **LOOP COUNT** parameter is added to the Workflow window.

Workflow: Untitled6

Workflow | Layout | Step Sequence | Deliver Settings | **WF Settings** | Ownership | Used By | User Data

Name: Workflow Scope: Packages

Description:

Enabled: Yes No First Step: None

Reopen Step:

Subworkflows

Subworkflow: Yes No Use in Release Distributions: Yes No

Validation:

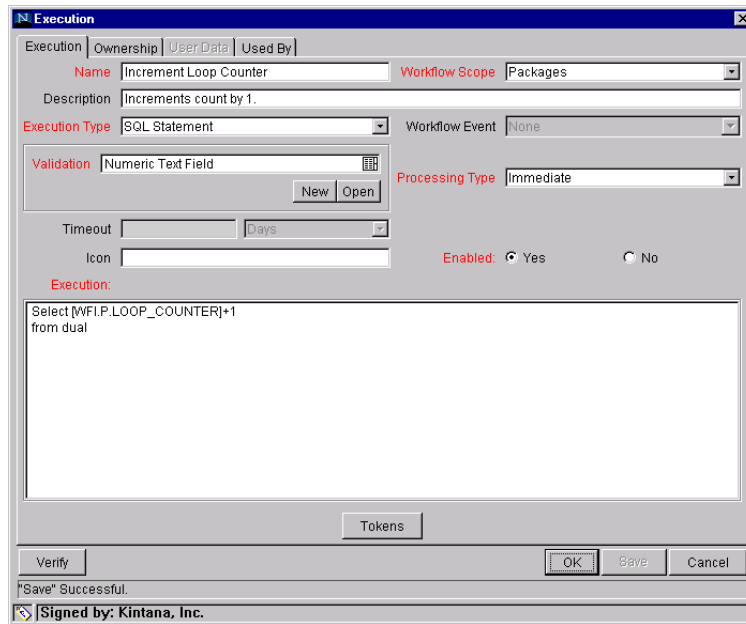
Icon Name:

Parameters

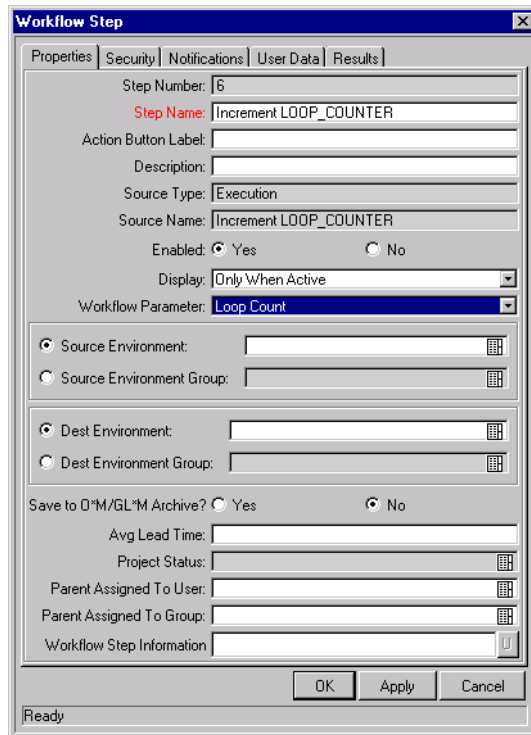
Prompt	Token	Description	Default Value
Loop Count	LOOP_COUNTER	Stores count	0

Ready

5. Generate a new Immediate SQL Execution Step. There are two key concepts to note about the new step definition.
 - The result of the SQL Execution step returns the result **LOOP_COUNTER + 1**. This return value is linked back into the parameter when the Workflow Step is generated on a Workflow.
 - A Validation for a **NUMERIC** Text Field is used. This allows \leq , $<$, \geq , and $>$ comparisons to be used in transitions off this step.



6. Add the Workflow Step to a Workflow and choose the new Workflow Parameter **LOOP_COUNTER**. By choosing **LOOP COUNT**, the Workflow Engine is told to assign the result of “select loop counter val + 1 from dual” back into the loop counter parameter.



It is now possible to add transitions to and from the new loop counter step.



The loop counter can be incremented each time a Kintana Deliver execution fails. If the execution fails three times, a notification can be sent to the user. If the execution fails five times, management can be notified.

Appendix B Validations

This chapter provides an overview for how to use Validations in your Kintana system. Validations determine the acceptable input values for user-defined fields (such as Object Type or Request Type fields). Validations also determine the possible results that a Workflow step can return. This appendix discusses the following topics:

- *What are Validations*
- *Validation Component Types - Overview*
- *Creating a Validation*
- *Editing Validations*
- *Deleting Validations*
- *Static List Validations*
- *Dynamic List Validations*
- *Using Auto-Complete Validations*
- *Using Directory and File Choosers*
- *Creating 1800 Character Text Areas*
- *Configuring the Table Component*
- *Package and Request Group Validations*
- *Validation Special Characters*
- *System Validations*

What are Validations

Validations are used in two main ways in Kintana:

- **Fields:**
Validations determine the field's component type (text field, drop down list, etc.) and the fields possible values. Fields can be created for a number of Kintana entities: Object Types, Request Types, Request Header Types, and User Data.
- **Workflow step results:**
Validations determine the possible results exiting a Workflow step. For example, the validation WF - STANDARD EXECUTION RESULTS contains the possible execution step results of **SUCCEEDED** or **FAILED**.

Kintana provides a number of pre-seeded (system) Validations with every installation or upgrade. When configuring your system, you can select to use these system Validations. If no Validation exists that meets your specific requirements, you can create a new Validation using the VALIDATION WORKBENCH. See "[Creating a Validation](#)" on page 291 for details.

Validation Component Types - Overview

The following table summarizes the types of field components that can be used in Kintana. Note that only certain component types can be used in a Workflow step source's Validation.

Table 11-10. Component Types




Component Type	Use In Workflow?	Example**	Description
Text Field	Yes		Text entry fields displayed on a single line.
Drop down list	Yes		Field showing a column of choices.
Radio Button	No		Field providing a Yes/No input.

Table 11-10. Component Types







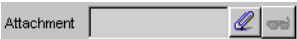





Component Type	Use In Workflow?	Example**	Description
Auto-complete list	Yes		Field showing list of choices with multiple columns.
Text Area	No		Text entry field that can span multiple lines.
Date Field	No		Supports a variety of date and time formats: long, medium, and short.
Web Address (URL)	No		Text entry field for entering a URL. Pressing the U button opens a browser window to the specified web address.
File Chooser	No		Used only in Object Types. Requires that two fields be defined with the following Tokens: P_FILE_LOCATION and P_SUB_PATH. See <i>“Using Directory and File Choosers”</i> on page 312 for configuration details.
Directory Chooser	No		Used only in Object Types. Requires that a parameter field be defined with the Token P_FILE_LOCATION.
Attachment	No		Field for indicating file attachments. Comes with buttons for locating files for previewing contents of the selected file.
Password field	No		Field for capturing passwords.

Table 11-10. Component Types

Component Type	Use In Workflow?	Example**	Description
Table Component	No	<p>Table Component (No Entries) </p>	<p>Used to enter multiple records into a single Kintana component. The table component can be configured to include multiple columns of varied data types. Additionally, this component supports rules for populating elements within the table and provides functionality for capturing column totals. See “Configuring the Table Component” on page 316 for details.</p> <p>Fields of this component can only be added to Request Types, Request Header Types and Request User Data.</p>
Budget	No	<p>Budget (No Budget) </p>	<p>Field that can be added to the Request Type to enable access to view, edit or create Budgets associated with a Request or Project.</p> <p>Fields of this component can only be added to a Request Type.</p>
Staffing Profile	No	<p>(No Staffing Profile) </p>	<p>Field that can be added to the Request Type to enable access to view, edit or create Staffing Profiles associated with a Request or Project.</p> <p>Fields of this component can only be added to a Request Type.</p>
Resource Pool	No	<p>Resource Pool (No Resource Pool) </p>	<p>Field that can be added to the Request Type to enable access to view, edit or create Resource Pools associated with a Request or Project.</p> <p>Fields of this component can only be added to a Request Type.</p>

Creating a Validation

Generating certain Workflow steps may require specific validations to ensure that business procedures are being followed. It is necessary to have both the Validation Editor and the Validation Values Editor access grants to add a new validation. See "[Kintana Security Model](#)" for a discussion of security groups and access grants.

To define a new Validation:

1. Click **NEW VALIDATION** on the VALIDATION WORKBENCH or select **FILE -> NEW -> VALIDATION** from the menu. The VALIDATION window opens.
2. Enter the name of the new Validation in the NAME field.
3. Enter a description of the new Validation in the DESCRIPTION field.
4. Select whether the Validation is enabled or not in the ENABLED check box.
5. In the USE IN WORKFLOW checkbox, specify whether or not this Validation can be used in a Workflow step source. You can only use Text Field, Drop Down List and Auto-Complete component types within Workflow step sources.
6. Select the desired type of Validation from the COMPONENT TYPE drop down list. Enter any additional information required for the component type selected. See the Validation chapter in "[Configuration Workbench Reference](#)" for descriptions of the fields required to define each component type.
7. Click **OWNERSHIP** to select which users will be able to edit, copy and delete this validation.
8. To save changes to the Validation without closing the window, click **SAVE**. To save changes and close the window, click **OK**.

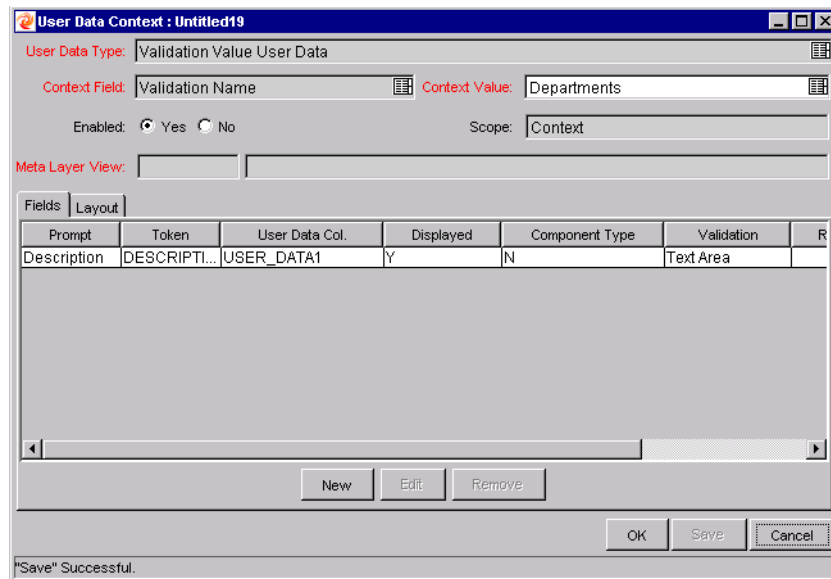
User Data on the Validation Value

You can enable the **USER DATA** tab to capture more information related to an individual Validation value within a specific Validation. For example, you can create a DESCRIPTION user data field that is associated with the DEPARTMENTS Validation. When you add new values to the validation, you can click on the **USER DATA** tab and enter a description for that value.

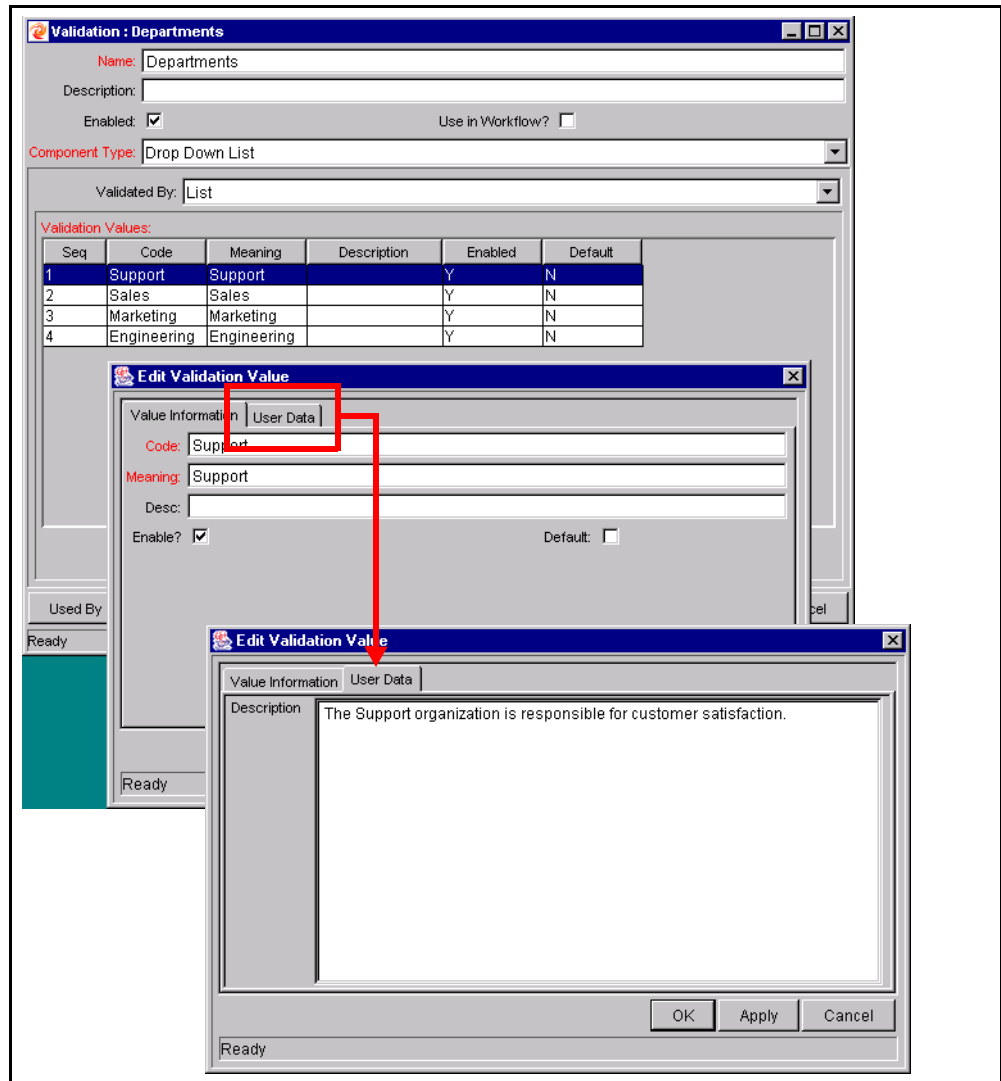
The **USER DATA** tab can only be used when creating a drop down or an auto-complete validated by a list.

To enable the **USER DATA** tab in the EDIT VALIDATION VALUE window:

1. Create the Validation and note its name.
2. Open the USER DATA workbench.
3. Click **NEW USER DATA CONTEXT**.
4. Select **VALIDATION VALUE USER DATA** from the USER DATA TYPE field.
5. Click **NEW** to create a User Data field.



6. Save the settings in the USER DATA window.
7. On the VALIDATION window, add or edit a Validation value. The **USER DATA** tab is now enabled. You can select the tab and enter information in the newly defined user data field.



See the User Data appendix in *"Configuring a Request Resolution System"* for more details on using User Data in Kintana.

Editing Validations

You can open and edit Validations using the Kintana Workbench. You should exercise caution when editing Validations that are currently used by fields or

Workflow step sources. Both field and Workflow step validations can be tied to Workflow logic. Changing the Validation values can invalidate a process.

For example, ACME changes the PRIORITY field Validation to include a new value **VERY EASY**. ACME uses a deployment system Workflow that has an EVALUATE PRIORITY step that routes the Package based on the value in the Priority field (using a Token execution type). ACME, however, did not update the Workflow to enable a transition out of the step for the case when PRIORITY = **VERY EASY**. When a **VERY EASY** Package enters the EVALUATE PRIORITY step, it will get stuck.

The following restrictions apply to editing Validations:

- User must have the following Access Grants:
 - Edit Validations
 - Edit Validation Values
- User must be a member of the Ownership Group for the Validation
- You can not change which Validation is associated with a Workflow step source after a Package has traversed that step. You can, however, still edit the values within that Validation.

Creating a URL to Open the Validation Window

You can create a URL that opens a specific Validation in the Kintana Workbench. This can provide a quick link to the configuration screen for a Validation that is expected to change frequently. This URL can be included on your internal or external Web pages or a list of browser Favorites to provide convenient access to the Validation's definition.

Use the following URL format to access a specific VALIDATION window:

```
http://host:port/kintana/servlet/SmartURL?screen=VAL&pkname=<ValidationName>
```



Note

The following URL opens the VALIDATION window for the Validation named "Development Priorities."

```
http://host:port/kintana/servlet/SmartURL?screen=VAL&pkname=Development+Priorities
```


Deleting Validations

Validations can be deleted from the Kintana Workbench. To delete a Validation, you must be a member of the Validation's Ownership Group and have the EDIT VALIDATIONS access grant.

A Validation can not be deleted when:

- It is a system Validation (a Validation that is delivered with Kintana as seed-data)
- It is being used by a Workflow step source. Validations referenced by Workflow step sources can only be disabled. A disabled Validation continues to function in existing Workflow steps, but can not be used when defining a new step source.
- It is being used by a field in a Kintana entity (Object Type, Request Type, User Data, Report Type, or Project Template field). Validations referenced by entity fields can only be disabled. A disabled Validation continues to function in existing fields, but can not be used when defining a new field.



Tip

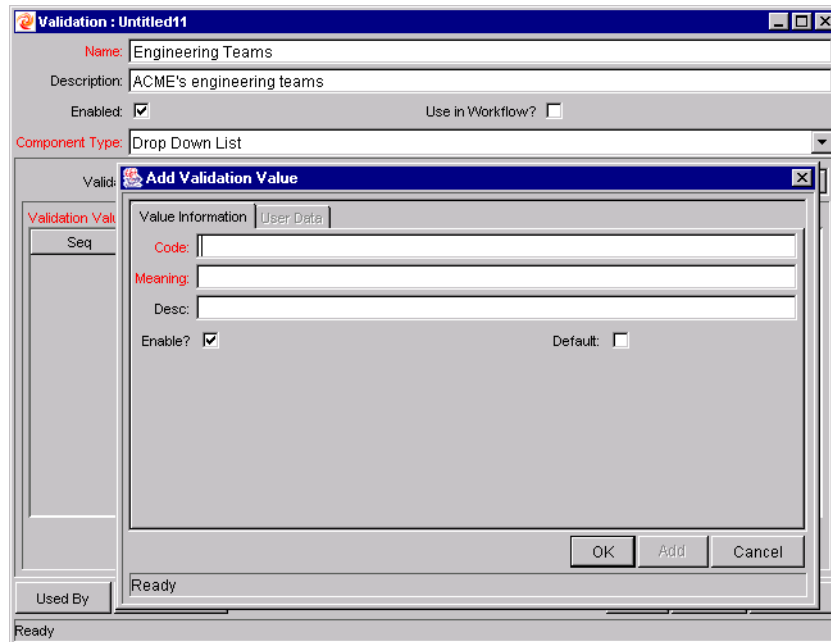
Although you may not be able to delete a custom Validation in all cases, you can disable it. This will allow the Validation to be used in any active Workflows or Kintana entities, but will keep it from being used in any new Workflow or entity definitions.

Static List Validations

You can create Validations that provide a static list of options to the user. For example, ACME, Inc. can create a Validation for their engineering teams. They create a Validation called ENGINEERING TEAMS, consisting of the following values: **NEW PRODUCT INTRODUCTION**, **PRODUCT ONE**, and **PRODUCT TWO**.

A static list validation can be a drop down or an auto-complete list component. To add values to the Validation list:

1. In the VALIDATION window, select **DROP DOWN LIST** or **AUTO COMPLETE LIST** from the COMPONENT TYPE field.
2. Select **LIST** from the VALIDATED BY field.
3. Click **NEW** and add a value. The ADD VALIDATION WINDOW opens.



4. Enter the CODE, MEANING and DESCRIPTION of the value. See "[Configuration Workbench Reference](#)" for definitions of these fields.
5. Optionally set the Validation value as the default by checking the DEFAULT field. The default option is only available for drop down lists.
6. Click OK to close the window and add the value to the Validation. Click **ADD** to add the value and keep the ADD VALIDATION VALUE open.

Validation values can be re-ordered using the up and down arrow buttons. The sequence of the Validation values determines the order that the values are displayed in the list.



Tip

You can copy existing values defined in other Validations using the **COPY FROM** button. Click **COPY FROM** and query an existing list-validated Validation and choose any of the Validation values. Click **ADD** or **OK** in the COPY FROM window and the selected value or values are added to the list.



Note

Be careful when creating Validations (drop down lists and auto-complete lists) that are validated by lists. Each time the set of values changes, you will be forced to update the Validation. Consider, instead, validating using a SQL query or PL/SQL function to obtain the values from a database table.

Dynamic List Validations

You can create Validations that provide a dynamic list to the user. This is often a better approach than defining static list validations. Each time a static list Validation needs to be updated, a manual update has to occur. Dynamic list Validations can often be constructed in such a way as to automatically pick up and display the altered values.

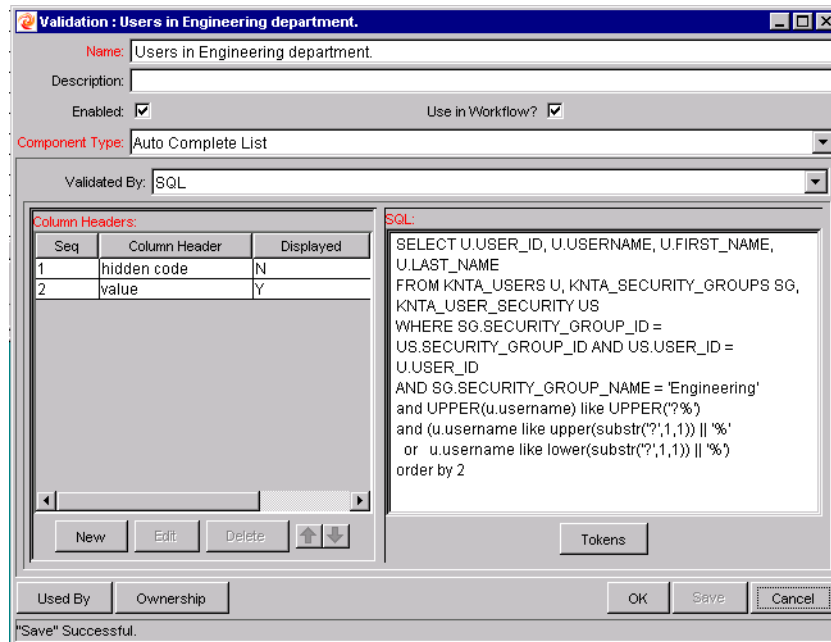
For example, ACME needs a field validation that will list all Kintana users who are on their Support Team. They could construct a Validation that is validated by a list of users, but any time the Support Team changed (members join or leave the department) the list would have to be manually updated. ACME decides instead to create a dynamic list validation. They create an auto-complete list validation that is validated by a SQL statement. The SQL statement returns all users who are a member of the **SUPPORT TEAM** Security Group. When the Security Group membership is altered, the Validation is automatically updated with the correct values.

A dynamic list validation can be created using a drop down or an auto-complete list component. The lists are dynamically generated using either:

- [SQL Validation](#)
- [Command Validation](#)

SQL Validation

You can use a SQL statement to generate the values in a Validation. SQL can be used as a validation method for drop down lists and auto-complete lists. To define a dynamic list of choices, set a drop down list or auto-complete list to VALIDATED BY - **SQL**. Then in the SQL area, enter the Select statement that queries the necessary database. See "[Configuration Workbench Reference](#)" for an explanation of each screen and field in the VALIDATION window.



Example

ACME, Inc. creates an auto-complete field that lists all Kintana users in the “Engineering” department. They choose to validate the list by SQL.

```
SELECT U.USER_ID, U.USERNAME, U.FIRST_NAME, U.LAST_NAME
FROM KNTA_USERS U, KNTA_SECURITY_GROUPS SG,
KNTA_USER_SECURITY US
WHERE SG.SECURITY_GROUP_ID = US.SECURITY_GROUP_ID AND
US.USER_ID = U.USER_ID
AND SG.SECURITY_GROUP_NAME = 'Engineering'
and UPPER(u.username) like UPPER('?%')
and (u.username like upper(substr('?',1,1)) || '%'
or u.username like lower(substr('?',1,1)) || '%')
order by 2
```

When a new user is added to Kintana and included in the “Engineering” Security Group, that user will automatically be included in the auto-complete list.



Tip

Kintana may already have a Validation that meets your process requirements. If it does, consider using that Validation in your process. Also consider copying and modifying Validations that are similar to the desired Validation. See *“System Validations”* on page 331 for a complete list of Validations that are delivered with Kintana.

SQL Validation Tips

The following guidelines are helpful when writing a SQL statement for a SQL-validated Validation:

- The SQL statement must query at least two columns. The first column is a hidden value which is never displayed, and is often stored in the database or passed to internal functions. The second column is the value that is displayed in the field. All other columns are for information purposes and are only displayed in the auto-complete window. Extra columns are not displayed for drop down lists.
- When something is typed into an auto-complete list field, the values in the auto-complete window that appear are constrained by what was first typed in the field. Generally, the constraint is case insensitive. This is accomplished by writing the SQL statement to query only values that match what was typed.

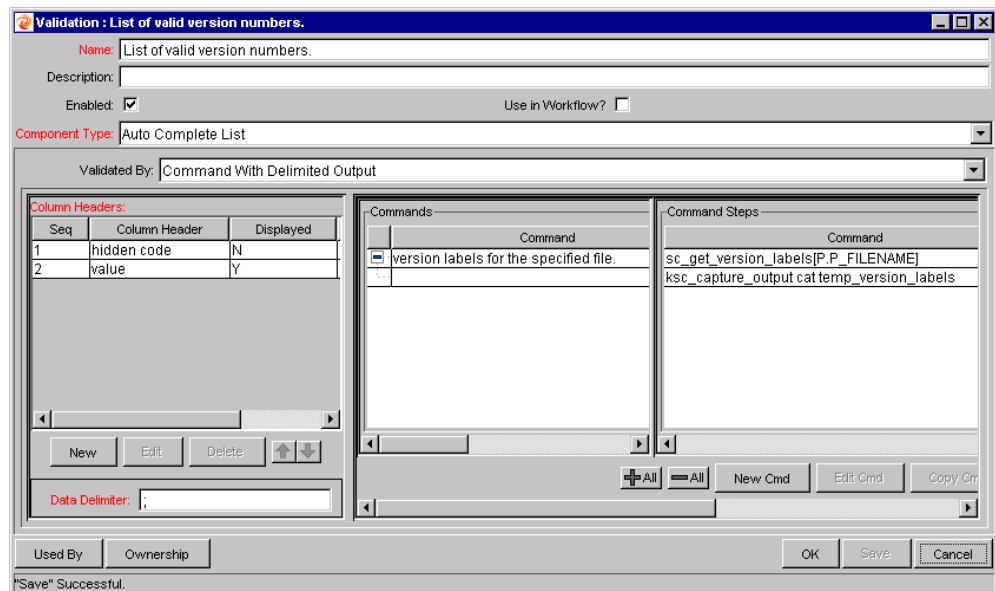
Before the auto-complete window is displayed, all question marks in the SQL statement are replaced by the text that the user typed. In general, if the following conditions are added to the WHERE clause in a SQL statement, the values in the auto-complete window are constrained by what the user typed.

```
where UPPER(<displayed_column>) like UPPER('?%')
and (<displayed_column> like upper(substr('?',1,1)) || '%')
or <displayed_column> like lower(substr('?',1,1)) || '%')
```

Any column aliases included directly in the SQL statement are not used. The names of the columns, as displayed in auto-complete lists, are determined from the Column Headers. Drop down lists do not have column headers

Command Validation

An auto-complete list can contain command line executions that return and display a list of values. To define a dynamic list of choices, set an auto-complete list to VALIDATED BY - **COMMAND WITH DELIMITED OUTPUT** or **COMMAND WITH FIXED WIDTH OUTPUT**. Then enter commands the COMMANDS area.



Using Auto-Complete Validations

The values in an auto-complete list can be specified in the following ways. In the VALIDATE BY field, select one of the following:

- **LIST:** Used to enter specific values.
- **SQL:** Uses a SQL statement to build the contents of the list.
- **COMMAND WITH DELIMITED OUTPUT:** Uses a system command to produce a character-delimited text string and uses the results to define the list.
- **COMMAND WITH FIXED WIDTH OUTPUT:** uses a system command to produce a text file and parses the result on the basis of the width of columns, as well as the headers.

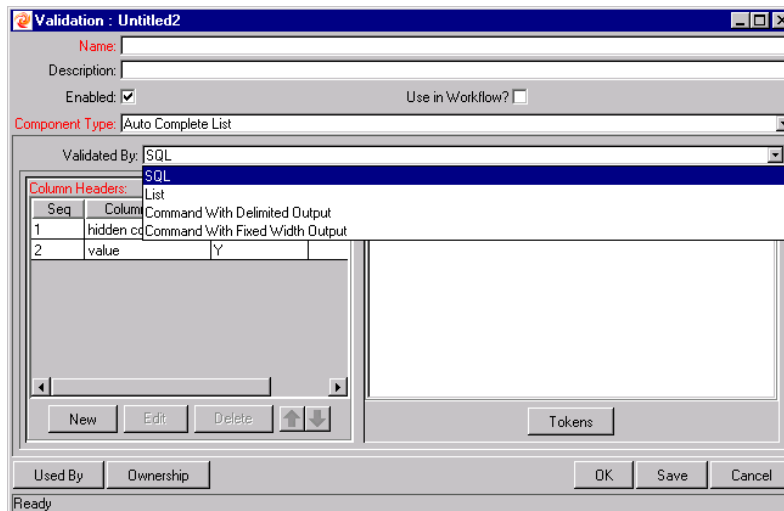


Figure 11-2 Auto-Complete List

The following sections discuss the following topics:

- [Validation by Command With Delimited Output](#)
- [Validation by Command With Fixed Width Output](#)
- [User-Defined Multi-Select Auto-Complete Fields](#)
- [Example: Token Evaluation and Validation by Command with Delimited Output](#)
- [Special Case - Limiting the Number of Returned Rows](#)

For more information on creating auto-completes validated by List or SQL, refer to the following sections:

- [“Static List Validations”](#) on page 295
- [“Dynamic List Validations”](#) on page 297

Validation by Command With Delimited Output

Validations by Command with Delimited Output can be used to get data from an alternate source, and use that data to populate an auto-complete field. This functionality provides additional flexibility when designing auto-complete lists.

Many enterprises need to use alternate sources of data within their applications. Examples of these sources are a flat file, an alternate database source, or output from a command line execution. Special commands may be used in conjunction with these alternate data sources, in the context of a Validation, to provide a list of values.

To configure a validation by command with delimited output:

1. In the VALIDATION WORKBENCH, under VALIDATED BY, choose **COMMAND WITH DELIMITED OUTPUT** and input the delimiting character.
2. Under NEW COMMAND, enter in the command steps to be executed. These can include Kintana Special Commands. Your commands should include the Special Command `ksc_capture_output`, which captures and parses the delimited command output. If the `ksc_capture_output` Special Command is surrounded by the `ksc_connect` and `ksc_disconnect` commands, the command will be run on the remote system. Otherwise, the command will be run locally on the Kintana server (similar to `ksc_local_exec`).



Example

The simple example below uses a comma for a delimiter and has the validation values red, blue and green. The script places the validations into the `newfile.txt` file, and then uses the Special Command `ksc_capture_output` to process the text of the file.

```
ksc_begin_script [AS.PKG_TRANSFER_PATH]newfile.txt
red,red
blue,blue
green,green
ksc_end_script
ksc_capture_output cat [AS.PKG_TRANSFER_PATH]newfile.txt
```

Table 11-11 shows the VALIDATION window for COMMAND WITH DELIMITED OUTPUT.

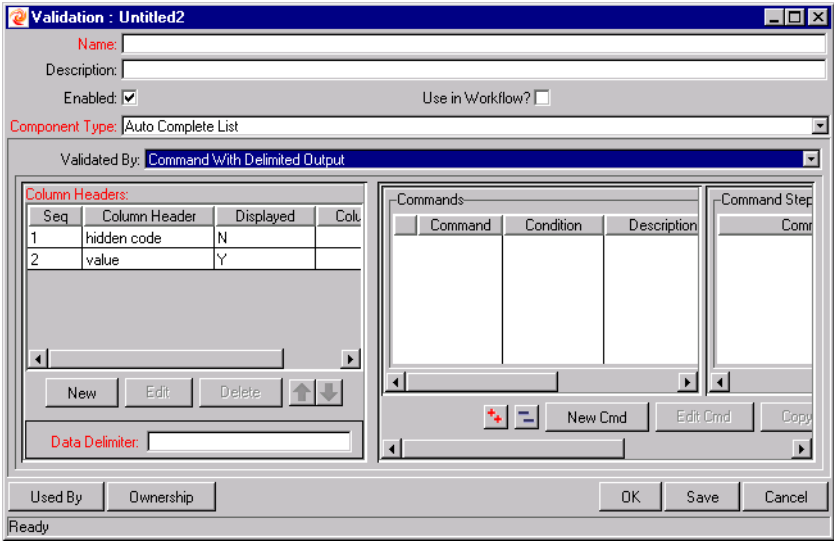


Figure 11-3 Validation by Command with Delimited Output

Table 11-11. Validation by Command With Delimited Output

Field	Definition
COMMAND PANEL	Panel where new commands can be added to capture Validation values.
DATA DELIMITER	Indicates the character or key by which the file will be separated into the Validation columns.

Headers can also be defined for the columns selected. These column headers are used in the window that opens when a value is selected from an auto-complete list. To define a new header, click **NEW** under COLUMN HEADER. [Table 11-12](#) shows the fields that can be entered for a column header. If a column header is not defined for each column in a Command, a default name is used.

Table 11-12. Column Headers

Field	Definition
COLUMN HEADER	The name of the column that is displayed in the auto-complete window.
DISPLAY	Determines whether or not the header is displayed in the Validation.

Validation by Command With Fixed Width Output

Validations by `COMMAND WITH FIXED WIDTH OUTPUT` can be used to obtain data from an alternate source, and use that data to populate an auto-complete field. This functionality provides additional flexibility when designing auto-complete lists.

Many enterprises need to use alternate sources of data within their applications. Examples of these sources are a flat file, an alternate database source, or output from a command line execution. Special commands may be used in conjunction with these alternate data sources, in the context of a Validation, to provide a list of values on the fly.

In the `VALIDATION WORKBENCH`, under `VALIDATED BY`, choose **COMMAND WITH FIXED WIDTH OUTPUT** and input the appropriate width information.

Then, under `NEW COMMAND`, enter in the command steps to be executed. These can include Kintana Special Commands. Your commands should include the Special Command `ksc_capture_output`, which captures and parses the delimited command output. If the `ksc_capture_output` Special Command is surrounded by the `ksc_connect` and `ksc_disconnect` commands, the command will be run on the remote system. Otherwise, the command will be run locally on the Kintana server (similar to `ksc_local_exec`).



The example below has the validations red, blue and green. The column width is set to a value of 6. The script places the validations into the `newfile.txt` file.

```
ksc_begin_script [AS.PKG_TRANSFER_PATH]newfile.txt
red      red
blue     blue
green    green
ksc_end_script
ksc_capture_output cat [AS.PKG_TRANSFER_PATH]newfile.txt
```

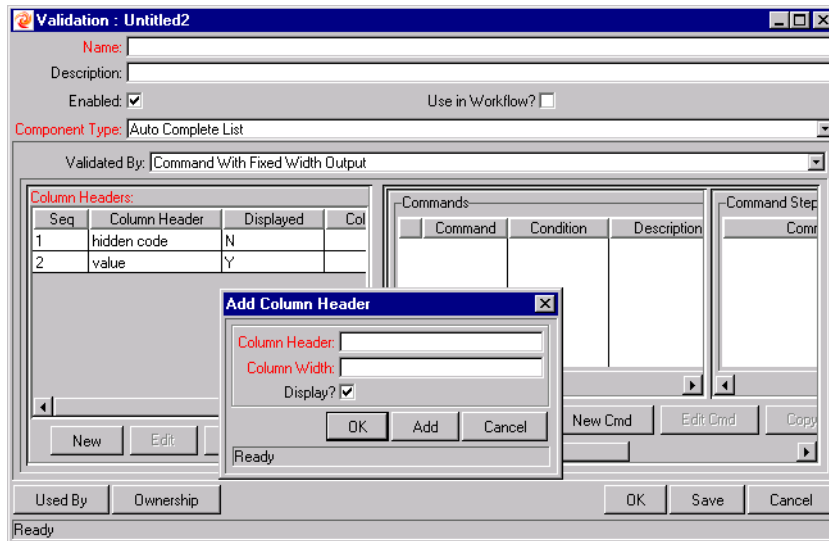


Figure 11-4 Validation by Command with Fixed Width Output

Table 11-13. Validation by Command With Fixed Width Output

Field	Definition
COMMAND PANEL	The panel where new commands can be added to capture Validation values.

Headers can also be defined for the columns selected. These column headers are used in the window that opens when a value is selected from an auto-complete list. To define a new column header, click **NEW** under COLUMN HEADER. [Table 11-14](#) shows the fields can be entered for a column header. If a column header is not defined for each column in a Command, a default name is used.

Table 11-14. Column Headers

Field	Definition
COLUMN HEADER	The name of the column that is displayed in the Auto Complete dialog.
DISPLAY	Whether or not the column is displayed. The first column is never displayed and the second column is always displayed.
COLUMN WIDTH	The number of characters in each column of the output generated as a result of the command.

User-Defined Multi-Select Auto-Complete Fields

A number of auto-complete fields in the Workbench have been configured by Kintana to allow users to open a separate window for selecting multiple values from a list. Users can also define custom auto-complete fields to have multi-select capability when creating various Kintana entities.

The user-defined multi-select capability is supported for:

- User Data fields
- Report Type fields
- Request Type fields
- Project Template fields

The user-defined Multi-Select capability is **not** supported for:

- Request Header Types
- Object Types

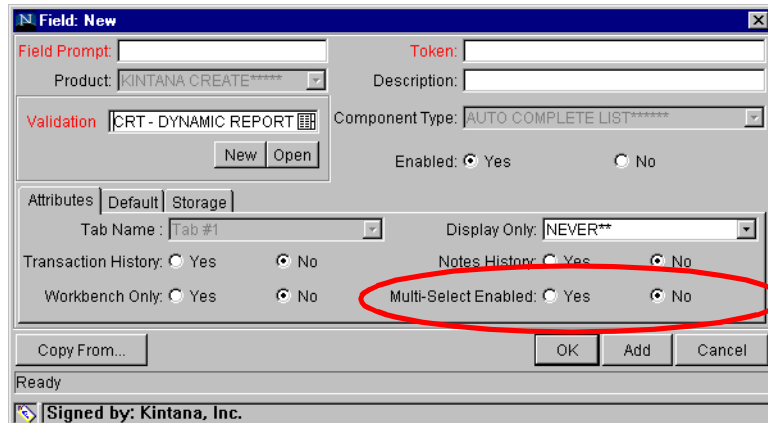
In order to use this feature when creating a new entity, users must:

- Select a Validation for the new entity that has **AUTO-COMPLETE LIST** as the Component Type. This enables the **MULTI-SELECT ENABLED** field in the **FIELD: NEW** window.
- In the **FIELD: NEW** window, users must click **YES** for the **MULTI-SELECT ENABLED** radio button.

The step-by-step procedure for defining multi-select capability in User Data, Report Type, Request Type Project Template fields is very similar. The procedure for enabling this capability for Request Type field is shown below as an example.

To define a multi-select auto-complete field for a Request Type:

1. Click the **CREATE** screen group and click the **REQUEST TYPES** screen. The **REQUEST TYPE WORKBENCH** opens.
2. Click **NEW REQUEST TYPE**. The **REQUEST TYPE** window opens.
3. Click **NEW**. The **FIELD: NEW** window opens.



4. Click the auto-complete icon for the **VALIDATION** field. The VALIDATE window opens.
5. In the VALIDATE window, select a Validation that has **AUTO-COMPLETE LIST** as the Component Type.
6. Click **OK** in the VALIDATE window. The VALIDATE window closes.

The **VALIDATION** field is populated with the selection from the VALIDATE window. The **MULTI-SELECT ENABLED** option is now enabled.

7. Click the **YES** radio button for the **MULTI-SELECT ENABLED** option.
8. The **POSSIBLE CONFLICTS** window opens. It warns you not to use a multi-select auto-complete for Advanced Queries, Workflow Transitions and Reports. If this field is not going to be used in Advanced Queries, Workflow Transitions or Reports, click **YES** to continue.
9. Configure the other options in this window for the new Request Type.
10. Click **OK**.

The field is now enabled for multi-select auto-complete.

Example: Token Evaluation and Validation by Command with Delimited Output

The Validation functionality can be extended to include field dependent token evaluation. Validations can be configured to dynamically change, depending on the client-side value entered in another field.

To use field dependent token evaluation, it is necessary to configure a Validation in conjunction with an Object Type, Request Type, Report Type, Project Template, or User Data definition. Consider the following example for setting up an Object Type using field dependent tokens.

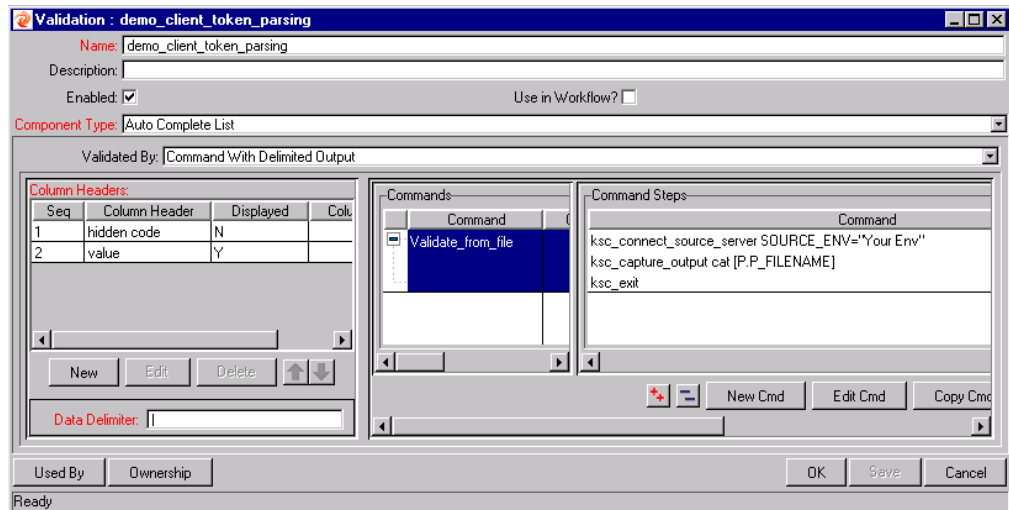
1. Generate a Validation and set the following parameters as shown:

- a. NAME: **DEMO_CLIENT_TOKEN_PARSING**
- b. COMPONENT TYPE: **AUTO COMPLETE LIST**
- c. VALIDATED BY: **COMMAND WITH DELIMITED OUTPUT**
- d. DATA DELIMITER: | (bar)
- e. COMMAND

o COMMAND: **VALIDATE_FROM_FILE**

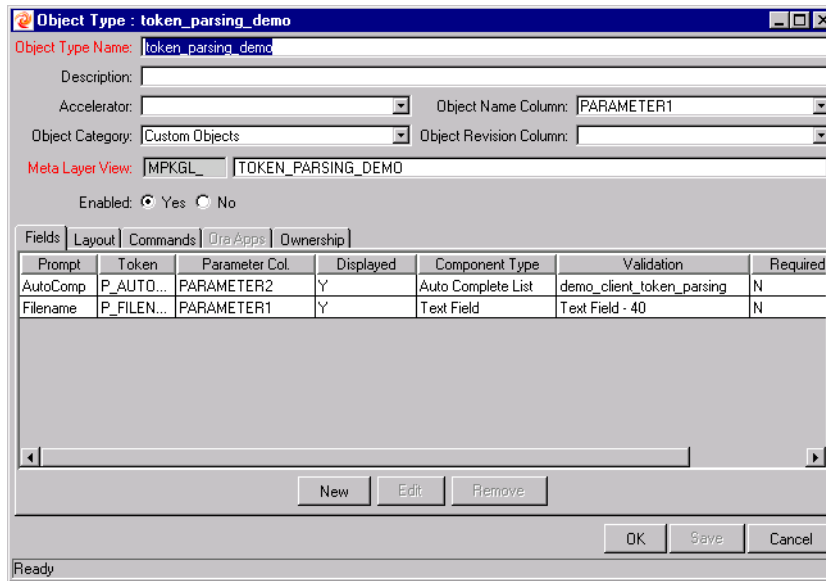
o STEPS

```
ksc_connect_source_server SOURCE_ENV="Your Env"  
ksc_capture_output cat [P.P_FILENAME]  
ksc_exit
```



When called, this Validation will connect to an Environment called 'YOUR ENV' and retrieve data from a file specified by the token P_FILENAME. The file should be located in the directory specified in the BASE PATH in the ENVIRONMENT window.

2. Generate an Object Type named **TOKEN_PARSING_DEMO**.



- a. Generate a new field with the following parameters:
 - o NAME: **FILENAME**
 - o TOKEN: **P_FILENAME**
 - o VALIDATION: **TEXT FIELD - 40**

- b. Generate a new field with the following parameters:
 - o NAME: **AUTOCOMP**
 - o TOKEN: **P_AUTOCOMP**
 - o VALIDATION: **DEMO_CLIENT_TOKEN_PARSING** (this is the Validation that was defined above)

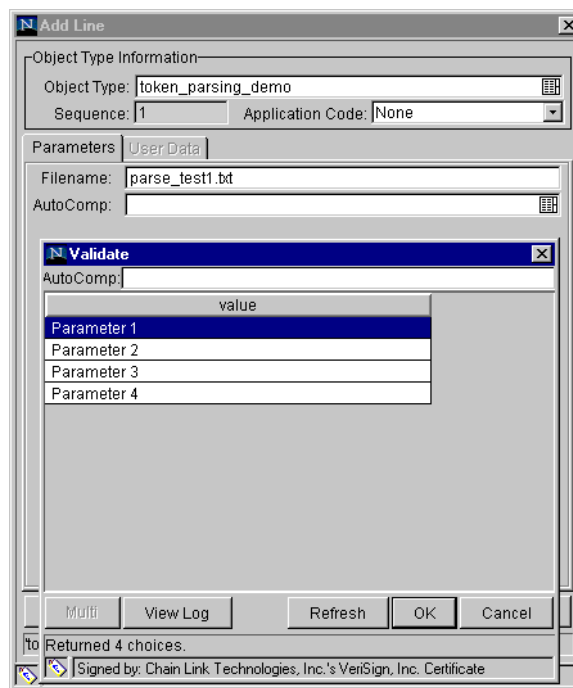
3. For this example to return any values in the auto-complete, a file must be generated in the directory specified in the Base Path in the Environment Detail of 'YOUR ENV' Environment. Generate a file named 'parse_test1.txt' with the following delimited data:

```

DELIMITED_TEXT1 | Parameter 1
DELIMITED_TEXT2 | Parameter 2
DELIMITED_TEXT3 | Parameter 3
DELIMITED_TEXT4 | Parameter 4
    
```

The Object Type 'token_parsing_demo' is now enabled to use this token evaluation. To test the above configuration sample:

1. Generate a new Package.
2. Select a Workflow and click **ADD LINE**.
3. Select **TOKEN_PARSING_DEMO** from the OBJECT TYPE drop down list. The following fields are displayed:
 - FILENAME
 - AUTOCOMP
4. Type 'parse_test1.txt' in the FILENAME field.
5. Click on the auto-complete box in the AUTOCOMP field. The following VALIDATION window opens, displaying the contents of the 'parse_test1.txt' file.



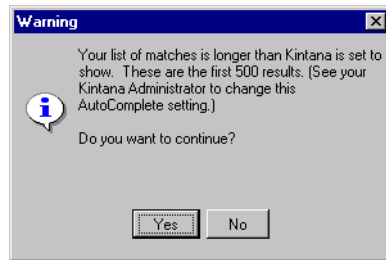
Special Case - Limiting the Number of Returned Rows

Fields that must be validated against a list of pre-defined values use auto-complete lists. When users type a partial value into an auto-complete field and try to tab out of the field, an auto-complete list opens that shows all values

matching the partial value. If no values match the partial value entered, the full list is returned.

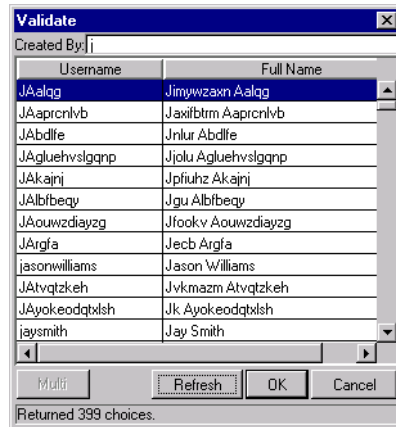
The number of results returned in the list affects how quickly the results are displayed. The larger the number of results, the longer it takes to load. When the number of returned results becomes very large, it becomes advantageous to narrow the search by typing in some limiting text. For example, if you type 'm' into the ASSIGNED TO USER field in the PACKAGE window, a list of all users whose Username starts with 'm' appears.

The auto-complete component can be configured to provide a warning when the number of rows returned from a search will exceed a certain amount. For example, assume that you have 2000 entries in a CREATED BY auto-complete list. When you click on the CREATED BY auto-complete list (without limiting the search), Kintana will display the following message:



You can then either:

- Click **YES** to display the first 500 (or other preconfigured number) results. You can then refine your search in the auto-complete's validate window. For example, if you are searching for a user named 'John Smith,' type 'J' to limit the search to only those users whose names start with the letter 'J.' You may have to click the **REFRESH** button if 'J' was not included in the first 500 rows.



- Click **No** to abort the auto-complete search.

Kintana System Administrators can choose at which number of rows the warning will appear. This is set by creating a `MAX_AUTOCOMPLETE_ROWS` parameter in the `server.conf` file located in the `<KNTA_Home>` directory on the Kintana server.

To enable this auto-complete feature, type the following text into your `server.conf` file:

```
com.kintana.core.server.MAX_AUTOCOMPLETE_ROWS=X
```

where `X` is the number of rows above which the warning will appear. The auto-complete is set to `MAX_AUTOCOMPLETE_ROWS=500` by default.

For more information on setting up the `server.conf` parameters, refer to the “Kintana System Administration Guide” available on the Kintana Download Center.



Note

This is a site-wide setting.

Using Directory and File Choosers

Directory and File Choosers are only used with Object Types. The following sections discuss them in more detail:

- [Directory Chooser](#)

- *File Chooser*

Directory Chooser

The DIRECTORY CHOOSER field can be used to select a valid directory from an Environment. Kintana Deliver connects to the first Source Environment on a Workflow and allows navigation through the directory structure and the selection of a directory from the list.

- The Directory Chooser field can only be used on an Object Type in Kintana Deliver.
- On every Object Type that a Directory Chooser is chosen, it is also necessary to have a field whose token is 'P_FILE_LOCATION' and whose validation is 'Kintana Deliver - File Location'. The possible values for this field are **CLIENT** and **SERVER**. If **CLIENT** is chosen, the Directory Chooser connects to the Client Base Path of the Source Environment. If **SERVER** is chosen, the Directory Chooser connects to the Server Base Path of the Source Environment.

File Chooser

A FILE CHOOSER field can be used by Object Types to select a valid file from an Environment. Kintana connects to the first Source Environment on a Workflow and provides the ability to view all files within a specific directory and select one from the list.

On every Object Type that a File Chooser is chosen, it is necessary to have two other fields defined.

1. The first is a field for the File Location for the directory chooser, described in the previous section.
2. The second is a field whose token is 'P_SUB_PATH'. This field is the directory from which the file is selected and is usually a Directory Chooser field.

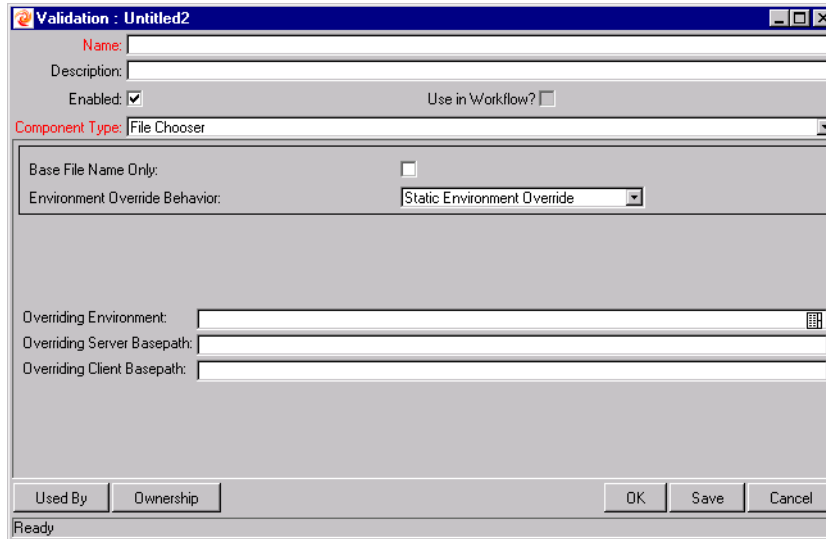


Figure 11-5 Validation Window for Static Environment Override in File Chooser.

Table 11-15. File Chooser Field

Field	Definition
BASE FILE NAME ONLY	Defines whether the base file name only (without its suffix) or the complete name is displayed.
ENVIRONMENT OVERRIDE BEHAVIOR	Used to select files from a specific environment other than the default environment.

The ENVIRONMENT OVERRIDE BEHAVIOR drop down list contains three options: **DEFAULT BEHAVIOR**, **STATIC ENVIRONMENT OVERRIDE**, and **TOKEN-BASED ENVIRONMENT OVERRIDE**.

STATIC ENVIRONMENT OVERRIDE provides the ability to override one Environment at a time. The fields for Static Environment Override are pictured in [Figure 11-5](#) and described in [Table 11-16](#).

Table 11-16. Static Environment Override

Field	Definition
OVERRIDING ENVIRONMENT	Selects the Environment to be overridden.
OVERRIDING SERVER BASEPATH	The server basepath of the Environment may be overridden.

Table 11-16. Static Environment Override

Field	Definition
OVERRIDING CLIENT BASEPATH	The client basepath of the Environment may be overridden.

TOKEN-BASED ENVIRONMENT OVERRIDE provides the ability to select a token that will resolve to the overriding Environment. The fields for **TOKEN-BASED ENVIRONMENT OVERRIDE** are shown in *Figure 11-6* and defined in *Table 11-17*.

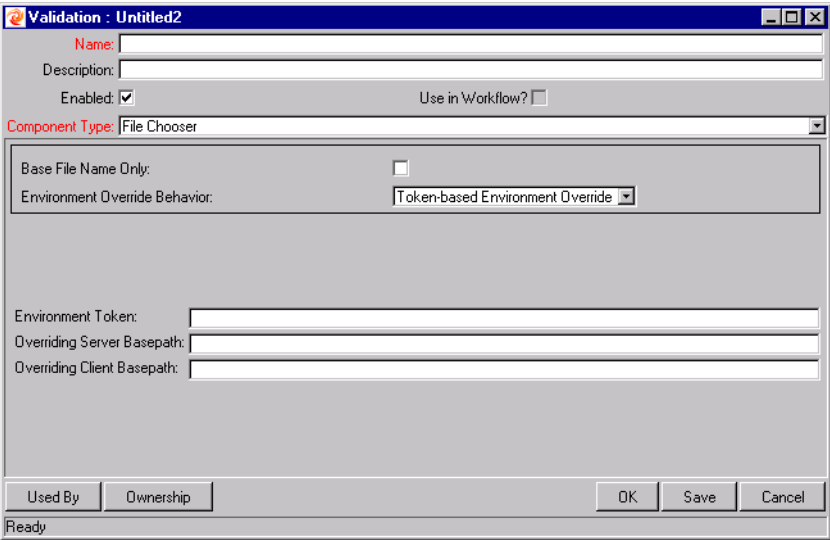


Figure 11-6 Validation Window for Token-Based Environment Override in File Chooser.

Table 11-17. Token-Based Environment Override

Field	Definition
ENVIRONMENT TOKEN	Select the token that will resolve to the overriding Environment.
OVERRIDING SERVER BASEPATH	The server basepath of the Environment that is to be resolved by the token may be overridden.
OVERRIDING CLIENT BASEPATH	The client basepath of the Environment that is to be resolved by the token may be overridden.

Creating 1800 Character Text Areas

Standard Text Areas are between 1 and 200 characters. Kintana does, however, allow you to create a Text Area Validation with a character length of 1800. To create this Validation:

1. Open the Validation Workbench.
2. Search for “**TEXT AREA - 1800.**”
3. In the results tab, select **TEXT AREA - 1800.**
4. Click **COPY.**
5. Rename the Validation.

The new Text Area Validation (with a length of 1800) can be used when defining a custom field in Kintana.

Configuring the Table Component

The table component is used to enter multiple records into a single Kintana field on a Kintana Request. The table component can be configured to include multiple columns of varied data types. Additionally, this component supports rules for populating elements within the table and provides functionality for capturing column totals.

1. Click the Table Component icon to open the Table Component entry page.

2. Add, edit, or delete entries in the list.

Table Component (No Entries)

Table Component

Instructions for using the table component on a Request.

Seq	Column 1	Column 2	Column 3
<input type="checkbox"/> 1	Entry 1	Entry 2	Entry 3

Check All Clear All Add Edit Copy Delete

Done Cancel

Fields of this component can only be added to Request Types, Request Header Types and Request User Data.

To configure and use a Table Component:

- *Define the Table Component in the Validation Workbench*
- *Add the Table Component to a Request Type*



Example

ACME creates a Request Type to request quotes and parts for hardware. Each entry of this type has four elements: PART, SUB-TYPE, PART NUMBER, and UNIT PRICE. ACME creates a Table Component field called HARDWARE INFORMATION to collect this information.

When the user logs a request for new hardware, the Request displays the HARDWARE INFORMATION field. The user opens the field. He selects a PART, which triggers a rule to populate the PART NUMBER and UNIT PRICE. He submits the Request, which now contains all of the information required to successfully order the hardware.

Define the Table Component in the Validation Workbench

To create a Table Component field:

1. Open the VALIDATION WORKBENCH in the CONFIGURATION screen group.
2. Click **NEW VALIDATION**. The VALIDATION window opens.
3. Select **TABLE COMPONENT** from the COMPONENT TYPE drop down list.

Validation : Untitled1

Name: _____

Description: _____

Enabled: Use in Workflow?

Component Type: Table Component

User Instructions: _____

Meta Layer View: MREQ_ _____

Table Columns | Form Layout | Rules

Column Seq.	Column Header	Column Token	Parameter Col.	Enabled	Component
-------------	---------------	--------------	----------------	---------	-----------

↑ ↓ New Edit Remove

Used By Ownership OK Save Cancel

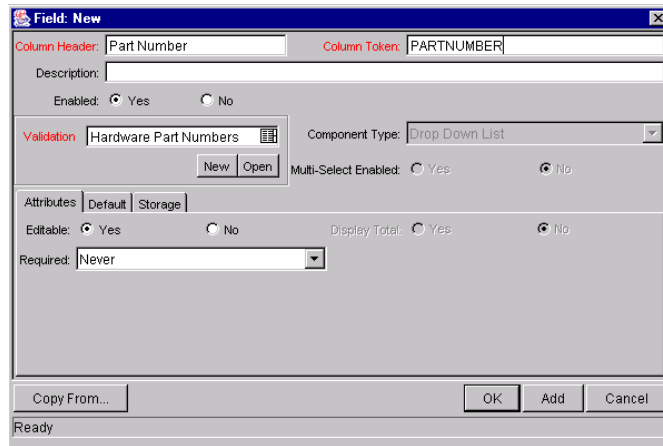
Ready

4. Enter a Validation NAME and DESCRIPTION.
5. Enter any USER INSTRUCTIONS. This text will appear on the top of the table entry page.
6. Create the Table Columns.
 - a. Click **NEW** in the **TABLE COLUMNS** tab. The FIELD window opens.
 - b. Define the type of information that will be stored in that column's entries. This may require you to create a new Validation for the column.

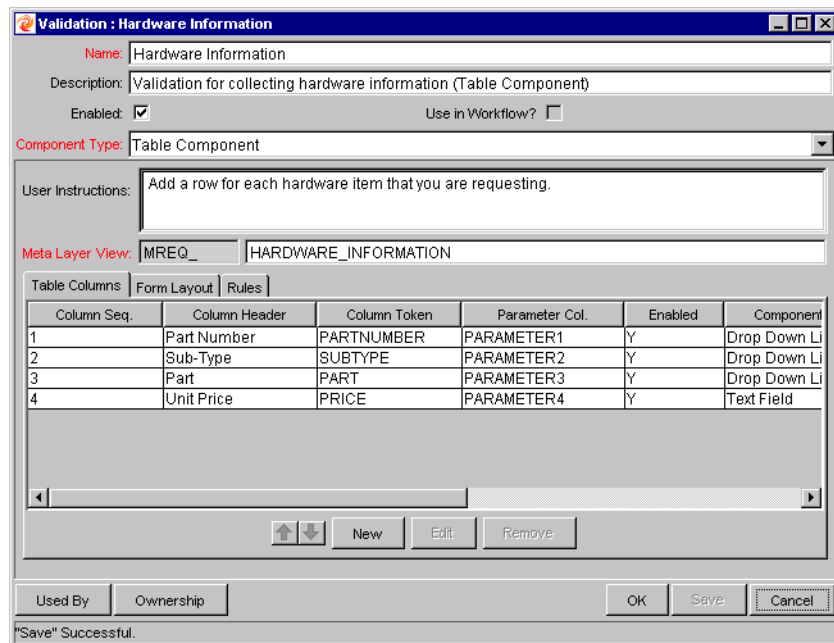


Note

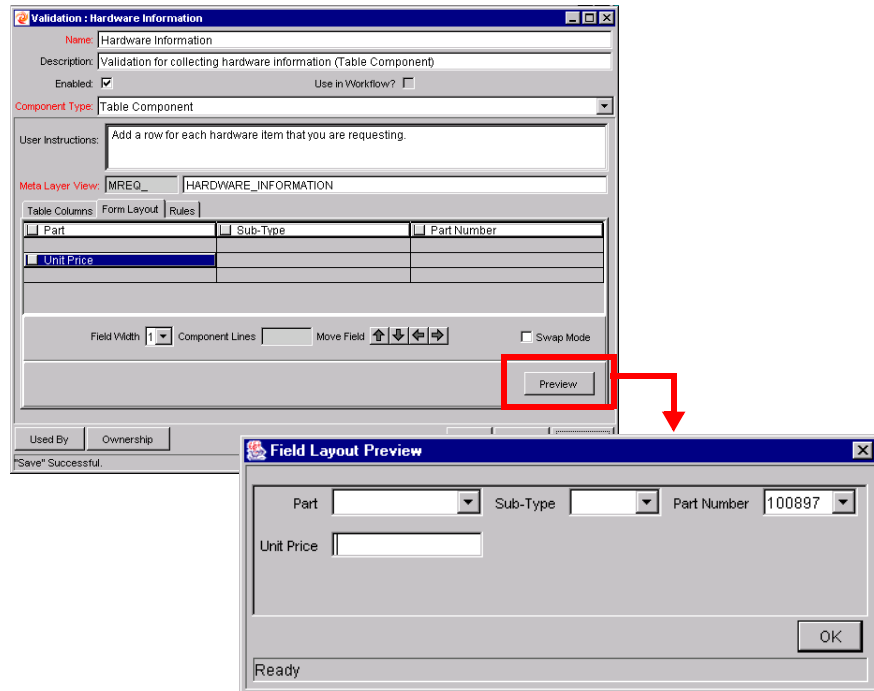
File attachments can not be used in a Table component column.



- c. Specify the **ATTRIBUTES** (EDITABLE OR REQUIRED) and any **DEFAULT** behavior.
- d. Click **ADD** to save the column information and add another column. When you are finished adding columns, click **OK** to close the **FIELD** window.



- 7. Configure the Form Layout.
 - a. Click the **FORM LAYOUT** tab.
 - b. Select the fields and move their positions using the arrow buttons.



- c. Click **Preview** to see a representation of the final positioning. Note that the **Preview** loads a window in the Workbench, but the actual table component will only be available to users in the standard Kintana interface (HTML).
8. Configure any Table logic in the **RULES** tab. Rules are used for advanced defaulting behavior and calculating column totals.
 - a. Click the **RULES** tab.
 - b. Click **NEW** to define a new rule. See [“Creating a Table Rule”](#) on page 320 for detailed instructions.
9. Click **OK** to save the Validation.

The new Table Component field can be included on a Request Type, Request Header Type or Request User Data field.

Creating a Table Rule

Table rules are configured in the same manner as advanced Request Type rules. Essentially, you can configure fields (columns) in the table to default to certain values based on an event or value in another field in the table. Because

the table component rules are configured using a SQL statement, you are given enormous flexibility for the data that is populated in the table cells.

Table rules are configured using the **RULES** tab on the **VALIDATION** window.

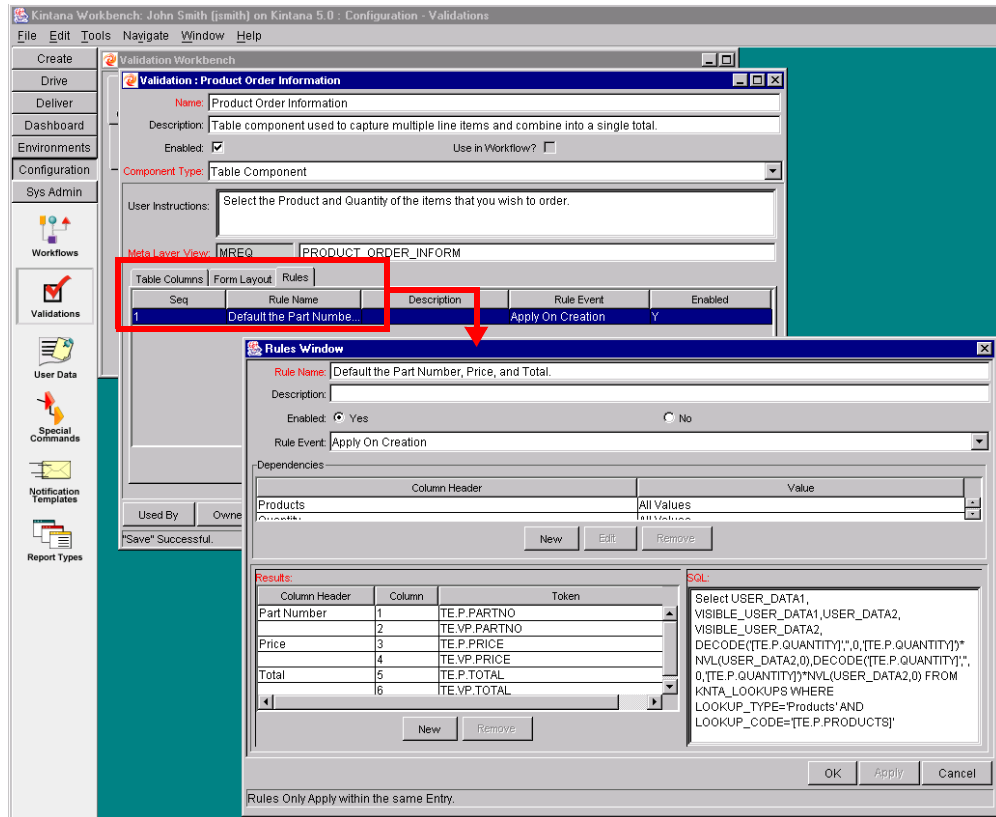


Figure 11-7 Rules window accessed from the Rules tab

Example: Using a Table Component on an Order Form

The following example illustrates the table component rules functionality.

ACME, Inc. uses a Request for creating and tracking employee computer hardware equipment orders. ACME has included a table component field on their Request Type for gathering the order information. When the employee selects a Product, the Unit Price is automatically updated. Then, when they update the Quantity, the total line cost is automatically calculated and displayed in the table.

To enable this functionality, ACME first has to configure a new Validation with the following specifications:

Table 11-18. Example - Table Component Validation Settings

Setting	Value / Description
Validation Name	Product Order Information
Component Type	Table Component
Column 1	Column Header = Products Column Token = PRODUCTS Validation = Auto complete list with the following list values: PC, MOUSE, MONITOR, KEYBOARD
Column 2	Column Header = Quantity Column Token = QUANTITY Validation = Numeric Text Field
Column 3	Column Header = Price Column Token = PRICE Validation = Numeric Text Field
Column 4	Column Header = Total Column Token = TOTAL Validation = Numeric Text Field

Validation : Product Order Information

Name:

Description:

Enabled: Use in Workflow?

Component Type:

User Instructions:

Meta Layer View:

Table Columns | Form Layout | Rules

Column Seq.	Column Header	Column Token	Parameter Col.	Enabled	Component Type	Validation
1	Products	PRODUCTS	PARAMETER5	Y	Auto Complete List	Product List for Order Form
2	Quantity	QUANTITY	PARAMETER3	Y	Text Field	Numeric Text Field
3	Price	PRICE	PARAMETER1	Y	Text Field	Numeric Text Field
4	Total	TOTAL	PARAMETER4	Y	Text Field	Numeric Text Field

Used By: Ownership:

OK Save Cancel

Ready

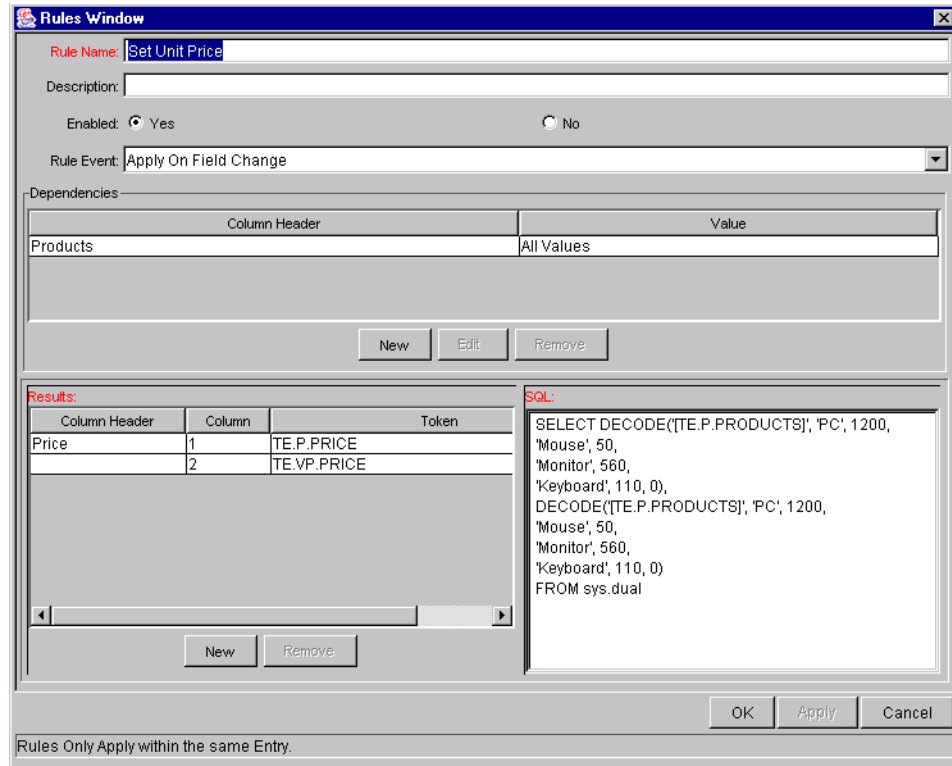
Once the Validation's columns have been defined, the Rules can be configured:

Rule 1: Set Unit Price.

ACME uses the following rule to set the default unit price in the PRICE cell based on the PRODUCT selection.

Table 11-19. Example - Set Unit Price Rule Settings

Setting	Value / Description
Rule Name	Set Unit Price
Rule Event	Apply on Field Change
Dependencies	Column = Products All Values = Yes
Results	Column Header = Price
SQL	<pre>SELECT DECODE(['TE.P.PRODUCTS'], 'PC', 1200, 'Mouse', 50, 'Monitor', 560, 'Keyboard', 110, 0), DECODE(['TE.P.PRODUCTS'], 'PC', 1200, 'Mouse', 50, 'Monitor', 560, 'Keyboard', 110, 0) FROM sys.dual</pre>



Rule 2: Set Unit Price.

ACME uses the following rule to set the calculate and display the total line price in the TOTAL column based on the values in the PRODUCTS and QUANTITY cells.

Table 11-20. Example - Calculate Total Rule Settings

Setting	Value / Description
Rule Name	Set Unit Price
Rule Event	Apply on Field Change
Dependencies	Column = Price [All Values = Yes] Column = Quantity [All Values = Yes]
Results	Column Header = Total
SQL	SELECT [TE.P.PRICE] * [TE.P.QUANTITY], [TE.P.PRICE] * [TE.P.QUANTITY] from sys.dual

Using the Table Component

Add a field to a Request Type that is validated by this Table Component Validation. When a user opens the field to enter information, the table rules will be applied to each row that is created.

Tokens in the Table Components

Each column included in the table component has an associated Token. These Tokens can be used in the same manner as other field tokens in Kintana, such as for commands, notifications or advanced field defaulting. See ["Using Commands and Tokens"](#) for details on referencing Tokens related to Table Components.

Calculating Column Totals

You can configure columns that are validated by a number to calculate the total for that column. This is configured in the Validation's FIELD window. The following example illustrates how to configure a column to calculate and display the column total.

ACME, Inc. uses a Request for creating and tracking simple employee equipment orders. ACME has included a table component field on their Request Type for gathering the order information. Employee enter the Purchase Items and Cost for each item. The table component automatically calculates the total cost for the Cost column.

ACME creates a Validation with the following settings:

- COMPONENT TYPE = **TABLE COMPONENT**.
- Column 1 = Purchase Item (text field)
- Column 2 = Cost (number). In the FIELD window for the COST column, select DISPLAY TOTAL = **Yes**. The DISPLAY TOTAL field is only enabled if the field's validation is a number.

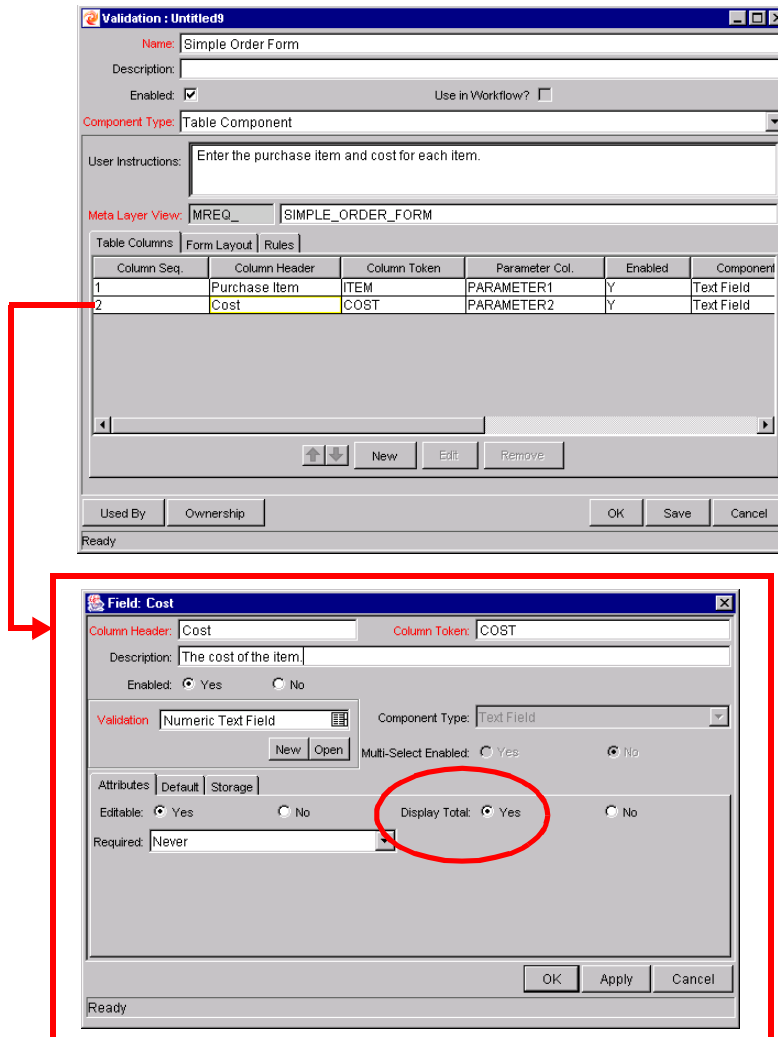


Figure 11-8 Sample validation for a Simple Order table component.

ACME includes adds a field to their Order Request Type that uses this Validation. When a user creates a Request using that Request Type, he can

click on the table component icon next to the field to open the order form. The total for the COST column is displayed at the bottom of the table.

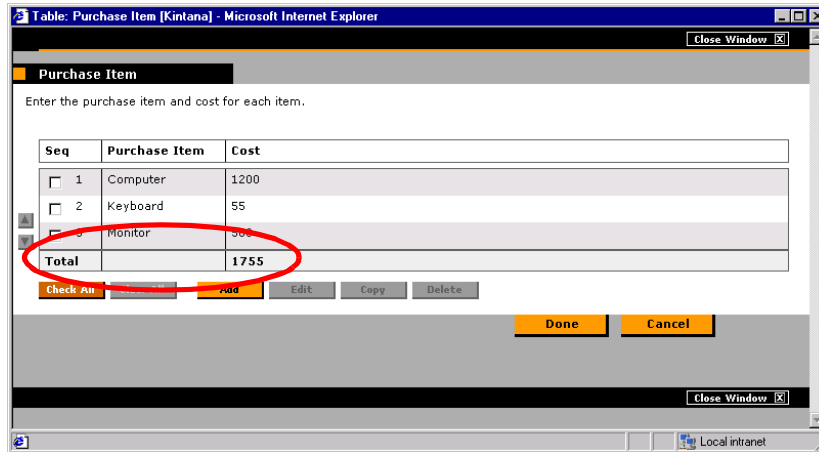


Figure 11-9 Sample table component displaying a column total.

Add the Table Component to a Request Type

Table Component fields can be included on a Request Type, Request Header Type or Request User Data field.

To add a Table Component field to a Request Type:

1. Open the REQUEST TYPE window.
2. Click NEW in the **FIELDS** tab. The FIELD window opens.
3. Enter the FIELD PROMPT, TOKEN, and DESCRIPTION.
4. In the Validation field, select a table component Validation. If you have not created a table component Validation, click **NEW** to create one. See *“Define the Table Component in the Validation Workbench”* on page 317 for instructions.

Configuring a Deployment System

Field: New

Field Prompt: Hardware Information Table Token: HARDWAREINFO

Description: Table Component for entering hardware information.

Enabled: Yes No

Validation: Hardware Information Component Type: Table Component

New Open

Multiselect: Yes No

Attributes Default Storage Security

Section Name: Request Type Fields Display Only: Yes No

Transaction History: Yes No Notes History: Yes No

Display on Search and Filter: Yes No Display: Yes No

Copy From... OK Add Cancel

Ready

5. Click **OK** to add the field to the Request Type.

6. Save the Request Type.

The table component field will now appear on Requests of this Request Type.

KINTANA

Create A Request > Create New Hardware Request Type

Welcome John Smith

Create New Hardware Request Type

Expand All Collapse All

Submit Cancel

Header

Tab #1

Created By: jsmith

Department: Manufacturing Sub-Type:

Workflow:

Priority: High Application:

Assigned To: Assigned Group:

Request Group:

Contact Name:

Contact Phone:

Contact Email:

Description: Requesting new hardware for John Smith.

Request Type Fields

Hardware Information Table 2 Entries

Kintana [Kintana] - Microsoft Internet Explorer

Hardware Information Table

Add a row for each hardware item that you are requesting.

Seq	Part Number	Sub-Type	Part	Unit Price
<input type="checkbox"/> 1	100897	18 inch	Monitor	\$320.00
<input type="checkbox"/> 2	899768		Keyboard	\$60.00

Check All Clear All Add Edit Copy Delete

Done Cancel

Package and Request Group Validations

Two particular entity-specific Validations can be accessed in the Kintana Workbench without entering the `VALIDATIONS` screen group:

- *Package and Request Groups*
- *Request Type Category*

Package and Request Groups

The KNTA-Package and Request Groups Validation can be accessed directly from the **PACKAGE** screen. To specify that a Package belongs to a new or unique Package Group that is not named in the auto-complete Validation list, it is not necessary to proceed through the `VALIDATION WORKBENCH`.

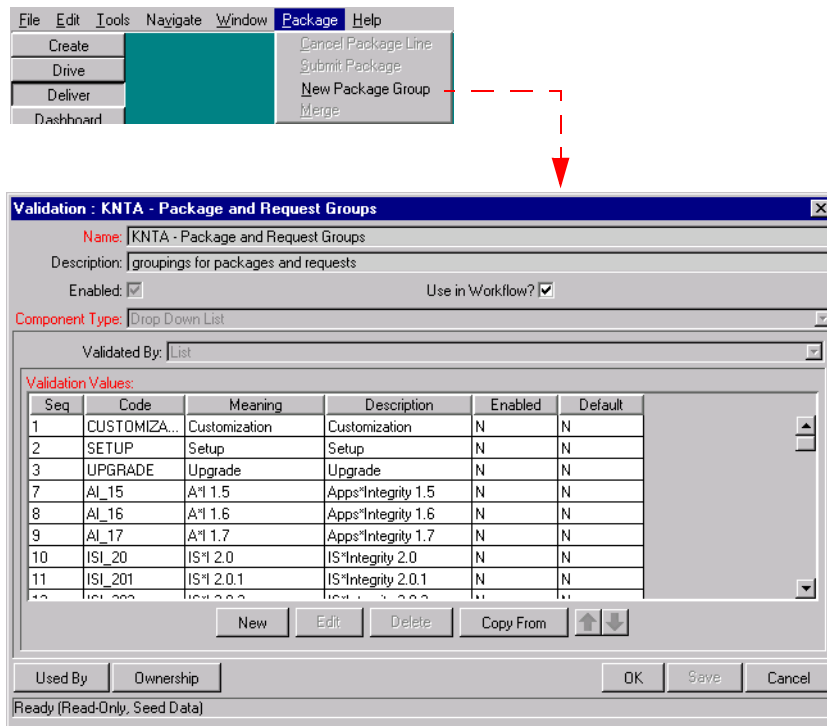
To access the KNTA-Package and Request Groups Validation window from the **PACKAGE** screen:

Select **NEW PACKAGE GROUP** from the **PACKAGE** menu. The `VALIDATION` window will appear, listing the existing Kintana Deliver Package Groups.



Note

All users are granted read access to this screen, but only users with appropriate security privileges can alter the KNTA-Package and Request Groups Validation list.



Request Type Category

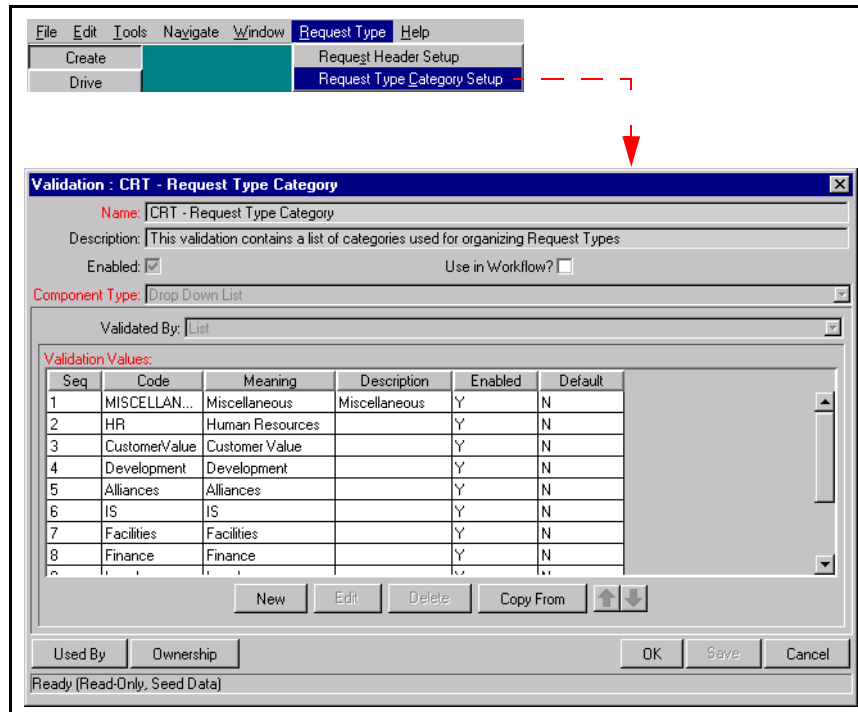
The CRT - REQUEST TYPE CATEGORY Validation can be accessed directly from the **REQUEST TYPES** screen.

To access the CRT - REQUEST TYPE CATEGORY Validation window from the **REQUEST TYPES** screen:

Select **REQUEST TYPE CATEGORY SETUP** from the **REQUEST TYPE** menu. The **VALIDATION** window will appear, listing the existing Kintana Request Type Categories.



All users are granted read access to this screen, but only users with appropriate security privileges can alter the CRT - Request Type Category Validation list.



Validation Special Characters

The VALIDATION NAME field for all Validations cannot contain a question mark ('?'). The Kintana Workbench prevents this character from being entered into the field, but all previously configured Validation Names (Validations entered before Kintana release 4.5) should be checked and corrected.

System Validations

The following is a list of the default validations that are installed with Kintana. Note that many of these validations may have been altered to better match your company's specific business needs. The table contains the following data:

- Validation Name
- Component Type: the type of field that the Validation represents
- Use in Workflows: whether the Validation is currently enabled for use in Workflows.



Note

Use the Validations report to get a list of all validations currently in your system. This includes information such as the validations' values and any SQL used to generate the values.

Table 11-21. Kintana System Validations

Validation Name	Component Type	Use in Workflow?
All Projects - Cost Enabled and Not New	Auto Complete List	N
Application Module	Drop Down List	N
Attachment	Attachment	N
CONNECTION_PROTOCOL	Drop Down List	N
CRT - All DSS Report Types	Auto Complete List	N
CRT - All Regular Report Types	Auto Complete List	N
CRT - Application - Enabled	Auto Complete List	N
CRT - Assigned Group - All	Auto Complete List	N
CRT - Assigned Group - Enabled	Auto Complete List	N
CRT - Assigned Group - Participant	Auto Complete List	N
CRT - Assigned To - Enabled	Auto Complete List	N
CRT - Assigned To - Participant	Auto Complete List	N
CRT - Company	Auto Complete List	N
CRT - Company - All	Auto Complete List	N
CRT - Contact Email - Enabled	Auto Complete List	N
CRT - Contact Email - Restricted Enabled	Auto Complete List	N
CRT - Contact Email by Company - Enabled	Auto Complete List	N
CRT - Contact Name - All	Auto Complete List	N
CRT - Contact Name - Enabled	Auto Complete List	N
CRT - Contact Name - Restricted	Auto Complete List	N
CRT - Contact Name - Restricted Enabled	Auto Complete List	N
CRT - Contact Name by Company - Enabled	Auto Complete List	N
CRT - Contact Phone - Enabled	Auto Complete List	N
CRT - Contact Phone - Restricted Enabled	Auto Complete List	N
CRT - Contact Phone by Company - Enabled	Auto Complete List	N
CRT - Contact Synch Driver	Drop Down List	N
CRT - DSS Report Types	Auto Complete List	N
CRT - Department - Enabled	Drop Down List	N
CRT - Difficulty	Drop Down List	N
CRT - Dynamic Reporting Column List	Auto Complete List	N
CRT - Enhancement Request Category	Drop Down List	N
CRT - Impact	Drop Down List	N
CRT - Max Custom Fields	Drop Down List	N
CRT - Modification Type	Drop Down List	N

Table 11-21. Kintana System Validations

Validation Name	Component Type	Use in Workflow?
CRT - Platform	Drop Down List	N
CRT - Priority - All	Drop Down List	N
CRT - Priority - Enabled	Drop Down List	Y
CRT - Priority - No Default	Drop Down List	Y
CRT - Priority AutoComplete	Auto Complete List	N
CRT - Prj Code - Enabled	Auto Complete List	N
CRT - Regular Report Types	Auto Complete List	N
CRT - Request Activity Portlet Group By	Drop Down List	N
CRT - Request Analyzed	Drop Down List	Y
CRT - Request Assigned	Drop Down List	Y
CRT - Request Detail Report Order By	Drop Down List	N
CRT - Request Group - All	Auto Complete List	N
CRT - Request Group - Enabled	Auto Complete List	N
CRT - Request Header Type Name (Migrator Source)	Auto Complete List	N
CRT - Request Header Types - All	Auto Complete List	N
CRT - Request Header Types - Enabled	Auto Complete List	N
CRT - Request Held	Drop Down List	Y
CRT - Request In Progress	Drop Down List	Y
CRT - Request Info Required	Drop Down List	Y
CRT - Request List	Auto Complete List	N
CRT - Request List Narrow Sort By	Drop Down List	N
CRT - Request List Wide Sort By	Drop Down List	N
CRT - Request List Wide Status	Auto Complete List	N
CRT - Request Listing Report Columns	Auto Complete List	N
CRT - Request Quick View Order By	Drop Down List	N
CRT - Request Reference Type	Drop Down List	N
CRT - Request Reviewed	Drop Down List	Y
CRT - Request Summary Group By	Auto Complete List	N
CRT - Request Summary Group By Types	Drop Down List	N
CRT - Request Type Category	Drop Down List	N
CRT - Request Type Fields	Auto Complete List	N
CRT - Request Type Fields - All	Auto Complete List	N
CRT - Request Type Name (Migrator Source)	Auto Complete List	N
CRT - Request Type Names - All	Auto Complete List	N
CRT - Request Type Names - Restricted by Package Workflow	Drop Down List	N
CRT - Request Type Notification Fields	Auto Complete List	N
CRT - Request Type Prompt - All	Auto Complete List	N
CRT - Request Type Restriction	Drop Down List	N
CRT - Request Type Status (Partial)	Auto Complete List	N
CRT - Request Type Status (Partial) REQ tokens	Auto Complete List	N
CRT - Request Type Status - All	Auto Complete List	N

Table 11-21. Kintana System Validations

Validation Name	Component Type	Use in Workflow?
CRT - Request Types - All	Auto Complete List	N
CRT - Request Types - All.	Auto Complete List	N
CRT - Request Types - Enabled	Auto Complete List	N
CRT - Request Types - Enabled.	Auto Complete List	N
CRT - Request Types - Interface Restricted	Auto Complete List	N
CRT - Request Types - Restricted	Auto Complete List	N
CRT - Request Types - Restricted Enabled	Auto Complete List	N
CRT - Request Types w. Description - Interface Restricted	Auto Complete List	N
CRT - Requests - All	Auto Complete List	N
CRT - Resolution	Drop Down List	Y
CRT - Rule Dependencies Fields	Auto Complete List	N
CRT - Rule Results Fields	Auto Complete List	N
CRT - Security Group With Description	Auto Complete List	N
CRT - Sort By	Drop Down List	N
CRT - Sub Types - All	Auto Complete List	N
CRT - SubTypes - All	Auto Complete List	N
CRT - SubTypes - Enabled	Auto Complete List	N
CRT - Validations - Enabled	Auto Complete List	N
CRT - Workflow Id - All	Auto Complete List	N
CRT - Workflow With Description	Auto Complete List	N
CRT - Workflows - Enabled	Auto Complete List	N
CRT - Workflows - Restricted	Auto Complete List	N
CRT Request - Queryable Fields	Auto Complete List	N
CST - All Budget Status	Drop Down List	N
CST - Budget Entities	Auto Complete List	N
CST - Budget Fiscal Periods	Auto Complete List	N
CST - Budget For Types	Drop Down List	N
CST - Budget Line Type	Drop Down List	N
CST - Budget Programs	Auto Complete List	N
CST - Budget Projects	Auto Complete List	N
CST - Budget Rolls Up To Types	Drop Down List	N
CST - Budget Search Sort By	Drop Down List	N
CST - Budgets Line Category	Auto Complete List	N
CST - Fiscal Quarters	Drop Down List	N
CST - Parent Org Unit Budgets	Auto Complete List	N
CST - Parent Program Budgets	Auto Complete List	N
CST - Parent Project Budgets	Auto Complete List	N
CST - Program Names	Auto Complete List	N
CST - Specific Entity Linked Budgets	Auto Complete List	N
Component Type	Drop Down List	N
DEM - All Assignable Users	Auto Complete List	N

Table 11-21. Kintana System Validations

Validation Name	Component Type	Use in Workflow?
DEM - Assignment Queue Order By	Auto Complete List	N
DEM - Demand Categories with Disposition	Auto Complete List	N
DEM - Demand Disposition	Auto Complete List	N
DEM - Demand Disposition Open and Satisfied	Auto Complete List	N
DEM - Demand Disposition Outstanding	Auto Complete List	N
DEM - Demand Fields	Auto Complete List	N
DEM - Demand Fields only from Demand Set view name	Auto Complete List	N
DEM - Demand Fields with Disposition	Auto Complete List	N
DEM - Demand Fields with Disposition & Request Type	Auto Complete List	N
DEM - Demand Set Views	Auto Complete List	N
DEM - Demand Sets - Enabled	Drop Down List	N
DEM - Demand Sets Request Types	Auto Complete List	N
DEM - Request Held	Drop Down List	N
DEM - Request Type Statuses	Auto Complete List	N
DEM - Request Types of a Demand Set	Auto Complete List	N
DEM - SLA Level	Auto Complete List	N
DEM - Search Validations - All	Auto Complete List	N
DEM Filter - User Id - with empty - when no category	Auto Complete List	N
DIST - Workflow Id - Enabled	Auto Complete List	N
DLV - Accelerator - Enabled	Drop Down List	N
DLV - Accelerator Panel, Env Screen - Enabled	Auto Complete List	N
DLV - All DSS Report Types	Auto Complete List	N
DLV - All Regular Report Types	Auto Complete List	N
DLV - Assigned Group - Enabled	Auto Complete List	N
DLV - Assigned Group - Participant	Auto Complete List	N
DLV - Assigned Group - Restricted	Auto Complete List	N
DLV - Assigned To - Participant	Auto Complete List	N
DLV - Assigned To - Restricted	Auto Complete List	N
DLV - DSS Report Types	Auto Complete List	N
DLV - Database Type	Drop Down List	N
DLV - Execution Order	Drop Down List	N
DLV - File Location	Drop Down List	N
DLV - File Type	Drop Down List	N
DLV - Files of Type	Drop Down List	N
DLV - Kintana Server Directory Chooser	Directory Chooser	N
DLV - Kintana Server File Chooser	File Chooser	N
DLV - ODF Mode	Drop Down List	N
DLV - Object Category - Enabled	Drop Down List	N
DLV - Object Category -All	Drop Down List	N
DLV - Object History Order By	Drop Down List	N
DLV - Object Name - All	Auto Complete List	N

Table 11-21. Kintana System Validations

Validation Name	Component Type	Use in Workflow?
DLV - Object Type - All	Auto Complete List	N
DLV - Object Type - Enabled	Auto Complete List	Y
DLV - Object Type ID - Enabled	Auto Complete List	N
DLV - Object Type ID - Restricted	Auto Complete List	N
DLV - Object Type Id - All	Auto Complete List	N
DLV - Object Type Name (Migrator Source)	Auto Complete List	N
DLV - Package Activity Portlet Group By	Drop Down List	N
DLV - Package Group - All	Auto Complete List	N
DLV - Package Group - Enabled	Auto Complete List	N
DLV - Package ID	Auto Complete List	N
DLV - Package List	Auto Complete List	N
DLV - Package List Portlet Narrow Sort By	Drop Down List	N
DLV - Package List Portlet Wide Sort By	Drop Down List	N
DLV - Package Number	Auto Complete List	N
DLV - Package Pending Filter	Drop Down List	N
DLV - Package Pending Order By	Drop Down List	N
DLV - Package Priority - All	Drop Down List	N
DLV - Package Priority - Enabled	Drop Down List	Y
DLV - Package Status - All	Auto Complete List	N
DLV - Package Type - All	Drop Down List	N
DLV - Package Type - Enabled	Drop Down List	N
DLV - Patch Env Order By	Drop Down List	N
DLV - Patch Object Type Names	Drop Down List	N
DLV - Priority AutoComplete	Auto Complete List	N
DLV - Regular Report Types	Auto Complete List	N
DLV - Release Number	Auto Complete List	N
DLV - SQL Paths	Drop Down List	N
DLV - Sort By	Drop Down List	N
DLV - Workflow Id - All	Auto Complete List	N
DLV - Workflow Id - Enabled	Auto Complete List	N
DLV - Workflow Id - Restricted	Auto Complete List	N
DLV - Workflow Id - Restricted Enabled	Auto Complete List	N
DLV Package - Queryable Fields	Auto Complete List	N
DRV - Activity Priority - Enabled	Drop Down List	N
DRV - Activity Status - Enabled	Drop Down List	N
DRV - Add User To Task	Auto Complete List	N
DRV - All DSS Report Types	Auto Complete List	N
DRV - All Project List	Auto Complete List	N
DRV - All Regular Report Types	Auto Complete List	N
DRV - Baseline for Comparision	Auto Complete List	N
DRV - Booked Skill	Auto Complete List	N

Table 11-21. Kintana System Validations

Validation Name	Component Type	Use in Workflow?
DRV - Calendar Reason	Drop Down List	N
DRV - Confidence	Drop Down List	N
DRV - Custom Fields base on Project Template	Auto Complete List	N
DRV - DSS Report Types	Auto Complete List	N
DRV - Exception Rule Types	Auto Complete List	N
DRV - Filter Reference Type	Drop Down List	N
DRV - Filter Relationship	Drop Down List	N
DRV - Interface Resources Tables	Auto Complete List	N
DRV - Master Projects - Enabled	Auto Complete List	N
DRV - Master Projects - Enabled and not New	Auto Complete List	N
DRV - Master Projects with Baselines - Enabled	Auto Complete List	N
DRV - My Tasks Sort By	Auto Complete List	N
DRV - Proj Manager - Restricted	Auto Complete List	N
DRV - Project Detail Report Order By	Auto Complete List	N
DRV - Project Fields	Auto Complete List	N
DRV - Project Grouping Type	Drop Down List	N
DRV - Project Header Fields	Auto Complete List	N
DRV - Project List Portlet Narrow Sort By	Drop Down List	N
DRV - Project List Portlet Wide Sort By	Drop Down List	N
DRV - Project Names - All	Auto Complete List	N
DRV - Project Names - All - Depend on [P_SHOW_MASTER_ONLY]	Auto Complete List	N
DRV - Project Names - In Templates	Auto Complete List	N
DRV - Project Names by Template	Auto Complete List	N
DRV - Project State Search	Auto Complete List	N
DRV - Project States	Auto Complete List	N
DRV - Project Status - All	Auto Complete List	N
DRV - Project Task Aging Report Order By	Auto Complete List	N
DRV - Project Team Resource	Auto Complete List	N
DRV - Project Template Names	Auto Complete List	N
DRV - Project Template Names - Enabled	Auto Complete List	N
DRV - Project Template with TemplateID and ParameterSetContextID	Auto Complete List	N
DRV - Project Type Names - All	Auto Complete List	N
DRV - Project Type Prompt - All	Auto Complete List	N
DRV - Project Type Status - All	Auto Complete List	N
DRV - Project Types - All	Auto Complete List	N
DRV - Project Types - Enabled	Auto Complete List	N
DRV - Project User Data Roll-Up Fields	Auto Complete List	N
DRV - Projects (Only) - Enabled	Auto Complete List	N
DRV - Projects - All	Auto Complete List	N
DRV - Projects - Enabled	Auto Complete List	N

Table 11-21. Kintana System Validations

Validation Name	Component Type	Use in Workflow?
DRV - Regular Report Types	Auto Complete List	N
DRV - Request Priority - All	Auto Complete List	N
DRV - Request Reference Relationships	Drop Down List	N
DRV - Resource Group - Restricted Enabled	Auto Complete List	N
DRV - Resource Names - Proj Teams	Auto Complete List	N
DRV - Resource or Project Manager	Auto Complete List	N
DRV - Scheduling Constraints	Drop Down List	N
DRV - Security Restrictions	Drop Down List	N
DRV - Sort By for Projects	Drop Down List	N
DRV - Sort By for Tasks	Drop Down List	N
DRV - Sub Project Ids - Enabled	Auto Complete List	N
DRV - Sub Projects - Enabled	Auto Complete List	N
DRV - Subproject Name for My Tasks	Auto Complete List	N
DRV - Summary Condition	Auto Complete List	N
DRV - Task Categories	Drop Down List	N
DRV - Task Categories - AutoComp	Auto Complete List	N
DRV - Task Exception Type Names	Auto Complete List	N
DRV - Task Ids - Enabled	Auto Complete List	N
DRV - Task Name - Enabled	Auto Complete List	N
DRV - Task Name - In Templates	Auto Complete List	N
DRV - Task Notification Dates	Auto Complete List	N
DRV - Task State Search	Auto Complete List	N
DRV - Task States	Auto Complete List	N
DRV - Task States for My Tasks	Auto Complete List	N
DRV - Task States for Notifications	Drop Down List	N
DRV - Task User Data Roll-Up Fields	Auto Complete List	N
DRV - Task and Project #s - Enabled	Auto Complete List	N
DRV - Task and Project Names - Enabled	Auto Complete List	N
DRV - Tasks - All	Auto Complete List	N
DRV - Unlinked Packages for Drive	Auto Complete List	N
DRV - Unlinked Project References for Drive	Auto Complete List	N
DRV - Unlinked Requests for Drive	Auto Complete List	N
DRV - Workflow - All	Auto Complete List	N
DRV - Workflow - Enabled	Auto Complete List	N
DSH - Column Type	Drop Down List	N
DSH - Hyperlink Type	Drop Down List	N
DSH - Portlet Category	Drop Down List	N
DSH - Portlet Category with ALL	Drop Down List	N
DSH - Portlet Names - All	Auto Complete List	N
DSH - Portlet Types in Category	Auto Complete List	N
DSH - Portlet Width	Drop Down List	N

Table 11-21. Kintana System Validations

Validation Name	Component Type	Use in Workflow?
DSH - Project References Portlet Reference Type	Auto Complete List	N
DSH - Project References Portlet Sort By	Drop Down List	N
DSH - Time-Out	Drop Down List	N
DSH - Validations - Enabled	Auto Complete List	N
DSS - Chart Type	Auto Complete List	N
DSS - Migration Status	Drop Down List	N
DSS - Object Migration Groupings	Drop Down List	N
DSS - Package Line Grouping Types	Drop Down List	N
DSS - Period End Dates	Auto Complete List	N
DSS - Period Start Dates	Auto Complete List	N
DSS - Period Types	Drop Down List	N
DSS - Request Grouping Types	Drop Down List	N
DSS - Workflows - All	Auto Complete List	N
Dashboard - All Users - Fullname	Auto Complete List	N
Data Mask	Drop Down List	Y
Date	Date Field	Y
Date (Short Format)	Date Field	N
Date Format	Drop Down List	Y
Debug Level	Drop Down List	N
Default Type	Drop Down List	N
Directory Chooser	Directory Chooser	N
ENV - All DB Environments	Auto Complete List	N
ENV - App Code - Enabled	Drop Down List	N
ENV - App Code - Restricted	Drop Down List	N
ENV - Compared App Codes	Auto Complete List	N
ENV - Comparison Types	Drop Down List	N
ENV - Custom Objects	Auto Complete List	N
ENV - Env Group Id - Access Specific	Auto Complete List	N
ENV - Env Group Id - Enabled	Auto Complete List	N
ENV - Environment Id - Access Specific	Auto Complete List	N
ENV - Environment Id - All	Auto Complete List	N
ENV - Environment Id - Enabled	Auto Complete List	N
ENV - Environment Id - OM installed	Auto Complete List	N
ENV - Environment Name - All	Auto Complete List	N
ENV - Environment Server/Client Type	Drop Down List	N
ENV - Filesystem Environments	Auto Complete List	N
ENV - Filesystem Exclusion Choices	Drop Down List	N
ENV - MS SQLServer Environments	Auto Complete List	N
ENV - Oracle Environments	Auto Complete List	N
ENV - Reference App Codes	Auto Complete List	N
ENV - Tier	Drop Down List	N

Table 11-21. Kintana System Validations

Validation Name	Component Type	Use in Workflow?
File Chooser - Base File Name	File Chooser	N
File Chooser - Full File Name	File Chooser	N
Gantt Overview Portlet Display and Tooltip label fields	Auto Complete List	N
Gantt Portlet Display and Tooltip label fields	Auto Complete List	N
Gantt Portlet Sort By	Drop Down List	N
KCST - Budget Names	Budget	N
KCST - Budget Names - All	Auto Complete List	N
KCST - Existing Org Unit Budgets	Auto Complete List	N
KCST - Existing Program Budgets	Auto Complete List	N
KCST - Existing Projects Budgets	Auto Complete List	N
KCST - New Org Unit Budgets	Auto Complete List	N
KCST - New Program Budgets	Auto Complete List	N
KCST - New Projects Budgets	Auto Complete List	N
KDRV - Project Predecessor Types	Drop Down List	N
KNTA - Authentication Mode All	Drop Down List	N
KNTA - Access Grant - All	Auto Complete List	N
KNTA - Application - Enabled	Auto Complete List	N
KNTA - Applications - All	Auto Complete List	N
KNTA - Autocomp Validation Type	Drop Down List	N
KNTA - Budgets	Auto Complete List	N
KNTA - Budgets - All	Auto Complete List	N
KNTA - Department - All	Drop Down List	N
KNTA - Department - Enabled	Drop Down List	N
KNTA - Departments - AutoComp	Auto Complete List	N
KNTA - Dependency Rule Usage	Drop Down List	N
KNTA - Dept - All	Auto Complete List	N
KNTA - Dropdown by List	Auto Complete List	N
KNTA - Enabled Combo	Drop Down List	N
KNTA - Entities for Notification History Report	Drop Down List	N
KNTA - FLS Denormalization Entity State	Drop Down List	N
KNTA - FLS Denormalization User Data Type	Drop Down List	N
KNTA - Field Entities	Auto Complete List	N
KNTA - Field Prompts	Auto Complete List	N
KNTA - Field Security Standard Tokens	Auto Complete List	N
KNTA - Field Security Types	Drop Down List	N
KNTA - Field Tokens	Auto Complete List	N
KNTA - Field Validations	Auto Complete List	N
KNTA - Finish Periods of type Fiscal Month	Auto Complete List	N
KNTA - Kintana Migrator Action	Drop Down List	N
KNTA - Kintana Server Names	Auto Complete List	N
KNTA - Kintana objects from migration source instance	Auto Complete List	N

Table 11-21. Kintana System Validations

Validation Name	Component Type	Use in Workflow?
KNTA - LDAP Import Authentication Modes	Drop Down List	N
KNTA - List Validations - All	Auto Complete List	N
KNTA - Long Date Formats	Drop Down List	N
KNTA - Lookup Types - All	Auto Complete List	N
KNTA - Medium Date Formats	Drop Down List	N
KNTA - Meta Layer Action	Drop Down List	N
KNTA - Meta Layer Scope	Drop Down List	N
KNTA - Meta Layer Views Autocomp	Auto Complete List	N
KNTA - Meta Layer view definition template file chooser	Auto Complete List	N
KNTA - Migrator Internationalization Modes	Drop Down List	N
KNTA - Migrator Source Types	Drop Down List	N
KNTA - Notification Formats	Drop Down List	N
KNTA - Notification Recipient Type Choices	Drop Down List	N
KNTA - Notification Recipient Type Code	Drop Down List	N
KNTA - Notification Types	Drop Down List	N
KNTA - Org Unit Linked Security Groups	Auto Complete List	N
KNTA - Org Unit Linked Security Groups - Editable	Auto Complete List	N
KNTA - Organization Unit Names - Enabled	Auto Complete List	N
KNTA - Package and Request Groups	Drop Down List	N
KNTA - Period Types	Drop Down List	N
KNTA - Period Types - All	Drop Down List	N
KNTA - Periods of type Fiscal Month	Auto Complete List	N
KNTA - Product Scope	Drop Down List	N
KNTA - Products	Auto Complete List	N
KNTA - Query Condition	Drop Down List	N
KNTA - Refresh Group Status	Drop Down List	N
KNTA - Relationships for Projects/Tasks	Auto Complete List	N
KNTA - Report Recurrence Pattern	Drop Down List	N
KNTA - Report Submission Status	Drop Down List	N
KNTA - Report Type - All	Auto Complete List	N
KNTA - Report View Access	Drop Down List	N
KNTA - Rule Events	Auto Complete List	N
KNTA - Security Group Id - Access Specific	Auto Complete List	N
KNTA - Security Group Id - All	Auto Complete List	N
KNTA - Security Group Id - Editable	Auto Complete List	N
KNTA - Security Group Id - Enabled	Auto Complete List	N
KNTA - Security Group Name - All	Auto Complete List	N
KNTA - Short Date Formats	Drop Down List	N
KNTA - Shortcut Bar Location	Drop Down List	N
KNTA - Solutions - All	Auto Complete List	N
KNTA - Special Command Names - All	Auto Complete List	N

Table 11-21. Kintana System Validations

Validation Name	Component Type	Use in Workflow?
KNTA - Special Command Type Name (Migrator Source)	Auto Complete List	N
KNTA - Special Commands - Enabled	Auto Complete List	N
KNTA - Specific Entity	Auto Complete List	N
KNTA - Table Comp Rule Dependencies Fields	Auto Complete List	N
KNTA - Table Comp Rule Results Fields	Auto Complete List	N
KNTA - True or False	Drop Down List	Y
KNTA - User Accelerator - Enabled	Auto Complete List	N
KNTA - User Data - All	Auto Complete List	N
KNTA - User Data Context Field	Auto Complete List	N
KNTA - User Data Type - All	Drop Down List	N
KNTA - User Data Type - Enabled	Auto Complete List	N
KNTA - User Data Validations - Enabled	Auto Complete List	N
KNTA - User Id - All	Auto Complete List	N
KNTA - User Id - Enabled	Auto Complete List	N
KNTA - User Names - All	Auto Complete List	N
KNTA - User Names - Enabled	Auto Complete List	N
KNTA - User Security Action	Drop Down List	N
KNTA - Validation Name (Migrator Source)	Auto Complete List	N
KNTA - Validation Names	Auto Complete List	N
KNTA - Validation Type	Drop Down List	N
KNTA - Validations (Exclude Table, Budget, Staffing Profile and Resource Pool)	Auto Complete List	N
KNTA - Validations (For Table Column Headers)	Auto Complete List	N
KNTA - Validations - All	Auto Complete List	N
KNTA - Validations - Lookups	Auto Complete List	N
KNTA - Workflow Name (Migrator Source)	Auto Complete List	N
KNTA - Workflow Steps	Auto Complete List	N
KNTA - Workflows - Enabled	Auto Complete List	N
KRSC - Organization Unit Member Action	Drop Down List	N
KRSC - Resource	Auto Complete List	N
KRSC - Resource Pool Names	Resource Pool	N
KRSC - Resource Pool Names - All	Auto Complete List	N
KRSC - Skill	Auto Complete List	N
KRSC - Staffing Profile Names	Staffing Profile	N
KRSC - Staffing Profile Names - All	Auto Complete List	N
Master Projects - Cost Enabled and Not New	Auto Complete List	N
Numeric Text Field	Text Field	Y
Numeric Text Field (length = 4)	Text Field	Y
Numeric Text Field - 10 decimals	Text Field	N
Numeric Text Field - 2 decimals	Text Field	N
PMO - Business Objective States	Drop Down List	Y

Table 11-21. Kintana System Validations

Validation Name	Component Type	Use in Workflow?
PMO - Business Objectives	Auto Complete List	N
PMO - CR Level	Drop Down List	Y
PMO - Issue Escalation Level	Drop Down List	N
PMO - Master Projects	Auto Complete List	N
PMO - Period	Drop Down List	N
PMO - Program Names	Auto Complete List	N
PMO - Program Projects	Auto Complete List	N
PMO - Program State	Drop Down List	N
PMO - Program State AC	Auto Complete List	N
PMO - Request Entities	Drop Down List	N
PMO - Risk Impact	Drop Down List	Y
PMO - Risk Probability	Drop Down List	Y
PMO - Scope Change Severity	Drop Down List	Y
PMO - Sort By for Programs	Drop Down List	N
PMO - Summary Condition Impact	Auto Complete List	N
PMO - Summary Condition Priority	Auto Complete List	N
PMO - Summary Condition Probability	Auto Complete List	N
PMO - Summary Condition Severity	Auto Complete List	N
Parameter Column	Drop Down List	N
Password Field	Password Field	N
RM - All Distributions By Name	Auto Complete List	N
RM - All Releases By ID	Auto Complete List	N
RM - Distribution Detail Order By	Drop Down List	N
RM - Object Types in Distribution	Auto Complete List	N
RM - Ready for Release	Drop Down List	Y
RM - Release Status	Drop Down List	N
RM - Releases - All	Auto Complete List	N
RM - Releases - Open	Auto Complete List	N
RM - Workflow Step Statuses For Step	Auto Complete List	N
RM Package - Filterable Fields	Auto Complete List	N
RSC - Location	Drop Down List	N
RSC - Location - Autocomp	Auto Complete List	N
RSC - Org Unit Category	Drop Down List	N
RSC - Org Unit Category - Autocomp	Auto Complete List	N
RSC - Org Unit ID - Enabled	Auto Complete List	N
RSC - Org Unit Name - All	Auto Complete List	N
RSC - Org Unit Sort By	Drop Down List	N
RSC - Org Units - Enabled (non-seeded)	Auto Complete List	N
RSC - Period Numbers	Drop Down List	N
RSC - Projects w Staffing Profiles	Auto Complete List	N
RSC - Resource Category	Drop Down List	N

Table 11-21. Kintana System Validations

Validation Name	Component Type	Use in Workflow?
RSC - Resource Category - Autocomp	Auto Complete List	N
RSC - Resource ID - Enabled	Auto Complete List	N
RSC - Resource Managers	Auto Complete List	N
RSC - Resource Pool	Auto Complete List	N
RSC - Resource Pool Sort By	Drop Down List	N
RSC - Resource Pool Status	Drop Down List	N
RSC - Resource Pool Status w/o Approved	Drop Down List	N
RSC - Resource Pools (restricted)	Auto Complete List	N
RSC - Resource Sort By	Drop Down List	N
RSC - Resource Title	Drop Down List	N
RSC - Resource Title - Autocomp	Auto Complete List	N
RSC - Resources - Enabled (non-seeded)	Auto Complete List	N
RSC - Skill Category	Drop Down List	N
RSC - Skill Name - All	Auto Complete List	N
RSC - Skill Proficiency	Drop Down List	N
RSC - Skills - Enabled	Auto Complete List	N
RSC - Staffing Profile Id - All	Auto Complete List	N
RSC - Staffing Profile Sort By	Drop Down List	N
RSC - Staffing Profile Status w/o Approved	Drop Down List	N
RSC - Status	Drop Down List	N
RSC - Vis - Assignment Load Group By	Drop Down List	N
RSC - Vis Resource Pool Group By	Drop Down List	N
RSC - Workload Category	Drop Down List	N
RTRULES_EXCLUDED_FIELDS	Drop Down List	N
Radio Buttons (Y/N)	Radio Button (Yes/No)	N
References	Drop Down List	N
Rule Types	Drop Down List	N
Sql Command	Drop Down List	N
Sub Paths	Drop Down List	N
TMG - Approval Types	Drop Down List	N
TMG - Approvals Search Order	Drop Down List	N
TMG - Charge Code Categories - All	Drop Down List	N
TMG - Charge Code Categories - Enabled	Drop Down List	N
TMG - Charge Code Filter Types	Drop Down List	N
TMG - Charge Code Id - All	Auto Complete List	N
TMG - Charge Code Id - Enabled	Auto Complete List	N
TMG - Clients - All	Auto Complete List	N
TMG - Clients - Enabled	Auto Complete List	N
TMG - Days of Week	Drop Down List	N
TMG - Managers	Auto Complete List	N
TMG - Master Projects - Enabled	Auto Complete List	N

Table 11-21. Kintana System Validations

Validation Name	Component Type	Use in Workflow?
TMG - Misc. Work Items	Drop Down List	N
TMG - Period Type Id - All	Auto Complete List	N
TMG - Period Type Id - Enabled	Auto Complete List	N
TMG - Resource Group Id - All	Auto Complete List	N
TMG - Resource Group Id - Enabled	Auto Complete List	N
TMG - Resource Id - All	Auto Complete List	N
TMG - Resource Id - Enabled	Auto Complete List	N
TMG - Resource Id - Restricted	Auto Complete List	N
TMG - Resource Pool Resource Group	Auto Complete List	N
TMG - Resource Types	Drop Down List	N
TMG - Time Entry Durations	Drop Down List	N
TMG - Time Entry Units	Drop Down List	N
TMG - Time Period Calculation Types	Drop Down List	N
TMG - Time Periods	Auto Complete List	N
TMG - Time Sheet Details - Work Item	Auto Complete List	N
TMG - Time Sheet Search Order	Drop Down List	N
TMG - Time Sheet Statuses	Drop Down List	N
TMG - Work Allocation Search - Work Items	Auto Complete List	N
TMG - Work Allocation Search Order	Drop Down List	N
TMG - Work Allocation Sets - Allocation Search	Auto Complete List	N
TMG - Work Allocation Statuses	Drop Down List	N
TMG - Work Allocation Wide Sort By	Drop Down List	N
TMG - Work Allocation Work Bench - Work Item	Auto Complete List	N
TMG - Work Item Sets	Auto Complete List	N
TMG - Work Item Types	Drop Down List	N
TMG - Work Items	Auto Complete List	N
TRANSFER_PROTOCOL	Drop Down List	N
Text Area	Text Area	N
Text Area - 1800	Text Area	N
Text Field - 200	Text Field	Y
Text Field - 40	Text Field	Y
Time Format	Drop Down List	Y
URL	Web Address (URL)	N
User Data Column	Drop Down List	N
User Sign Off	Drop Down List	Y
VC - Applications Existing	Auto Complete List	N
VC - Branching	Drop Down List	N
VC - Check In Source	Auto Complete List	Y
VC - Check Out Destination	Auto Complete List	Y
VC - Dev Environment	Drop Down List	N
VC - Dev Pkg Number	Auto Complete List	N

Table 11-21. Kintana System Validations

Validation Name	Component Type	Use in Workflow?
VC - Directory Chooser	Directory Chooser	N
VC - Employees	Auto Complete List	N
VC - Existing Sub Paths	Auto Complete List	Y
VC - File Chooser	File Chooser	N
VC - File Revisions	Auto Complete List	N
VC - File Type	Drop Down List	N
VC - Files Existing	Auto Complete List	Y
VC - Report Type	Drop Down List	N
Version Control Application codes	Auto Complete List	N
WF - Allowed Package Workflows	Auto Complete List	N
WF - Approval Step	Drop Down List	Y
WF - Approval Step w/ Cancel	Drop Down List	Y
WF - Approval Step w/Cancel	Drop Down List	Y
WF - Child Package Results	Drop Down List	Y
WF - Close Statuses	Drop Down List	Y
WF - Decisions - All	Auto Complete List	N
WF - Decisions Required	Drop Down List	N
WF - Default Package Workflow	Auto Complete List	N
WF - Default Request Types	Auto Complete List	N
WF - Evaluation Step	Drop Down List	Y
WF - Events	Drop Down List	N
WF - Execution Step w/ Retry and Abort	Drop Down List	Y
WF - Execution Type	Drop Down List	N
WF - Executions - All	Auto Complete List	N
WF - Jump/Receive Step Labels	Drop Down List	N
WF - Migrate Step	Drop Down List	Y
WF - Notification Intervals - Enabled	Drop Down List	N
WF - Package Status	Drop Down List	N
WF - Parent Status Code - All	Drop Down List	N
WF - Processing Type	Drop Down List	N
WF - Product Scope (with ALL)	Drop Down List	N
WF - Product Scopes	Drop Down List	N
WF - QA Test Step	Drop Down List	Y
WF - Recipient Tokens - Enabled	Auto Complete List	N
WF - Recipient Tokens w/ no Security Groups - Enabled	Auto Complete List	N
WF - Request to Child Package Reference Relationships	Drop Down List	N
WF - Request to Child Request Reference Relationships	Drop Down List	N
WF - Rework Code	Drop Down List	Y
WF - Show All Workflow Steps	Drop Down List	N
WF - Standard Condition Results	Drop Down List	Y
WF - Standard Execution Results	Drop Down List	Y

Table 11-21. Kintana System Validations

Validation Name	Component Type	Use in Workflow?
WF - Standard Execution Results w/ Reset, Abort	Drop Down List	Y
WF - Standard Execution Results w/ Reset, Rejected	Drop Down List	Y
WF - Standard Execution Results w/ Reset, Reset Failures, Abort	Drop Down List	Y
WF - Step Authentication Type	Drop Down List	N
WF - Step Security Type	Drop Down List	N
WF - Step Security Type Choices	Drop Down List	N
WF - Step Timeout Type	Drop Down List	Y
WF - Timeout Unit	Drop Down List	Y
WF - Tokens - Enabled	Auto Complete List	N
WF - Txn Operators - NonNumeric	Drop Down List	N
WF - Txn Operators - Numeric	Drop Down List	N
WF - Validations	Auto Complete List	N
WF - Workflow Command - Enabled	Drop Down List	N
WF - Workflow Errors	Drop Down List	N
WF - Workflow ID - All	Auto Complete List	N
WF - Workflow ID w/o Subworkflow	Auto Complete List	N
WF - Workflow Name - All	Auto Complete List	N
WF - Workflow Step Display Types	Drop Down List	N
WF - Workflow Steps - All	Auto Complete List	N
Web Address	Web Address (URL)	N
Yes No Radio Buttons	Radio Button (Yes/No)	N
Yes or No Drop Down list	Drop Down List	Y
Yes or No Drop Down list with ALL	Drop Down List	Y

Appendix

C

Tokens

While configuring certain features in Kintana, it is often necessary to reference information that is undefined until the Kintana product is actually used a particular context. Instead of generating objects that are valid only in specific contexts, Kintana uses variables can be used to facilitate the creation of general objects that can be applied to a variety of contexts. These variables are called tokens.

There are two types of tokens found within Kintana: custom tokens and standard tokens. Standard tokens are provided with the product. Custom tokens are generated to suit specific needs. Each field of the following Kintana entities can be referenced as a custom token:

- Object Types
- Request Types and Request Header Types
- Report Types
- User Data
- Workflow Parameters

In addition, numerous standard Tokens are available that provide other useful pieces of information related to the Kintana system. For example, Kintana has a Token that represents the users currently logged onto the system.

For instructions on using Tokens and for a list of available system Tokens, see *"Using Commands and Tokens"*.

Appendix D

User Data Creation and Processing

Every entity in Kintana (such as Packages, Workflows, Requests and Projects) has a set of standard fields that provide information about the entity. While these fields are normally sufficient for day to day processing, it is possible to capture additional information specific to each organization. Kintana's User Data provides the ability to capture this additional information.

For every major entity in Kintana, up to 20 User Data fields can be defined. These fields are displayed in the **USER DATA** tab for the specific entity. The major attributes of each of these fields, such as their graphical presentation, the validation method, and whether or not they are required can be configured.

User Data fields are available for each entity instance generated; the fields are available globally. For example, you can configure a **MANAGER** field to appear on the **USER DATA** tab in the **USER** window. You could then specify each user's manager when setting up their Kintana account.

For some entities, context-sensitive custom fields can be set up. For example, User Data fields could be defined for the Request entities that are only available when the priority of a Request is **CRITICAL**.

The following entities support User Data functionality:

- Budgets
- Organizations
- Resource Pools
- Staffing Profiles
- Packages
- Package Lines
- Environments

- Environment Application
- Environment Refresh
- Requests
- Request Types
- Projects
- Tasks
- Security Groups
- Users
- Validation Values
- Workflows
- Workflow Steps
- Workflow Executions
- Workflow Decisions

The following topics are discussed:

- [“Creating and Editing Kintana User Data”](#) on page 352
- [“Creating and Editing Context Sensitive User Data”](#) on page 362
- [“Project/Task User Data Roll-Up”](#) on page 375
- [“Referring to User Data”](#) on page 388
- [“Migrating User Data”](#) on page 388



Note

For information on screens and fields in the User Data Workbench, refer to the [“Configuration Workbench Reference”](#) document.

Creating and Editing Kintana User Data

The following sections provide detailed instructions for creating and editing Kintana User Data:

- [Adding User Data Fields](#)
- [Copying a Field's Definition](#)
- [Editing User Data Fields](#)
- [Configuring User Data Field Dependencies](#)
- [Removing Fields](#)
- [Modifying the User Data Layout](#)



Note

To configure User Data, you must have the **CONFIG: EDIT USER DATA** Access Grant.

Adding User Data Fields

To generate a new User Data field:

1. Click the **CONFIGURATION** screen group and click the **USER DATA** screen. The **USER DATA WORKBENCH** opens.
2. Search for the desired User Data Type from the **QUERY** tab and select it from the **RESULTS** tab of the **USER DATA WORKBENCH**.
3. Click **OPEN**. The **USER DATA** window opens.
4. Click **NEW** in the **FIELDS** Tab. The **FIELD: NEW** window opens.

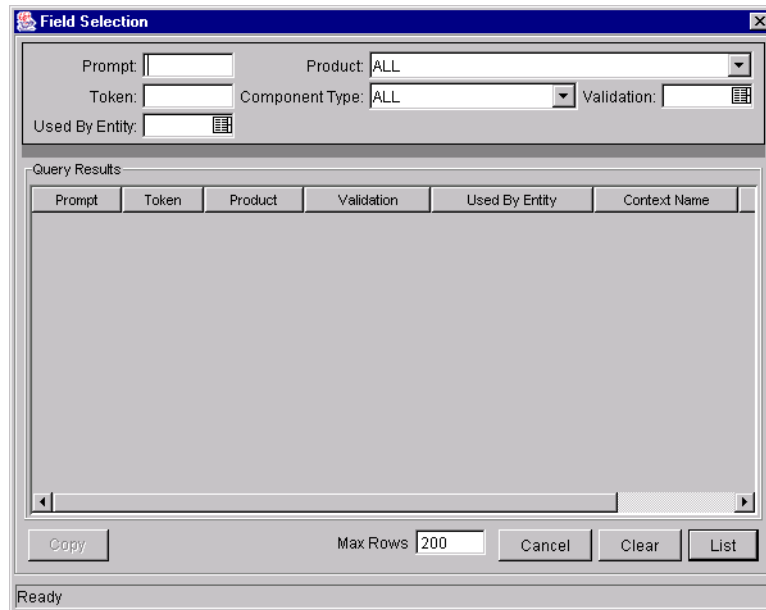
5. Enter the general information region fields for the User Data field. Fields appearing in the general information region are defined in "[Configuration Workbench Reference](#)".
6. Click the **ATTRIBUTES** tab to define the field's basic properties (DISPLAY, DISPLAY ONLY, REQUIRED). Fields appearing in the **ATTRIBUTES** tab are defined in "[Configuration Workbench Reference](#)".
7. Click the **DEFAULTS** tab to define the default value for that field. Fields appearing in the **DEFAULTS** tab are defined in "[Configuration Workbench Reference](#)".
8. Click the **DEPENDENCIES** tab to define the field dependent properties of the field (CLEAR WHEN, DISPLAY ONLY WHEN, REQUIRED WHEN). Fields appearing in the **DEPENDENCIES** tab are defined in "[Configuration Workbench Reference](#)".
9. Click **OK** to add the User Data field to the entity.

Copying a Field's Definition

The **COPY FROM** functionality can also be utilized to streamline the process of adding Fields by copying the definition of existing Fields.

To copy a field's definition:

1. Click the **CONFIGURATION** screen group and click the **USER DATA** screen. The **USER DATA WORKBENCH** opens.
2. Search for the desired User Data Type from the **QUERY** tab and select it from the **RESULTS** tab of the **USER DATA WORKBENCH**.
3. Click **OPEN**. The **USER DATA** window opens.
4. Click **NEW** in the **FIELDS** Tab. The **FIELD: NEW** window opens.
5. Click **COPY FROM**. The **FIELD SELECTION** window opens.



6. Search for a field to copy. Query fields by a number of criteria, such as the Token Name or field prompt. It is also possible to perform more complex queries such as listing all fields that reference a certain Validation or are used by a certain entity.

Note

Because of the large number of fields in the Kintana system, you should limit the list of fields by one or more of the query criteria.

7. Select the desired field.
8. Click **COPY**. This closes the window and copies the definition of the selected field into the **FIELD: NEW** window.
9. Make any necessary modifications and click **OK**.

Editing User Data Fields

To edit an existing field:

1. Click the **CONFIGURATION** screen group and click the **USER DATA** screen. The **USER DATA WORKBENCH** opens.
2. Search for the desired User Data Type from the **QUERY** tab and select it from the **RESULTS** tab of the **USER DATA WORKBENCH**.

3. Click **OPEN**. The USER DATA window opens.
4. Select the field you wish to edit.
5. Either double-click on the Field in the **FIELDS** tab or select the field and click **EDIT**. This opens up a FIELD window.

6. Make the desired changes in the header region, **ATTRIBUTES** tab, **DEFAULT** tab, and **DEPENDENCIES** tab.
7. Click **APPLY** to save the changes to the **FIELDS** tab without closing the FIELD window, or click **OK** to save the changes and close the FIELD window.

The field has been updated with the changes.

Configuring User Data Field Dependencies

Field behavior and properties can be linked to the value of other fields defined for that entity.



A Report Type field can become required when the value in another field in that Report Type is **CRITICAL**.

A field can be configured to:

- Clear when another field changes.
- Become read only when another field meets a logical condition, defined in [Table 11-22](#).

- Become required when another field meets a logical condition, defined in [Table 11-22](#).

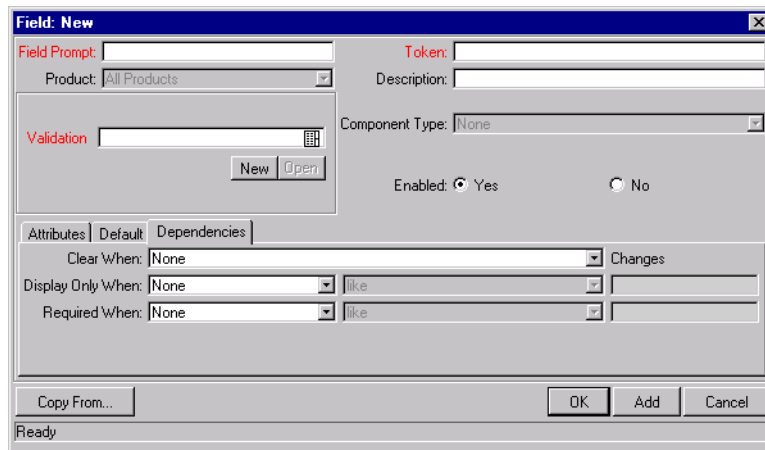
Table 11-22. Field Dependency Logical Qualifiers

Logical qualifier	Definition
like	A 'like' condition looks for close matches of the value to the contents of the field chosen.
not like	A 'not like' condition looks for contents in the selected field that are not close matches to the Value field.
is equal to	An 'is equal to' condition looks for an exact match of the Value to the contents of the Field chosen.
is not equal to	An 'is not equal to' condition is true when there are no results exactly matching the value of the field contents.
is null	An 'Is null' condition is true when the field selected is blank.
is not null	An 'Is not null' condition is true when the field selected is not blank.
is greater than	An 'Is greater than' condition looks for a numerical value larger than the value entered in the Value field.
is less than	An 'Is less than' condition looks for a numerical value below the value entered in the Value field.
is less than equal to	An 'Is less than equal to' condition looks for a numerical value below or the same as the value entered in the Value field.
is greater than equal to	An 'Is greater than equal to' condition looks for a numerical value larger than or the same as the value entered in the Value field.

To configure a User Data field dependency:

1. Click the **CONFIGURATION** screen group and click the **USER DATA** screen. The **USER DATA WORKBENCH** opens.
2. Search for the desired User Data Type from the **QUERY** tab and select it from the **RESULTS** tab of the **USER DATA WORKBENCH**.
3. Click **OPEN**. The **USER DATA** window opens.
4. Select the field you wish to edit.

5. Either double-click on the Field in the **FIELDS** tab or select the field and click **EDIT**. The **FIELD** window opens.
6. Click the **DEPENDENCIES** tab.



The screenshot shows the 'Field: New' dialog box with the 'Dependencies' tab selected. The dialog is divided into several sections:

- Field Prompt:** A text input field.
- Token:** A text input field.
- Product:** A dropdown menu currently set to 'All Products'.
- Description:** A text input field.
- Validation:** A text input field with a small icon to its right. Below it are 'New' and 'Open' buttons.
- Component Type:** A dropdown menu currently set to 'None'.
- Enabled:** Radio buttons for 'Yes' (selected) and 'No'.
- Attributes | Default | Dependencies:** A tabbed interface with 'Dependencies' selected.
- Clear When:** A dropdown menu set to 'None'.
- Changes:** A dropdown menu.
- Display Only When:** A dropdown menu set to 'None', followed by a logical qualifier dropdown set to 'like', a checkbox checked, and a field name input.
- Required When:** A dropdown menu set to 'None', followed by a logical qualifier dropdown set to 'like', a checkbox checked, and a field name input.
- Buttons:** 'Copy From...', 'OK', 'Add', and 'Cancel'.
- Status:** 'Ready'.

7. Set the field dependencies. It is possible to:
 - Select a field name from the **CLEAR WHEN** drop down list to indicate that the current field should be cleared when the selected field changes.
 - Select a field name from the **DISPLAY ONLY WHEN** drop down list to indicate that the current field should for display only (i.e. not editable) when certain logical criteria are satisfied. This field functions with two adjacent fields. These are a drop down list containing logical qualifier and another field which dynamically changes to a date field, drop down list, or text field, depending on the selected field's validation.
 - Select a field name from the **REQUIRED WHEN** drop down list to indicate that the current field should be required when certain logical criteria are satisfied. This field functions with two adjacent fields. These are a drop down list containing logical qualifier and another field which dynamically changes to a date field, drop down list, or text field, depending on the selected field's Validation.
8. Click **OK**.

Removing Fields

To remove a field permanently from a User Data Type:

1. Click the **CONFIGURATION** screen group and click the **USER DATA** screen. The USER DATA WORKBENCH opens.
2. Search for the desired User Data Type from the **QUERY** tab and select it from the **RESULTS** tab of the USER DATA WORKBENCH.
3. Click **OPEN**. The USER DATA window opens.
4. Select the field in the **FIELDS** tab.
5. Click **REMOVE**.
6. Click **OK** to save the change to the database and close the window.

Modifying the User Data Layout

The layout of User Data fields can be changed in the **LAYOUT** tab of the USER DATA window.

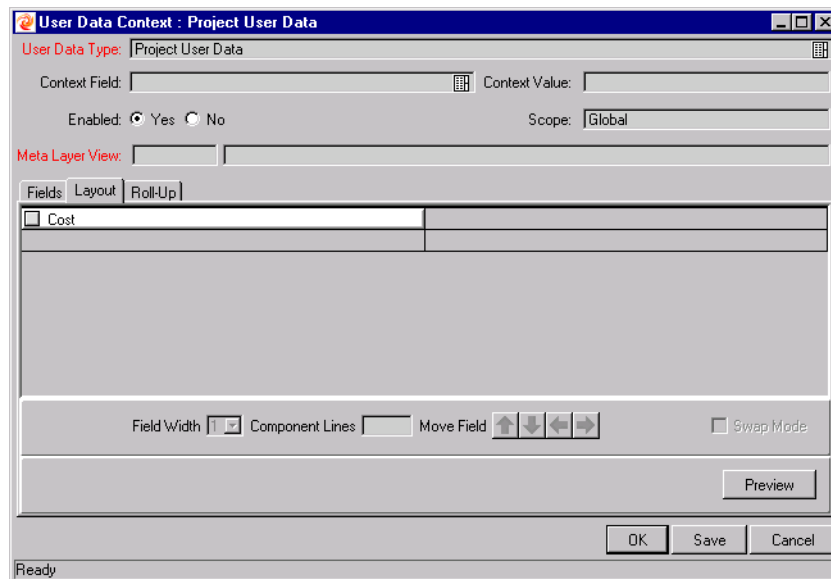


Figure 11-10 User Data Window - Layout Tab

The following sections discuss modifying User Data field layout in more detail:

- [Changing Column Width](#)
- [Moving a Field](#)

- *Swapping Positions of Two Fields*

Changing Column Width

To change the column width of a Field:

1. Click the **CONFIGURATION** screen group and click the **USER DATA** screen. The **USER DATA WORKBENCH** opens.
2. Search for the desired User Data Type from the **QUERY** tab and select it from the **RESULTS** tab of the **USER DATA WORKBENCH**.
3. Click **OPEN**. The **USER DATA** window opens.
4. Click the **LAYOUT** tab.
5. Select the Field.
6. Select either **1** or **2** in the **FIELD WIDTH** radio button.



Note

The Layout editor will not allow changes to be made if it conflicts with another field in the layout (for example, a field's width cannot be changed from one to two if another field exists in column two on the same row).

Additionally, for fields of component type **TEXT AREA**, it is possible to determine the number of lines the text area will display. Select the **TEXT AREA** type field and change the value in the **COMPONENT LINES** attribute. If the selected field is not of type **TEXT AREA**, this attribute will be blank and non-updateable.

Moving a Field

To move a field or a set of fields:

1. Click the **CONFIGURATION** screen group and click the **USER DATA** screen. The **USER DATA WORKBENCH** opens.
2. Search for the desired User Data Type from the **QUERY** tab and select it from the **RESULTS** tab of the **USER DATA WORKBENCH**.
3. Click **OPEN**. The **USER DATA** window opens.
4. Click the **LAYOUT** tab.

5. Select the field(s). To select more than one field, press the Shift key while selecting the last field in a set. It is only possible to select a continuous set of fields.
6. Use the directional arrow buttons to move the fields to the desired location in the layout builder.



A field, or a set of fields, cannot be moved to an area where other fields already exist. Those other fields must be moved out of the way first.

Swapping Positions of Two Fields

To swap the positions of two fields:

1. Click the **CONFIGURATION** screen group and click the **USER DATA** screen. The USER DATA WORKBENCH opens.
2. Search for the desired User Data Type from the **QUERY** tab and select it from the **RESULTS** tab of the USER DATA WORKBENCH.
3. Click **OPEN**. The USER DATA window opens.
4. Click the **LAYOUT** tab.
5. Select the first field.
6. Select the **SWAP MODE** check box. This causes an **S** to appear in the check box area of the selected field.
7. Once the **S** appears, double-click on the field to be swapped with. This causes the two fields to change positions.
8. Following the swap, the Swap Mode is turned off.

The fields have now been swapped. To swap another set of fields, repeat the above procedure.

Previewing the Layout

To check what the layout will look like in actual use, click **PREVIEW**. This opens a small window that shows the fields as they will appear in the window, shown in [Figure 11-11](#).

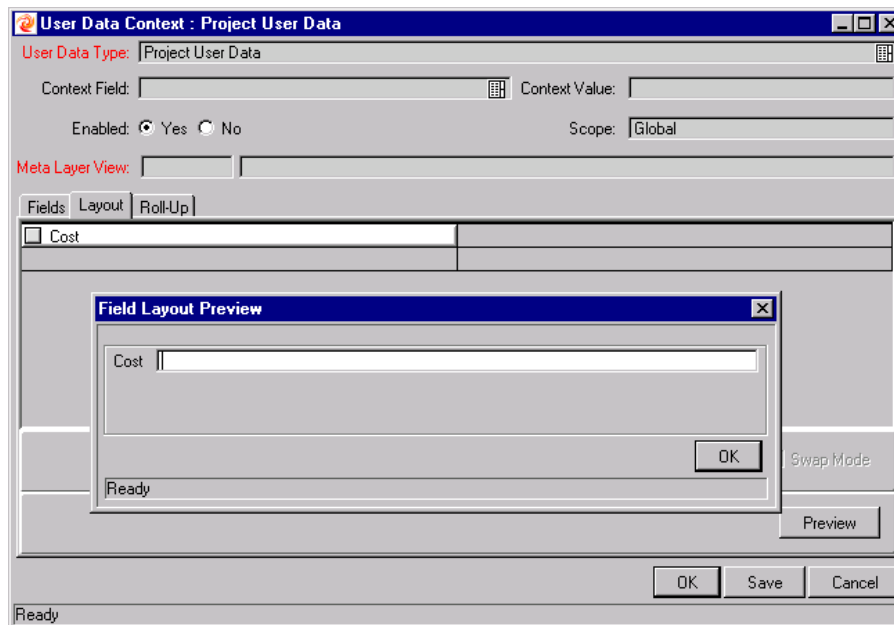


Figure 11-11 Layout Tab in Preview Mode

User Data fields are also visible in the Kintana interface.



Note

If all the fields have a width of one column, all displayed columns will automatically span the entire available area when an entity of the given User Data is being viewed or generated.

Non-displayed fields do not affect the layout. The layout engine considers them the same as a blank field.

Creating and Editing Context Sensitive User Data

The following sections provide detailed instructions for creating and editing Context Sensitive User Data:

- [Creating Context Sensitive User Data](#)
- [Editing Context Sensitive User Data](#)
- [Deleting Context Sensitive User Data](#)
- [Copying Context Sensitive User Data](#)

- [Example - Using Context Sensitive User Data for a Field in a Request Header Type](#)

Creating Context Sensitive User Data

Context Sensitive User Data can be defined for the Request, Package, and Validations (Validation value region) windows in the USER DATA WORKBENCH. To define Context Sensitive User Data:

1. Define a Context Field in the Global User Data scope. See [“Defining the Context Field”](#) on page 363.
2. Define a Context Value. See [“Defining a Context Value”](#) on page 364.
3. Define and configure the fields which appear under certain contexts. See [“Defining the Context Sensitive Fields”](#) on page 365.



Note

When defining or editing Context Sensitive User Data fields for the same User Data Type, make sure to save any changes to the Global User Data fields before working on the Context User Data fields.

Defining the Context Field

Only one field can be defined as the Context Field at any given time. To specify the Context Field:

1. Click the **CONFIGURATION** screen group and click the **USER DATA** screen. The USER DATA WORKBENCH opens.
2. Click **LIST** to display all of the existing User Data Types.

The screenshot shows the 'User Data Workbench' window with a table of records. The table has columns for 'User Data Type', 'Scope', 'Context Field', 'Context Value', and 'Enabled'. The records are as follows:

User Data Type	Scope	Context Field	Context Value	Enabled
Request Type User Data	Global			Y
Request User Data	Global	Application		Y
Request User Data	Context	Application	CSM App	Y
Request User Data	Context	Application	ERP Application	Y
Request User Data	Context	Application	HR Application	Y
Request User Data	Context	Application	HR Application	Y
Security Group User Data	Global			Y
Task User Data	Global			Y
User User Data	Global			Y
Validation Value User Data	Global	Validation Name		Y
Validation Value User Data	Context	Validation Name	CONNECTION_PR...	Y
Validation Value User Data	Context	Validation Name	TRANSFER_PROT...	Y

At the bottom of the window, there are buttons for 'New', 'Open', 'Copy', 'Delete', and 'Refresh'. A status bar at the very bottom indicates '27 User Data Context Records are loaded.'

3. Select the desired User Data Type (Package, Request or Validation) with a **GLOBAL** scope.
4. Click **OPEN**. The USER DATA CONTEXT window opens.
5. Select the CONTEXT FIELD from the auto-complete list.

Note

The CONTEXT FIELD is disabled if any specific contexts for the User Data has been defined. In order to change the CONTEXT FIELD, all specific contexts for the User Data Type must first be deleted. For more information on editing Context Sensitive User Data, see *“Editing Context Sensitive Fields”* on page 367.

6. Verify that the global context is enabled (ENABLED=YES).
7. Click **OK** to save and close the window.

The CONTEXT FIELD has now been specified.

Note

The CONTEXT FIELD for the Validations Value User Data Type is always **VALIDATION NAME**.

Defining a Context Value

Context Values are the predefined possible values for the selected CONTEXT FIELD. Different User Data fields can be defined to be displayed for each of the possible Context Values. To define a Context Value:

1. Click the **CONFIGURATION** screen group and click the **USER DATA** screen. The USER DATA WORKBENCH opens.

2. Click **NEW** in the **RESULTS** tab or **NEW USER DATA CONTEXT** in the **QUERY** tab. The **USER DATA CONTEXT** window opens.
3. Select a **USER DATA TYPE** from the auto-complete list. The **CONTEXT FIELD** is displayed as read-only.



Note

Only User Data Types with a defined **CONTEXT FIELD** appear in the list. To define a **CONTEXT FIELD** for Request, Package, or Validation Values, see [“Defining the Context Field”](#) on page 363.

4. Select a **CONTEXT VALUE** from the drop down list or auto-complete list.

Defining the Context Sensitive Fields

User Data fields to be used with the specified **CONTEXT VALUE** are defined and configured just as in other areas of the Kintana Product Suite. For details on defining the field content and layout, see one of the following sections:

- [“Adding User Data Fields”](#) on page 353
- [“Editing User Data Fields”](#) on page 355
- [“Removing Fields”](#) on page 358

To define different fields based on a different **CONTEXT VALUE**, see [“Defining a Context Value”](#) on page 364.

Editing Context Sensitive User Data

Context Sensitive User Data can be edited for the Request, Package, Environment, and Validations (Validation value region) windows in the **USER DATA WORKBENCH**. For details on editing Context Sensitive User Data, see one of the following sections:

- [Changing the Context Field](#)
- [Changing the Context Value](#)
- [Editing Context Sensitive Fields](#)

Changing the Context Field

In order to change the **CONTEXT FIELD**, all specific contexts for the User Data Type must first be deleted. To change the **CONTEXT FIELD** for a particular User Data Type:

1. Click the **CONFIGURATION** screen group and click the **USER DATA** screen. The **USER DATA WORKBENCH** opens.
2. Click **LIST** to display all of the existing User Data Types.
3. Locate the desired User Data Type.
4. Select the rows where the desired User Data Type's **SCOPE=CONTEXT**.
5. Click **DELETE**.
6. Select the desired User Data Type. It will have a **GLOBAL** Scope.
7. Click **OPEN**. The **USER DATA CONTEXT** window opens.
8. Select the **CONTEXT FIELD** from the auto-complete list.
9. Verify that the Global Context is enabled (**ENABLED=YES**).
10. Click **OK** to save and close the window.

The User Data Type's **CONTEXT FIELD** has now been changed.



Note

The **CONTEXT FIELD** for the **Validations Value** User Data Type is always **VALIDATION NAME** and cannot be changed.

Changing the Context Value

To change an existing User Data Type's **CONTEXT VALUE**:

1. Click the **CONFIGURATION** screen group and click the **USER DATA** screen. The **USER DATA WORKBENCH** opens.
2. Click **LIST** to display all of the existing User Data Types.
3. Select the desired User Data Type that has the **CONTEXT VALUE** to be changed.
4. Click **OPEN**. The **USER DATA CONTEXT** window opens.
5. Select a new **CONTEXT VALUE** from the drop down list or auto-complete list.
6. Click **OK** to save the changes and close the window.

The User Data's **CONTEXT VALUE** has been changed.

Editing Context Sensitive Fields

User Data fields to be used with the specified CONTEXT VALUE are edited just as in other areas of the Kintana Product Suite. For details on editing the field content and layout, see one of the following sections:

- [“Adding User Data Fields”](#) on page 353
- [“Editing User Data Fields”](#) on page 355
- [“Removing Fields”](#) on page 358

Deleting Context Sensitive User Data

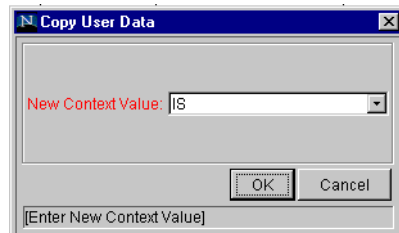
To delete a Context Sensitive User Data Type:

1. Click the **CONFIGURATION** screen group and click the **USER DATA** screen. The **USER DATA WORKBENCH** opens.
2. Click **LIST** to display all of the existing User Data Types.
3. Select the User Data Type to be deleted.
4. Click **DELETE**. A **QUESTION** dialog opens with the message “Delete 1 User Data Context[s]?”
5. Click **YES** to confirm deletion.

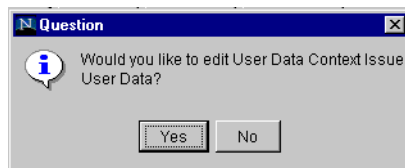
Copying Context Sensitive User Data

To copy a Context Sensitive User Data Type:

1. Click the **CONFIGURATION** screen group and click the **USER DATA** screen. The **USER DATA WORKBENCH** opens.
2. Click **LIST** to display all of the existing User Data Types.
3. Select the User Data Type with **SCOPE=CONTEXT** that is to be copied.
4. Click **COPY**. The **COPY USER DATA** window opens.



5. Select a New CONTEXT VALUE and click **OK**. A QUESTION dialog opens.



6. Click **YES** to edit the context sensitive fields or **No** to accept.

Example - Using Context Sensitive User Data for a Field in a Request Header Type

Different values can appear in the Request's APPLICATIONS field depending on which Request Header Type is used. For the APPLICATIONS field in an ERP Request, the following distinct set of fields are available:

- Accounts Receivable
- Accounts Payable
- General Ledger
- Inventory

For an eCommerce Request, the following fields are available:

- Registration
- User Preferences
- Order Entry
- Order Tracking



Note

Changing the Validation of a field in a Request Header Type can affect how information is returned from queries and reports. Kintana, therefore, recommends a context sensitive User Data approach when setting up such a system.

See "[Create Workbench Reference](#)" for more information on Request Header Types.

The following procedure provides an example for setting up the APPLICATIONS Validation for the ERP Request introduced above:

Setting Up the Context Sensitive User Data

First, the Context Sensitive User Data must be configured.

1. Click the **CONFIGURATION** screen group and click the **USER DATA** screen. The USER DATA WORKBENCH opens.
2. Click **NEW USER DATA CONTEXT**. The USER DATA CONTEXT window opens.
3. Select **VALIDATION VALUE USER DATA** from the USER DATA TYPE auto-complete list.
4. Select **KNTA - APPLICATION - ENABLED** from the CONTEXT VALUE auto-complete list.
5. Click **NEW**. The FIELD: NEW window opens.
6. Create a new field with the following specifications:
 - FIELD PROMPT = **USED BY REQUEST HEADER TYPE**
 - TOKEN = **REQUEST_HEADER_TYPE_ID**
 - VALIDATION = **CRT - REQUEST HEADER TYPES - ALL**

Field: Used By Request Header Type

Field Prompt: Used By Request Header Type Token: REQUEST_HEADER_TYPE_ID

Product: All Products Description:

Validation: >RT - Request Header Typ Component Type: Auto Complete List

Enabled: Yes No

Attributes: Default | Dependencies

User Data Col.: USER_DATA1 Display Only: Never

Display: Yes No Required: Always

OK Apply Cancel

Ready

Signed by: Kintana, Inc.

7. Click **OK**.

Example: Configuring the Validations

The Validation can now be configured:

1. Click the **CONFIGURATION** screen group and click the **VALIDATIONS** screen. The **VALIDATION WORKBENCH** opens.
2. Search for and open the global 'KNTA - Application - Enabled' Validation. The **VALIDATION** window opens.

Validation : KNTA - Application - Enabled

Name: KNTA - Application - Enabled

Description: KNTA - Application - Enabled

Enabled: Use in Workflow?

Component Type: Auto Complete List

Validated By: List

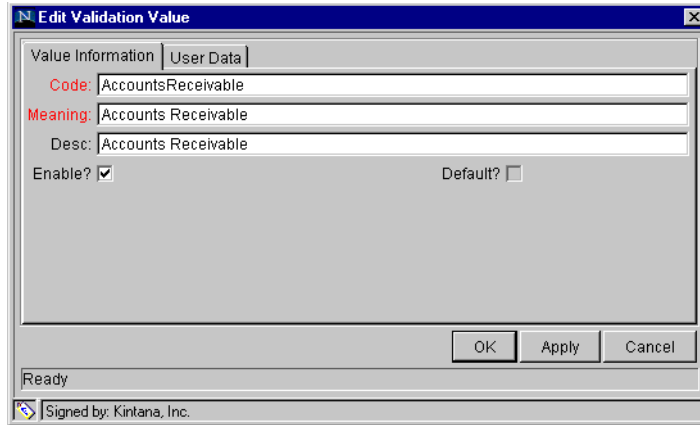
Seq	Code	Meaning	Description	Enabled	Default
1	Apps*Integrity_1.5	Apps*Integrity 1.5	Apps*Integrity 1.5	N	N
2	Apps*Integrity_1.6	Apps*Integrity 1.6	Apps*Integrity 1.6	N	N
3	Apps*Integrity_1.7	Apps*Integrity 1.7	Apps*Integrity 1.7	N	N
4	Apps*Integrity_2.0	Kintana Deliver	Kintana Deliver	Y	N
5	GL*Migrator_1.0	GL*Migrator	GL*Migrator	Y	N
6	Calendar	Calendar		N	N
7	Env*Integrity_1.0	Env*Integrity 1.0	Env*Integrity 1.0	N	N
8	Accelerators	Accelerators	Accelerators	Y	N
9	Support*Express	Support*Express		N	N

New Edit Delete Copy From ↑ ↓

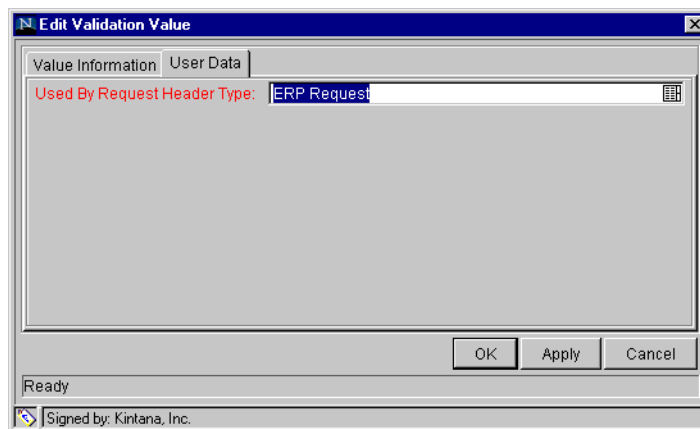
Used By Ownership OK Save Cancel

Ready (Read-Only, Seed Data)

3. Associate each Validation Value with the appropriate Request Header Type shown in *Table 11-23*. To associate a Validation Value with a Request Header Type:
 - a. Select a Validation Value from the VALIDATION VALUES list.
 - b. Click **EDIT**. The EDIT VALIDATION VALUE window opens.



- c. Click the **USER DATA** tab.
 - d. Select the Request Header Type (**ERP REQUEST** in this example) from the USED BY REQUEST HEADER TYPE auto-complete list.



- e. Click **OK**.
 - f. Repeat these steps for each row in *Table 11-23*.

Table 11-23. Validation values associated with Request Header Types

Validation Value	Used by Request Header Type
Accounts Receivable	ERP Request
Accounts Payable	ERP Request
General Ledger	ERP Request
Inventory	ERP Request
Registration	eCommerce Request
User Preferences	eCommerce Request
Order Entry	eCommerce Request
Order Tracking	eCommerce Request

Example: Modifying the SQL

You can now create variants of the standard Application Validation which vary depending on which Request Header Type is being used.

1. From the VALIDATIONS WORKBENCH, copy the 'KNTA - Application - All' Validation.
2. Rename the Validation to 'ERP Applications - All.'
3. Edit the Validation's SQL as shown below.

Validation : ERP Applications - All

Name: ERP Applications - All
 Description: ERP - Applications - All
 Enabled: Use in Workflow?
 Component Type: Auto Complete List
 Validated By: SQL

Seq	Column Header	Displayed	Col.
1	Lookup code	N	
2	Application	Y	
3	Default Flag	N	

SQL:

```
select LOOKUP_CODE, MEANING, DEFAULT_FLAG
from KNTA_LOOKUPS
from UPPER(MEANING) like UPPER('??%')
and LOOKUP_TYPE = 'APPLICATION'
and VISIBLE_USER_DATA1 = 'ERP Request'
order by 2
```

Buttons: New, Edit, Delete, Tokens, Used By, Ownership, OK, Save, Cancel

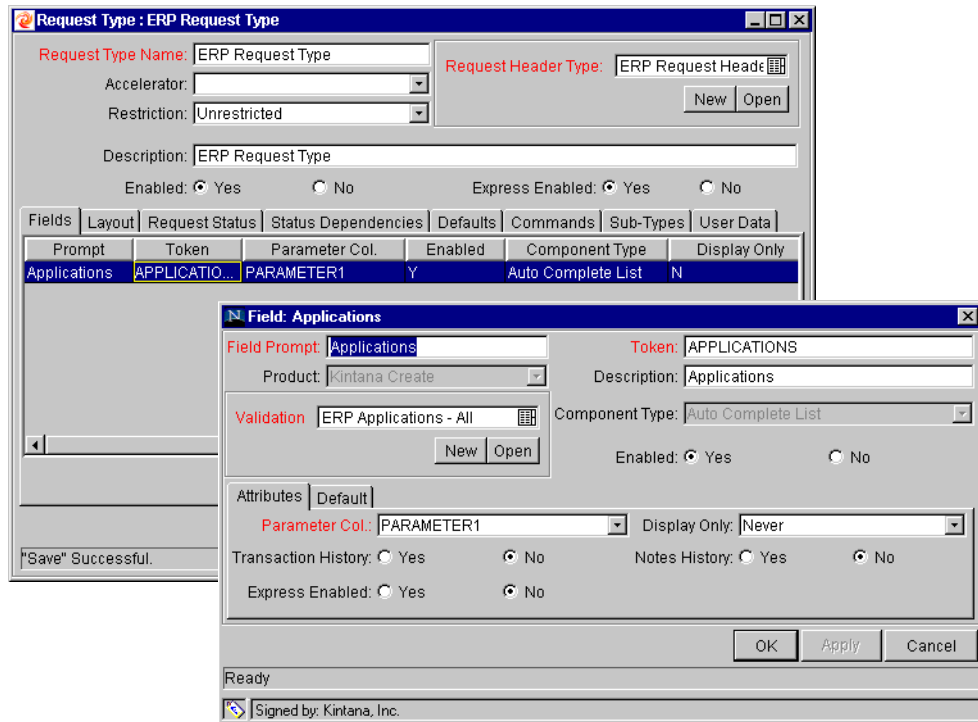
Ready

The specific variant for the ERP Request Header Type would be (assuming that the context-specific user data field was captured in the USER_DATA1 column):

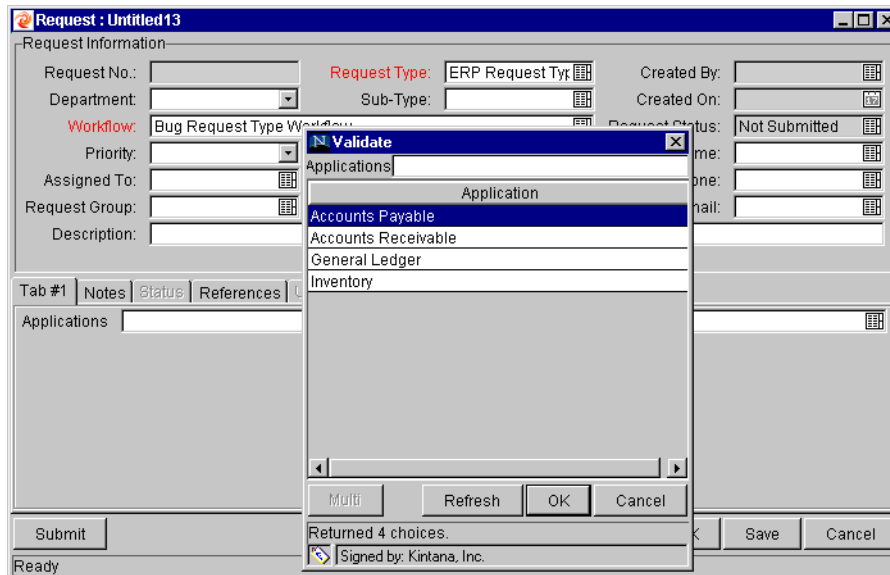
```
select LOOKUP_CODE, MEANING, DEFAULT_FLAG
from KNTA_LOOKUPS
where UPPER(MEANING) like UPPER('??%')
and LOOKUP_TYPE = 'APPLICATION'
and VISIBLE_USER_DATA1 = 'ERP Request'
order by 2
```

Example: Resulting Behavior

You can now create a context-sensitive APPLICATIONS field for any Request Type. In this example, simply create a new field with the Validation 'ERP - Applications - All.'



When a user creates a Request with this Request Type (which references the ERP Request Header Type), the following Validations are associated with the **APPLICATIONS** field.



Project/Task User Data Roll-Up

Values from Task User Data fields can be configured to “roll up” (combine/process values in a meaningful way) into parent Project User Data fields. The following types of Task User Data can roll up into Project User Data:

- Numeric fields (Text Field component type with Numeric data mask)
- Date fields

For each Project, a Project User Data field can show a roll-up of Task User Data values using one of the following methods:

- **AVERAGE** — Shows the average of all values of a specified Task User Data field for every Task under the Project (Numeric fields).
- **MAXIMUM** — Shows the largest of all values of a specified Task User Data field for every Task under the Project (Numeric and Date fields).
- **MINIMUM** — Shows the smallest of all values of a specified Task User Data field for every Task under the Project (Numeric and Date fields).
- **SUM** — Shows the summation of all values of a specified Task User Data field for every Task under the Project (Numeric fields).

Project/Task User Data Roll-Up can be used to capture various important aspects of a Project.



Example

Using the **AVERAGE** Roll-Up Method, the average cost of all a Project’s Tasks can be easily determined and automatically recalculated each time a Task is updated.

Using the **MAXIMUM** Roll-Up Method, the latest date out of a Project’s Tasks can be captured.

Using the **MINIMUM** Roll-Up Method, the earliest date out of a Project’s Tasks can be captured.

Using the **SUM** Roll-Up Method, the total cost of a Project’s Tasks can be easily determined and automatically recalculated each time a Task is updated.

Creating Project/Task User Data Roll-Up

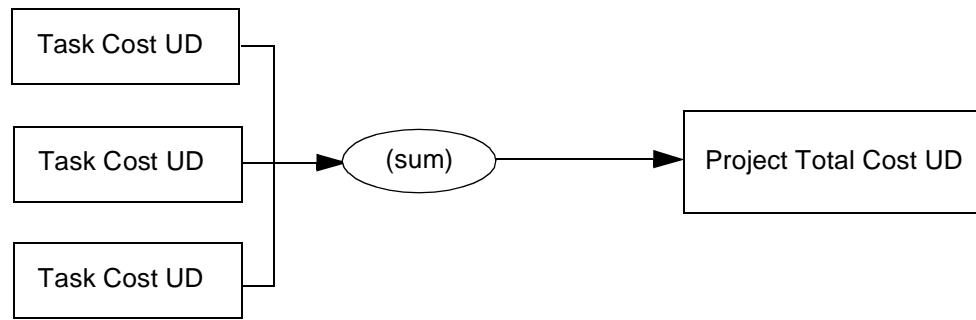
User Data must be configured for Projects and Tasks before specifying Roll-Up Methods.

1. Create and configure Project and Task User Data fields.
2. Link Project and Task User Data fields with Roll-Up Method.

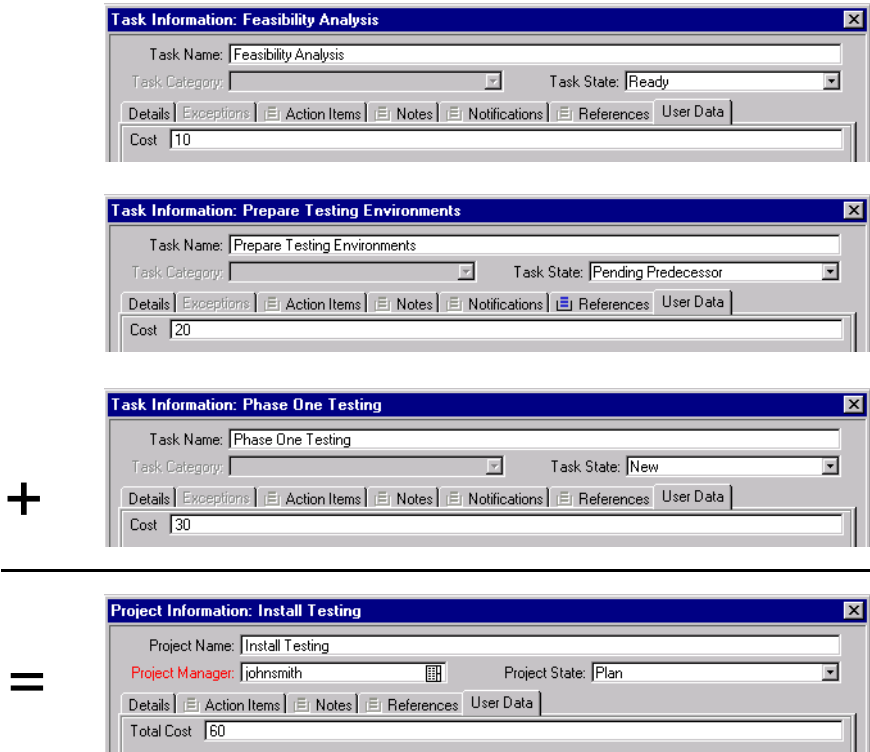
For more detailed information on configuring User Data, see [“Creating and Editing Kintana User Data”](#) on page 352.

Example: Using Project/Task User Data Roll-Up

Company X would like to capture total cost for its Projects. Total Project cost in this case is to be calculated by adding the costs of individual Tasks. User Data fields for Task cost and Project total cost are each defined. The relationship is illustrated below:



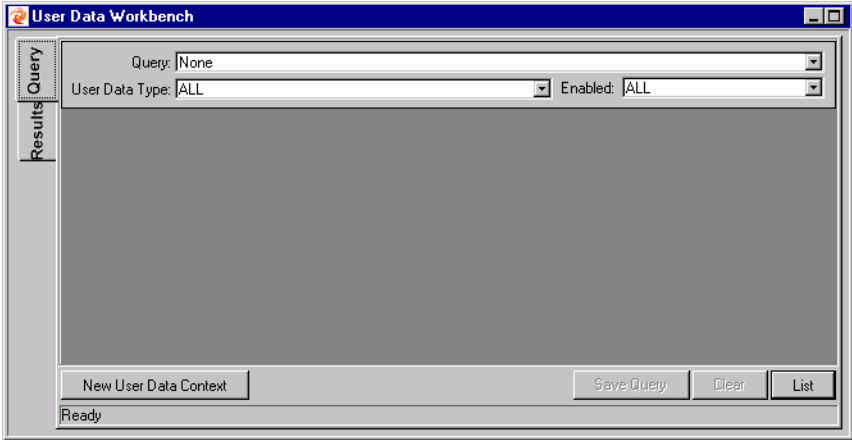
Each Task has its own Cost User Data field. The values for each Task Cost User Data field are rolled up using the **SUM** Roll-Up Method into the Project Total Cost User Data field.



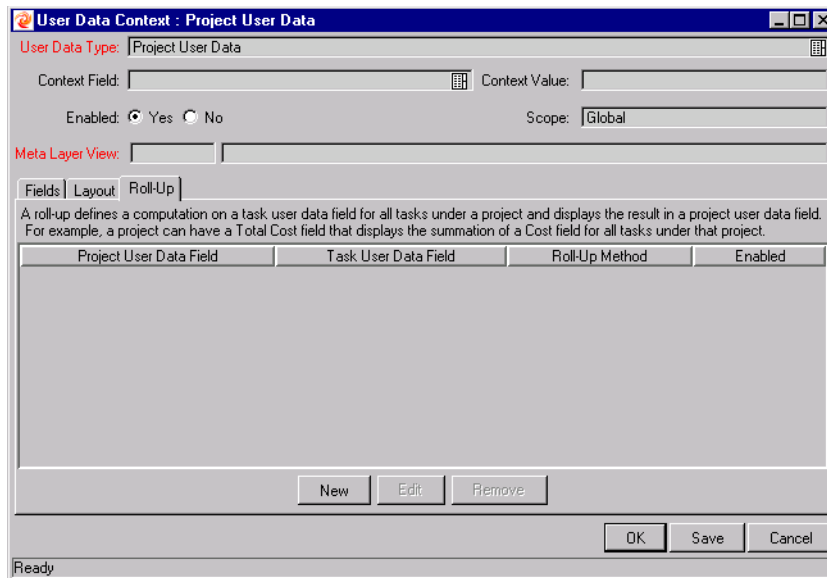
Once Project and Task User Data fields have been configured and saved, the Roll-Up relationship can be specified.

To specify the Roll-Up Method:

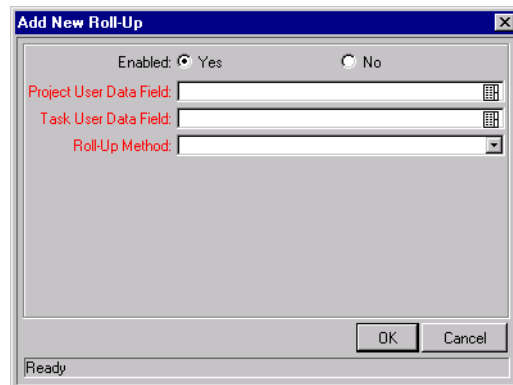
- 1. Click the **CONFIGURATION** screen group and click the **USER DATA** screen. The **USER DATA WORKBENCH** opens.



2. Select **PROJECT USER DATA** from the USER DATA TYPE drop down list and click **LIST**. The **RESULTS** tab opens with the Project User Data Type loaded.
3. Select the Project User Data and click **OPEN**.
4. Click the **ROLL-UP** tab.



5. Click **NEW**. The ADD NEW ROLL-UP window opens.



6. Select the **PROJECT USER DATA FIELD** that will contain rolled-up Task User Data values.
7. Select the **TASK USER DATA FIELD** whose values will roll up into the chosen **PROJECT USER DATA FIELD**.
8. Select the **ROLL-UP METHOD** from the drop down list.

The drop down list will only display valid options for the data types of the Project and Task User Data fields.

9. Select **YES** to enable the Roll-Up.
10. Click **OK**.

The Roll-Up relationship is added to the **ROLL-UP** tab.

11. Click **SAVE**.



Note

Only two User Data fields of the same type can be selected for Roll-Up (for example, a Numeric field cannot roll up into a Date field, nor vice versa).

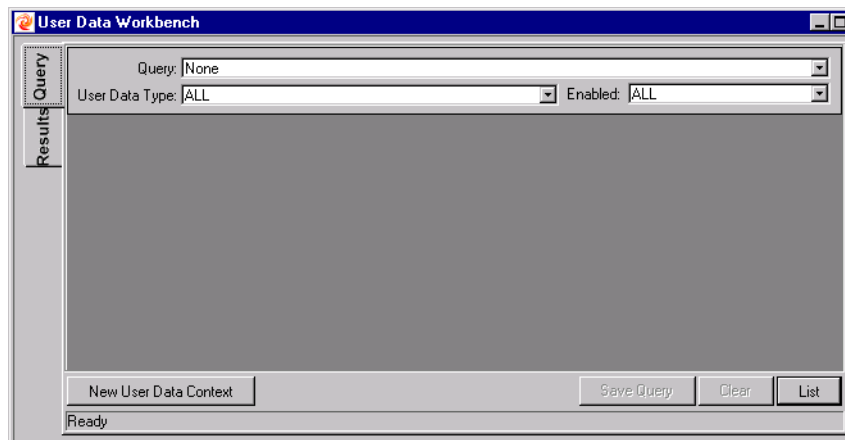
While a Task User Data field can have multiple Roll-Up relationships associated with it, a Project User Data field cannot have more than one Roll-Up relationship defined.

Editing Project/Task User Data Roll-Up

Project/Task User Data Roll-Up can be edited from the Kintana Workbench once it has been created.

To edit a Project/Task User Data Roll-Up relationship:

1. Click the **CONFIGURATION** screen group and click the **USER DATA** screen. The **USER DATA WORKBENCH** opens.



2. Select **PROJECT USER DATA** from the **USER DATA TYPE** drop down list and click **LIST**. The **RESULTS** tab opens with the Project User Data Type loaded.
3. Select the Project User Data and click **OPEN**.

4. Click the **ROLL-UP** tab.

User Data Context : Project User Data

User Data Type: Project User Data

Context Field: Context Value:

Enabled: Yes No Scope: Global

Meta Layer View:

Fields | Layout | Roll-Up

A roll-up defines a computation on a task user data field for all tasks under a project and displays the result in a project user data field. For example, a project can have a Total Cost field that displays the summation of a Cost field for all tasks under that project.

Project User Data Field	Task User Data Field	Roll-Up Method	Enabled
Total Cost	Cost	Sum	Y

New Edit Remove

OK Save Cancel

Ready

5. Select the Roll-Up relationship you wish to edit.
6. Click **EDIT**. The EDIT ROLL-UP window opens.

Add New Roll-Up

Enabled: Yes No

Project User Data Field: Project Cost

Task User Data Field: Cost

Roll-Up Method: Sum

OK Cancel

Ready

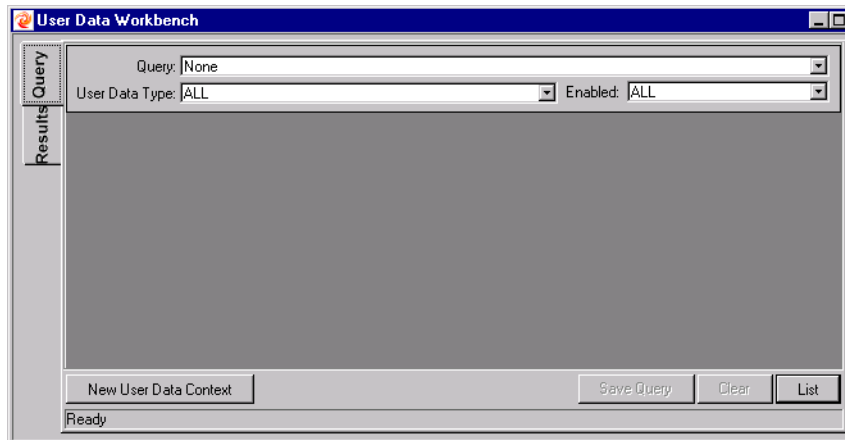
7. Make any desired changes to the Project/Task User Data field or Roll-Up Method.
 8. Click **OK**.
- The Roll-Up definition is updated in the **ROLL-UP** tab.
9. Click **SAVE**.

Deleting Project/Task User Data Roll-Up

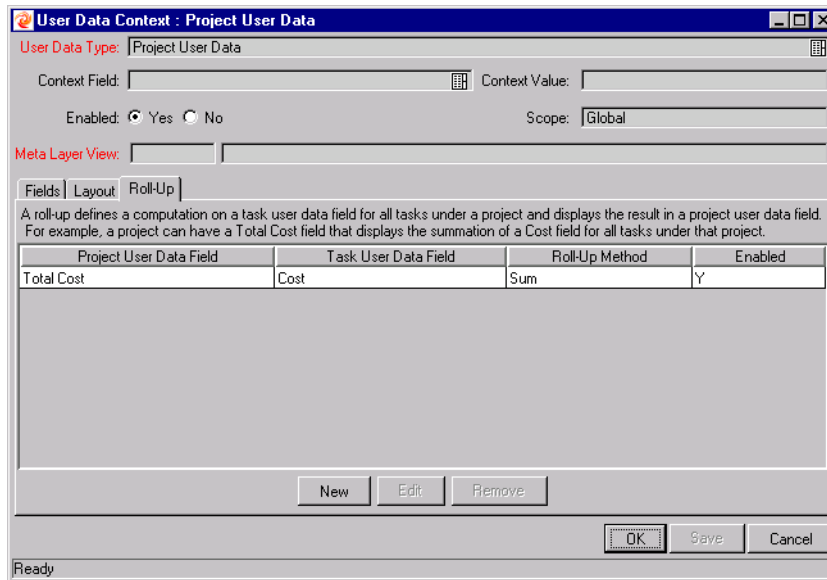
Project/Task User Data Roll-Up can be deleted. This deletion only removes the Roll-Up relationship; it does not delete the referenced User Data fields.

To delete a Project/Task User Data Roll-Up relationship:

1. Click the **CONFIGURATION** screen group and click the **USER DATA** screen. The **USER DATA WORKBENCH** opens.



2. Select **PROJECT USER DATA** from the **USER DATA TYPE** drop down list and click **LIST**. The **RESULTS** tab opens with the Project User Data Type loaded.
3. Select the Project User Data and click **OPEN**.
4. Click the **ROLL-UP** tab.



User Data Context : Project User Data

User Data Type: Project User Data

Context Field: Context Value:

Enabled: Yes No Scope: Global

Meta Layer View:

Fields | Layout | Roll-Up

A roll-up defines a computation on a task user data field for all tasks under a project and displays the result in a project user data field. For example, a project can have a Total Cost field that displays the summation of a Cost field for all tasks under that project.

Project User Data Field	Task User Data Field	Roll-Up Method	Enabled
Total Cost	Cost	Sum	Y

New Edit Remove

OK Save Cancel

Ready

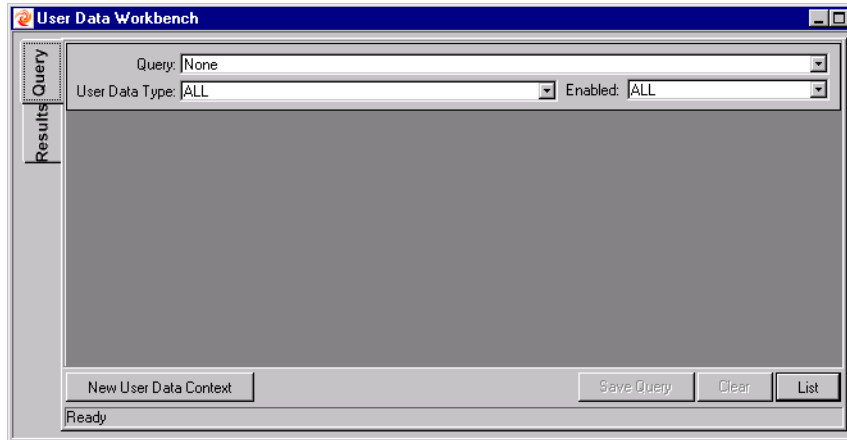
5. Select the Roll-Up relationship you wish to remove.
6. Click **REMOVE**.
7. Click **SAVE**.

Example: Creating and Using Project/Task User Data Roll-Up

Company X wants to capture the total cost of any Project. This value will be calculated as the sum of all Task costs. They also want the calculated cost to be updated every time a Task cost is changed. They will accomplish this using Project/Task User Data Roll-Up.

To create Project/Task User Data Roll-Up that will calculate total Project cost:

1. Click the **CONFIGURATION** screen group and click the **USER DATA** screen. The **USER DATA WORKBENCH** opens.



2. Create the Project User Data field.
 - a. Select **PROJECT USER DATA** from the USER DATA TYPE drop down list.
 - b. Click **LIST**. The **RESULTS** tab opens with the Project User Data Type loaded.
 - c. Open the Project User Data Type.
 - d. Click **NEW** in the **FIELDS** tab. The **FIELD: NEW** window opens.
 - e. Fill in the following information:

Field	Value
FIELD PROMPT	Total Cost
TOKEN	(any useful token)
DESCRIPTION	(any useful description)
VALIDATION	Numeric Text Field
ENABLED	Yes
USER DATA COL	(any available User Data column)
DISPLAY ONLY	Never
DISPLAY	Yes
REQUIRED	Never
WORKBENCH ONLY	No

- f. Click **OK** in the FIELD: NEW window. Click **OK** in the PROJECT USER DATA window to save the new field.
3. Create the Task User Data field.
 - a. In the USER DATA WORKBENCH **QUERY** tab, select **TASK USER DATA** from the USER DATA TYPE drop down list.
 - b. Click **LIST**. The **RESULTS** tab opens with the Task User Data Type loaded.
 - c. Open the Task User Data Type.
 - d. Click **NEW** in the **FIELDS** tab. The FIELD: NEW window opens.
 - e. Fill in the following information:

Field	Value
FIELD PROMPT	Cost
TOKEN	(any useful token)
DESCRIPTION	(any useful description)
VALIDATION	Numeric Text Field
ENABLED	Yes
USER DATA COL	(any available User Data column)
DISPLAY ONLY	Never
DISPLAY	Yes
REQUIRED	Never
WORKBENCH ONLY	No

Field: Cost

Field Prompt: Cost

Token: TASK_COST

Product: All Products

Description: Used to calculate total Project cost

Validation: Numeric Text Field

Component Type: Text Field

Enabled: Yes No

Attributes | Default | Dependencies

User Data Col.: USER_DATA1

Display Only: Never

Display: Yes No

Required: Never

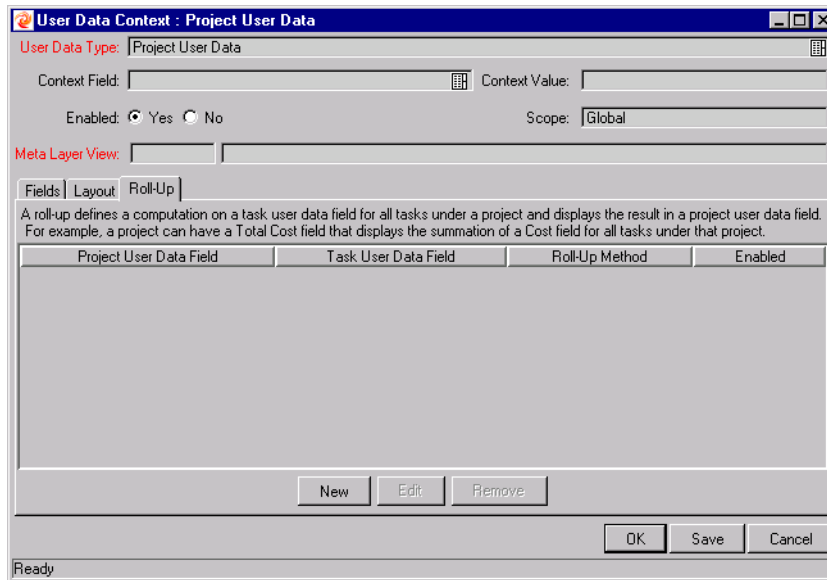
Workbench Only: Yes No

Multi-Select Enabled: Yes No

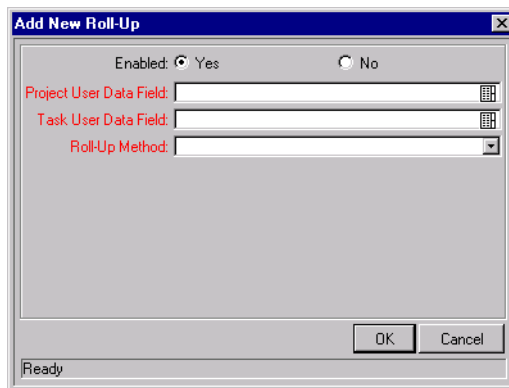
OK Apply Cancel

Ready

- f. Click **OK** in the FIELD: NEW window. Click **OK** in the TASK USER DATA window to save the new field.
4. Create the Roll-Up relationship between the Task and Project User Data fields.
 - a. In the USER DATA WORKBENCH **QUERY** tab, select **PROJECT USER DATA** from the USER DATA TYPE drop down list.
 - b. Click **LIST**. The **RESULTS** tab opens with the Project User Data Type loaded.
 - c. Open the Project User Data Type.
 - d. Click the **ROLL-UP** tab.



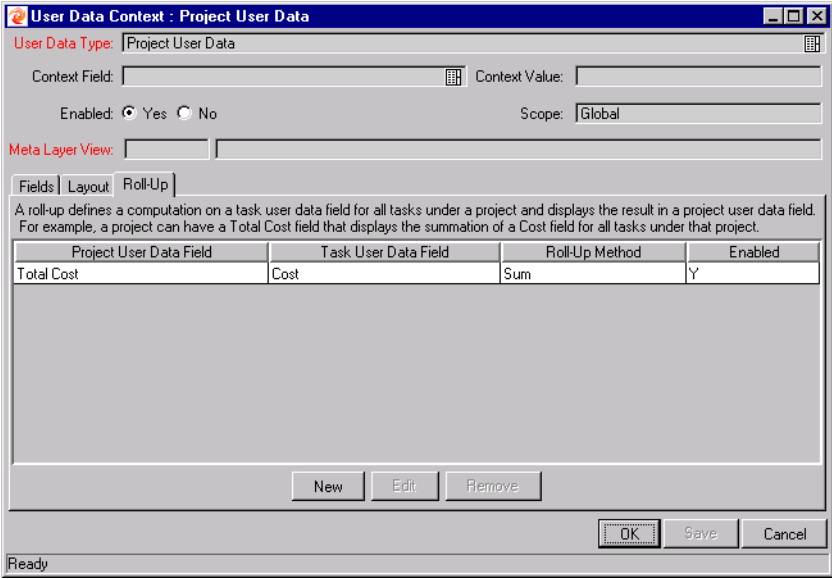
e. Click **NEW**. The ADD NEW ROLL-UP window opens.



- f. Select **TOTAL COST** for the PROJECT USER DATA FIELD.
- g. Select **COST** for the TASK USER DATA FIELD.
- h. Select **SUM** from the ROLL-UP METHOD drop down list.
- i. Click **OK**.

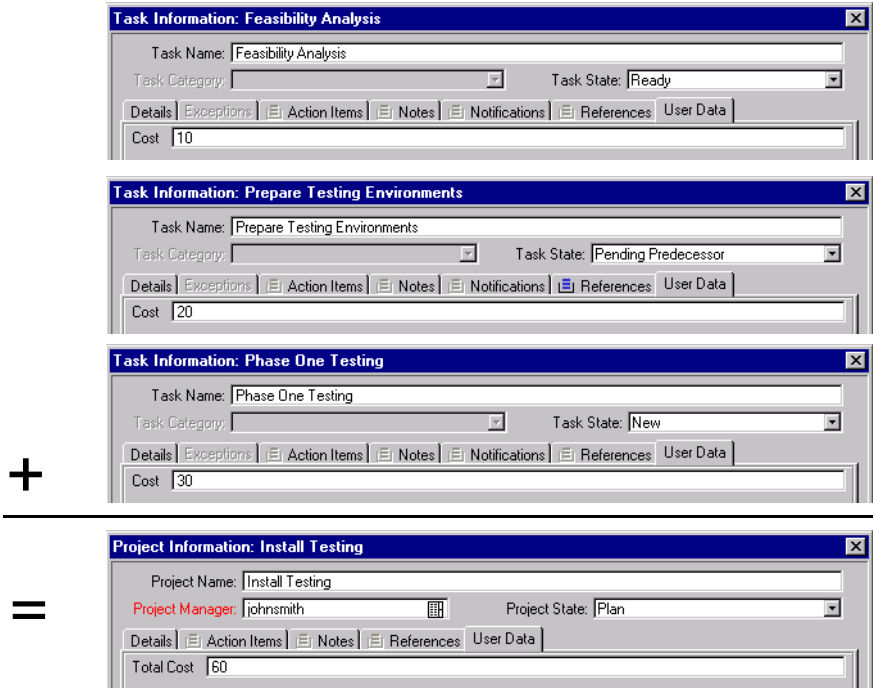
The Roll-Up relationship is added to the **ROLL-UP** tab.

5. Click **SAVE**.



The rolled-up fields can now be accessed from the **USER DATA** tab of the respective Project or Task.

Each Task has its own Cost User Data field. The values for each Task Cost User Data field are rolled up using the **SUM** Roll-Up Method into the Project **TOTAL COST** User Data field.



Referring to User Data

Once a User Data field has been created, it is possible to refer to it from other parts of the Kintana Product Suite by its Token name, preceded by the entity abbreviation and the 'UD' qualifier.



It is possible to define a custom field for the Users entity to store the Department each user is in. This custom field would be defined using the user's User Data and would generate a field with a token of 'DEPARTMENT.' Then, in an Object Type command, a Workflow Step, or in a Report, refer to this new field as the Token [USR.UD.DEPARTMENT]. The Kintana Product Suite would then look into the custom field for the value of this Token. For more information on Tokens and their use, see the Tokens chapter in "[Using Commands and Tokens](#)".

Migrating User Data

Kintana configuration data such as Workflows, Validations, and Request Types can be migrated between instances (installations) of Kintana. User Data values and configurations can also be migrated between instances. The following sections contain more detailed information:

- [Migrating User Data Values](#)
- [Migrating User Data Contexts](#)

Migrating User Data Values

For any particular Kintana configuration entity with User Data fields (Request Type, Object Type, Workflow, etc.) the data in the User Data fields is migrated along with the entity.

- If the two instances have identical User Data configurations, then the User Data will be migrated correctly.
- If the two instances do not have identical User Data configurations, then the User Data will be mapped into the data model according to the storage configuration in the source instance. For this reason, the two instances should be configured with the same User Data fields, or the User Data should be corrected after migration.

- If the User Data is Context Sensitive, then a corresponding Context Sensitive configuration must exist in the destination instance, or the migration will fail.



Note

User Data fields that have different hidden and visible values may be problematic. When the hidden value of a User Data field refers to a primary key (example: Security Group ID) that can be different in the source and destination instances, then the migrator does NOT correct the hidden value. The User Data should be corrected manually after migration.

Migrating User Data Contexts

User Data field contexts can also be migrated between Kintana instances using the Kintana User Data Context Migrator Object Type. This Migrator Object Type can migrate Global as well as Context Sensitive User Data Contexts.

The screenshot shows the 'Add Line' dialog box for the 'Kintana User Data Context Migrator'. The 'Object Type Information' section shows 'Object Type: Kintana User Data Context Migrator', 'Sequence: 1', and 'Application Code: None'. The 'Parameters' section is expanded to 'User Data' and includes the following fields and options:

- Migrator action:
- Preview import?: Yes No
- Kintana source password:
- Kintana dest password:
- User data context:
- Content bundle directory:
- Content bundle filename:
- Replace existing user data context?: Yes No
- Replace existing validations?: Yes No
- Replace existing special cmds?: Yes No

Buttons at the bottom include 'Clear', 'OK', 'Add', and 'Cancel'. A status bar at the bottom indicates: 'Kintana User Data Context Migrator' parameters loaded.

For more detailed information on the Kintana User Data Context Migrator, see "[Kintana Migrators](#)".

Appendix

E

Configuration Worksheets

This appendix provides worksheets that can be printed out and used to capture data required for configuring a deployment system in Kintana. Worksheets are provided for the following entities:

- Workflows
- Workflow Steps
- Object Types and Commands
- Object Type Fields
- Security Groups

For more information on any of the settings of entity parameters, refer to the appropriate Workbench Reference guide.

Information on:	Kintana manual
Workflows and Workflow Steps (screen and field info)	<i>"Configuration Workbench Reference" -- Workflow chapter "Configuration Workbench Reference" -- Validations chapter</i>
Object Types and Object Type Fields	<i>"Deliver Workbench Reference" "Configuration Workbench Reference" -- Validations chapter</i>
Object Type Commands	<i>"Using Commands and Tokens"</i>
Participant and Security	<i>"Kintana Security Model"</i>

Table E-1. Workflow Skeleton

Step No.	Step Name	Description	Type (Execution, Decision, Condition, or Subworkflow)	Transition Values	Validation
1					
2					
3					
4					
5					
6					
7					
8					
9					
10					
11					
12					
13					
14					
15					
16					
17					
18					
19					

Table E-2. Workflow Step [Execution] -- Step Number ____.

	Value
Step Name	
Goal / Result of Step	
Validation*	
Execution Type**	
Processing Type (IMMEDIATE or MANUAL execution?)	
Timeout (Days)	
Source Environment (Group)	
Dest Environment (Group)	
Security (who can act on step):	
<ul style="list-style-type: none"> • Security Group • User Name • Standard Token • User Defined Token 	
Include Notification (Yes/No)	
Notification Event	
Notification Recipient:	
<ul style="list-style-type: none"> • Username • Email Address • Security Group • Standard Token • User Defined Token 	
Notification Message	

Validation Information*	Value
Existing Validation?	
New Validation?	
Validation Type: (<i>text field, auto-complete, drop down list, etc.</i>)	
Validation Definition (list of values or SQL)	

Execution Type**	Value
Built-in Workflow Event:	
<ul style="list-style-type: none"> • Execute Commands • Close • Jump / Receive • Ready for Release • Return from Subworkflow 	
PL/SQL Function	
Token	
SQL Statement	
Workflow step commands	

Table E-3. Workflow Step [Decision] -- Step Number ____.

	Value
Step Name	
Goal / Result of Step	
Validation*	
Decisions Required (Vote on Step's outcome?)	<ul style="list-style-type: none"> • One • At Least One • All
Timeout (Days)	
Security (who can act on step):	
<ul style="list-style-type: none"> • Security Group • User Name • Standard Token • User Defined Token 	
Include Notification (Yes/No)	
Notification Event	
Notification Recipient:	
<ul style="list-style-type: none"> • Username • Email Address • Security Group • Standard Token • User Defined Token 	
Notification Message	

Validation Information*	Value
Existing Validation?	
New Validation?	
Validation Type: (text field, auto-complete, drop down list, etc.)	
Validation Definition (list of values or SQL)	

Table E-4. Workflow Step [Sub-Workflow] -- Step Number ____.

	Value
Step Name	
Goal / Result of Step	
Validation*	
Vote on Step's outcome?	
Timeout (Days)	
Source Environment (Group)	
Dest Environment (Group)	
Security (who can act on step):	
<ul style="list-style-type: none"> • Security Group • User Name • Standard Token • User Defined Token 	
Include Notification (Yes/No)	
Notification Event	
Notification Recipient:	
<ul style="list-style-type: none"> • Username • Email Address • Security Group • Standard Token • User Defined Token 	
Notification Message	

Validation Information*	Value
Existing Validation?	
New Validation?	
Validation Type: (text field, auto-complete, drop down list, etc.)	
Validation Definition (list of values or SQL)	

See “Using Subworkflows” on page 257 for notes on Validations for transitions into and out of Subworkflows.

Table E-5. Object Type Information.

	Value
Object Type Name	
Description	

#	Field Names	Description
1		
2		
3		
4		
5		
6		
7		
8		
9		
10		
11		
12		
13		
14		
15		
16		
17		
18		

Table 11-24. Object Type Commands

Goal of Commands	
Command Steps	
Conditions (When to execute)	

Table E-6. Object Type Field Information.

Field Name	
Validation *	
Field Behavior:	
Attributes (select one):	<ul style="list-style-type: none"> • Display • Editable • Display Only • Required
Default Value	
Dependencies:	
Clear field when	
Display only when	
Required when	

Table E-7. Object Type Field Information.

Field Name	
Validation *	
Field Behavior:	
Attributes (select one):	<ul style="list-style-type: none"> • Display • Editable • Display Only • Required
Default Value	
Dependencies:	
Clear field when	
Display only when	
Required when	

Table 11-25. Field Validation Information

Existing Validation?	
New Validation?	
Validation Type: (text field, auto-complete, drop down list, etc.)	
Validation Definition (list of values or SQL)	

Table 11-26. Field Validation Information

Existing Validation?	
New Validation?	
Validation Type: (text field, auto-complete, drop down list, etc.)	
Validation Definition (list of values or SQL)	

Participant and Security

Table E-8. Security Groups.

Security Group Name	Members	Act on Workflow Steps	Description

Index

A

- Adding Decision Steps 113
 - configure notifications 116
 - general information 113
 - specify security 115
- Adding Execution Steps 117
 - configure notifications 120
 - general information 118
 - specify security 119
- Adding Steps 112
- Adding Subworkflows 121
- Adding Transitions 112, 122
 - all but one value 125
 - all results 126
 - back to step 130
 - based on data 125
 - based on error 127
 - field value 123
 - request workflows 134
 - specific result 123
 - subworkflows 134
 - workflow parameters 131
- Additional Resources
 - Kintana documentation 13
 - Kintana education 16
 - Kintana services 16
 - Kintana support 17
- Advanced Configuration Guides 13
- AND 271
- App Codes 170, 172
 - copying 173

- Application Fields 172
 - App Codes 172
- Archiving Configurations 43
- Auto Complete Validations
 - example 307
 - limiting returned rows 310
 - user-defined multi-select 306
- Auto-Complete Validations 300
 - command with delimited output 301
 - Command With Fixed Width Output 304

B

- Base Paths
 - mass update 191
- Build Object Type
 - creating fields 137
 - modifying layout 150
 - object type commands 156
 - setting name 152
 - setting revision 153
- Build Workflow 79
 - add execution steps 117
 - add subworkflow 121
 - adding decision steps 113
 - adding steps and transitions 112
 - adding transitions 122
 - configure transitions 109

- create step source 81
- overview 79

- Building Object Type 135
 - overview 136
- Business Flow 52
 - example 52

C

- Client Base Path 172
- Client Password 172
- Command Conditions 161
 - examples 161
- Command Requirements 68
 - example 68
- Command Steps 160
- Command Validation 299
- Command with Delimited Output
 - validation 301
- Command With Fixed Width Output
 - validation 304
- Commands 28, 156
 - changing field values 150
 - special commands 28
 - triggering from Workflow 159
- Communication Paths 215
 - dashboard 236
 - reports 241
 - workflow step notifications 216
- Communication Require-

- ments 76
 - example 77
 - Component Type
 - file chooser (static environment override) 314
 - file chooser (token-based environment override) 315
 - Component Types 288
 - directory chooser 313
 - file chooser 313
 - multi-select auto-complete 306
 - Configuration Security 210
 - ownership for entities 211
 - removing access grants 213
 - Configuration Worksheets 391
 - object type field worksheet 397
 - object type worksheet 396
 - security worksheet 398
 - subworkflow step 395
 - workflow 392
 - workflow decision step 394
 - workflow execution step 393
 - Configuring Dashboard 236
 - controlling portlet access 237
 - custom portlets 240
 - default dashboard 240
 - Configuring Field Behavior 144
 - Configuring Field Dependencies 148
 - Context Field
 - changing 365
 - defining 363
 - Context Sensitive
 - defining fields 365
 - editing fields 367
 - Context Sensitive User Data
 - changing context field 365
 - changing context value 366
 - copying 367
 - defining context field 363
 - defining context value 364
 - defining fields 365
 - deleting 367
 - editing 365
 - editing fields 367
 - example for Request Header Type 368
 - generating 363
 - Context Value
 - changing 366
 - defining 364
 - Controlling Portlet Access 237
 - disabling portlets 237
 - restricting access 238
 - Copy From 154, 173
 - Copy Function
 - fields in Object Type 154
 - Copying Environment Groups 183
 - Copying Environments 167
 - Creating Object Type Fields 137
 - Custom Portlets 240
- D**
- DB
 - Password 172
 - username 172
 - Decision Step Source
 - creating 85
 - general information 86
 - timeout 89
 - validation 88
 - voting requirements 88
 - Default Dashboard 240
 - Define Executions 94
 - close package failure 99
 - close package success 97
 - evaluate token 104
 - execute object type commands 95
 - jump to request workflow 101
 - PL/SQL function 103
 - receive from request workflow 101
 - set ready for release 101
 - SQL statement 104
 - subworkflow return 102
 - system level commands 106
 - Defining Environment Groups 176
 - adding environments 184
 - copying 183
 - linking to Workflows 188
 - removing environments 186
 - setting execution order 187
 - setting ownership 180
 - setting user access 181
 - using app codes 189
 - Defining Environments 163, 164
 - app codes 170
 - connection protocol 167
 - copying app codes 173
 - linking to Workflows 188
 - requirements 163

- setting user access 174
 - transfer protocol 168
- Deployment 20
- Deployment System
 - archiving configurations 43
 - build workflow 79
 - building object type 135
 - business flow 52
 - command requirements 68
 - communication paths 215
 - communication visibility requirements 76
 - configuration security 210
 - configuring reports 241
 - dashboard 236
 - defining environments 163
 - defining environments overview 164
 - developing 37
 - environment groups 176
 - environment maintenance 190
 - environment requirements 69
 - environments and workflows 188
 - example 46
 - gathering requirements 49
 - integrate participants 195
 - migrate finished 254
 - migrating configurations 40
 - object name 152
 - object requirements 64
 - object revision 153
 - object type commands 156
 - overview 45
 - package creation security 202
 - package processing security 206
 - participant overview 195
 - participant security 197
 - participants and security 71
 - release management 62
 - required workflow settings 80
 - rollout checklists 243
 - security 195
 - security groups 197
 - security overview 195
 - step information requirements 59
 - subworkflows 61
 - technical flow requirements 54
 - transitions 109
 - using multiple instances 38
 - validation execution relationship 111
 - workflow overview 79
 - workflow step notifications 216
 - worksheets 391
- Deployment System Requirements 49
 - process 50
- Directory Chooser 313
- Documentation 13
- Dynamic List Validations 297
 - command 299
 - SQL 297
- E**
- Environment
 - setting user access 174
- Environment Groups 27
 - adding Environments to 184
 - copying 183
 - defining 176
 - removing Environments from 186
 - setting ownership 180
 - setting the execution order 187
 - setting user access 181
 - using App Codes 189
 - when to use 176
- Environment Maintenance
 - mass update base paths 191
 - password management 192
 - testing setup 190
- Environment Password
 - updating with Kintana 192
- Environment Requirements 69, 163
 - example 70
- Environments 27
 - adding to environment groups 184
 - app codes 170
 - copying 167
 - copying app codes 173
 - defining 163
 - deleting 194
 - maintenance 190
 - mass update of base paths 191
 - password management 192
 - removing from environment group 186
 - selecting connection protocol 167

- selecting transfer protocol 168
- execute_object_commands 93
- Execution Step Source
 - close package failure 99
 - close package success 97
 - creating 90
 - define executions 94
 - evaluate token 104
 - execute object type commands 95
 - general information 91
 - jump to request workflow 101
 - PL/SQL function 103
 - receive from request workflow 101
 - select validation 109
 - set ready for release 101
 - SQL statement 104
 - subworkflow return 102
 - system level commands 106
 - timeouts 109
- Executions
 - and validations 111
- F**
- Field Dependencies
 - configuring for Object Type 148
- Field Window
 - attributes tab 146
 - default tab 147
 - dependencies tab 147
- Fields
 - changing column width 150
 - editing in Object Types 155
- moving 151
- preview layout 152
- removing in Object Types 155
- File Chooser 313
 - static environment override 314
 - token-based environment override 315
- First Line 273
- G**
- Gathering Requirements 49
- Gathering Step Information 59
 - example 60
- I**
- Instance Requirements 42
- Integrating Participants 195
- J**
- Jump Step
 - generating 266
- Jump/Receive Step
 - pairing in Workflows 269
- Jump/Receive Step Label
 - validation 264
- Jump/Receive Step Labels 263
- K**
- Key Concepts 19
- Kintana Interface 19
- Kintana Product Integration 32
 - accelerators 34
 - create 34
 - dashboard 33
 - drive 33
 - projects 33
 - requests 34
 - solutions 32
- Kintana Workbench 19
- L**
- Last Line 275
- Loop counter
 - example in Workflow 281
- M**
- Mass Update
 - base paths 191
- Migrating Configurations 40
- Migrators 37
 - instance requirements 42
 - overview 40
 - using 41
- Modifying Active Workflows 276
 - copy and test 277
 - disabling step 278
 - execution steps 279
 - move requests 277
 - redirecting 278
 - security/performance considerations 279
 - verifying workflow logic 280
- Modifying Object Type Lay-

- out 150
- Multiple Instances 38
 - new instance 39
 - single PROD instance 39
- Multi-Select Auto-Complete
 - user-defined 306

N

- New App 172
- Notification Message 231
 - HTML tokens 233
 - smart URLs 234
 - smart URLs in HTML 235
 - tokens 233
- Notification Recipients 229
- Notifications
 - tokens in message 233

O

- Object Name 152
- Object Requirements 64
 - example 65
- Object Revision 153
- Object Type Commands 156
 - and workflow 159
 - command conditions 161
 - command steps 160
 - examples 162
 - interface 156
 - overview 156
 - special commands 159
- Object Type Field Worksheet 397
- Object Type Fields
 - available types 139
 - building validation 142
 - changing values with commands 150
 - configuring behavior 144
 - copying 154

- creating 137
- dependencies 148
- field window attributes tab 146
- field window default tab 147
- field window dependencies tab 147
- selecting validation 139, 141
- text area 151

Object Type Worksheet 396

Object Types 20

- command requirements 68
- commands and Workflow 159
- editing fields 155
- removing fields 155
- workflow integration 25

OR 272

Ownership

- setting for environment groups 180

P

Package 22

- package lines 23

Package Creation Security 202

- enabling 203
- object type restriction 205
- workflow restriction 204

Package Lines 23

Package Processing Security 206

- general access 206
- participant restriction 209
- workflow security 207

Package Workflow

- integration 262

PACKAGE_NO_LINK 235

PACKAGE_URL 235

Parameters

- copy from 354
- Workflow 280

Participant Requirements 71

Participant Security 197

Participants 31

Passwords
 updating with command prompt 193

Process Requirements 50
 business flow 52
 release management 62
 step information 59
 subworkflow considerations 61
 technical flow 54
 workflow considerations 50

Projects 375

R

Receive Step
 generating 267

Release Management 62
 example 63

Remove App 172

Reports 35

Request Workflow
 integration 262

REQUEST_URL 235

Requests
 setting reopen workflow step 275

required settings 80

Required Workflow Settings 80

Requirements
 deployment process 50

rm_ready_for_release 93

Rollout
 additional resources 256
 educate users 255
 enable entities 255
 enable user access 255
 migrate deployment system 254

Rollout Checklists 243
 cross entity 253
 dashboard 252
 environment 250
 general 244
 object type 249

security 251
workflow 246

S

Security

- general package access 206
- object type restriction 205
- package creation 202
- package creation enabling 203
- package processing 206
- package workflow processing 207
- participant package restriction 209
- workflow restriction 204

Security Groups 30

- establishing 197
- specifying users 198
- using resource management 201

Security Requirements 71

- example 73

Security Worksheet 398

Server Base Path 172

Special Commands 28, 159

SQL Validations 297

- tips 299

Static List Validations 295

Step Sources

- creating 81
- creating decision step 85
- creating execution step 90
- creation overview 82
- restrictions 85

Subworkflow Considerations 61

- example 62

Subworkflow Step Worksheet 395

Subworkflows

- overview 257
- transitioning out of 260
- transitioning to 258

Swap Mode 151

SYNC 272

T

Table Component Validations 316

- adding to request type 327
- column totals 325
- creating rules 320
- defining 317
- rules example 321
- tokens 325

Tasks

- user data roll-up 375

Technical Flow 54

- example 55

Text Area 151, 360

Token Evaluation

- example 307

Tokens 29

- types 349

Transitions 109

- adding 122

U

URL to Validation 294

User Access 174

- setting for environment groups 181

User Data

- adding fields 353
- changing field width 360
- configuring field dependencies 356
- context sensitive in validation 291
- copying fields 354
- creating project-task roll-up 375
- deleting project-task roll-up 381
- editing fields 355
- editing project-task roll-up 379
- migrating 388
- migrating contexts 389
- migrating values 388
- moving fields 360
- overview 351
- preview layout 361

- project task roll-up 375
 - referring to 388
 - removing fields 358
 - roll-up example 382
 - supported functionality 351
 - text area 360
 - using in the Kintana product suite 388
- user data roll-up 375
- Using Migrators 37

V

- Vaildations
- and executions 111
- Validations 29, 109, 139
- auto-complete 300
 - auto-complete vs drop down 143
 - available field types 139
 - building 142
 - Command 299
 - Command With Delimited Output 301
 - Command With Fixed Width Output 304
 - configuration tips 144
 - context sensitive user data and 291
 - creating 291
 - defined 288
 - deleting 295
 - directory chooser 313
 - dynamic list 297
 - editing 293
 - file chooser 313
 - file chooser (static environment override) 314
 - file chooser (token-based environment override) 315
 - overview 288
 - package and request group 329
 - quick link 294
 - request type category 330
 - seeded 331
 - special characters and 331
 - SQL 297
 - SQL tips 299
 - static lists 295

- system 331
- table component 316
- text area 1800 316

Visibility Requirements 76

Voting

- All 89
- At Least One 88
- One 88

W

wf_close_failure 93

wf_close_success 93

wf_jump 93, 263

wf_receive 93, 263

wf_return 93

WORKBENCH_PACKAGE_URL 235

Workflow 24, 80

- and object type commands 159

- condition steps 271

- jump/receive 262

- modifying while in use 276

- object type integration 25

- package request integration 262

- setting reopen step for requests 275

- step notifications 216

Workflow Considerations 50

- decisions vs executions 51

- immediate vs manual 51

- timeouts 52

Workflow Decision Step Worksheet 394

Workflow Execution Step Worksheet 393

Workflow Integration 262

- jump step source 266

- jump/receive pair 269

- jump/receive step label 264

- receive step source 267

Workflow Logical Rules 247

Workflow Parameters 280

- example 281

- generating 280

- Workflow Step Notifications 216
 - based on field value 228
 - configuring intervals 225
 - configuring message 231
 - configuring recipients 229
 - eligible 218
 - multiple notifications 223
 - overview 216
 - reminders 227
 - specific error 221
 - specific result 219
 - specifying time 224
 - when to send 218
- Workflow Steps
 - conditions 271
- Workflow Worksheet 392
- Workflows
 - linking to environment groups 188
 - linking to environments 188