

---

# PeopleSoft 9.1 PeopleBook: Approval Framework

---

**May 2012**

Copyright © 2001, 2012, Oracle and/or its affiliates. All rights reserved.

### **Trademark Notice**

Oracle and Java are registered trademarks of Oracle and/or its affiliates. Other names may be trademarks of their respective owners.

### **License Restrictions Warranty/Consequential Damages Disclaimer**

This software and related documentation are provided under a license agreement containing restrictions on use and disclosure and are protected by intellectual property laws. Except as expressly permitted in your license agreement or allowed by law, you may not use, copy, reproduce, translate, broadcast, modify, license, transmit, distribute, exhibit, perform, publish or display any part, in any form, or by any means. Reverse engineering, disassembly, or decompilation of this software, unless required by law for interoperability, is prohibited.

### **Warranty Disclaimer**

The information contained herein is subject to change without notice and is not warranted to be error-free. If you find any errors, please report them to us in writing.

### **Restricted Rights Notice**

If this is software or related documentation that is delivered to the U.S. Government or anyone licensing it on behalf of the U.S. Government, the following notice is applicable:

U.S. GOVERNMENT END USERS: Oracle programs, including any operating system, integrated software, any programs installed on the hardware, and/or documentation, delivered to U.S. Government end users are "commercial computer software" pursuant to the applicable Federal Acquisition Regulation and agency-specific supplemental regulations. As such, use, duplication, disclosure, modification, and adaptation of the programs, including any operating system, integrated software, any programs installed on the hardware, and/or documentation, shall be subject to license terms and license restrictions applicable to the programs. No other rights are granted to the U.S. Government.

### **Hazardous Applications Notice**

This software or hardware is developed for general use in a variety of information management applications. It is not developed or intended for use in any inherently dangerous applications, including applications that may create a risk of personal injury. If you use this software or hardware in dangerous applications, then you shall be responsible to take all appropriate failsafe, backup, redundancy, and other measures to ensure its safe use. Oracle Corporation and its affiliates disclaim any liability for any damages caused by use of this software or hardware in dangerous applications.

### **Third Party Content, Products, and Services Disclaimer**

This software or hardware and documentation may provide access to or information on content, products and services from third parties. Oracle Corporation and its affiliates are not responsible for and expressly disclaim all warranties of any kind with respect to third-party content, products, and services. Oracle Corporation and its affiliates will not be responsible for any loss, costs, or damages incurred due to your access to or use of third-party content, products, or services.

# Contents

## Preface

<b>Approval Framework Preface .....</b>	<b>xi</b>
PeopleSoft Approval Framework .....	xi
PeopleBooks and the PeopleSoft Online Library .....	xi
Common Elements Used in This PeopleBook .....	xii

## Chapter 1

<b>Getting Started with Approval Framework .....</b>	<b>1</b>
Approval Framework Overview .....	1
Approval Framework Implementation .....	1
Configuring Approval Processes .....	1
Creating New Approval Processes .....	2

## Chapter 2

<b>Understanding the Approval Framework .....</b>	<b>5</b>
Understanding the Approval Framework Feature .....	5
Understanding the Approval Framework Process Flow .....	6
Understanding Transaction Approval Flows .....	7
Understanding Header- and Line-Level Approvals .....	9
Understanding Criteria for Approval Framework Processes .....	10
Understanding Approval Features .....	10
Understanding Tasks in the Approval Framework .....	14

## Chapter 3

<b>Setting Up Approval Framework Process Definitions .....</b>	<b>17</b>
Defining the Setup Process Definitions Component .....	17
Pages Used to Define Approval Framework Processes .....	17
Defining Approval Framework Processes .....	18
Defining Criteria for Approval Framework Processes .....	23
Defining Paths for Approval Framework Processes .....	30

Defining Steps for Approval Framework Processes .....	32
 <b>Chapter 4</b>	
<b>Defining the Approval Transaction Registry .....</b>	<b>37</b>
Understanding the Approval Transaction Registry .....	37
Prerequisites .....	37
Setting Up the Transaction Registry .....	38
Page Used to Set Up the Transaction Registry .....	38
Registering Approval Transactions .....	38
Configuring Approval Transactions .....	43
Page Used to Configure Approval Transactions .....	44
Configuring Approval Transactions .....	44
 <b>Chapter 5</b>	
<b>Defining Notification Templates and Users for Approval Framework .....</b>	<b>51</b>
Defining Notification Templates for Approval Framework .....	51
Pages Used to Define Notification Templates for Approval Framework .....	51
Entering Generic Template Definitions .....	51
Defining Users for Approval Framework .....	52
Pages Used to Define Users for Approval Framework .....	53
Attaching Workflow Roles to Users .....	53
Defining Workflow for User Profiles .....	54
Defining User Lists .....	56
 <b>Chapter 6</b>	
<b>Defining Dynamic Approvals .....</b>	<b>59</b>
Understanding Dynamic Paths .....	59
Understanding Dynamic Approval Authorizations .....	60
Understanding Approval Authorizations .....	62
Defining Dynamic Approvals .....	63
Pages Used to Define Dynamic Approvals .....	63
Defining User Lists for Dynamic Authorizations .....	63
Setting Up Approval Authorizations .....	64
Defining Dynamic Approval Paths .....	65

## Chapter 7

<b>Using Email Collaboration .....</b>	<b>67</b>
Understanding Email Collaboration .....	67
Setting Up the PSFT_EMG_GETMAIL Node .....	70
Pages used to Set Up the PSFT_EMG_GETMAIL Node .....	70
Setting Up the PSFT_EMG_GETMAIL Node .....	71
Defining Message and Service Operation .....	71
Pages Used to Define Message and Service Operation .....	71
Defining Integration Broker Message .....	72
Defining Service .....	73
Defining Service Operation .....	73
Defining and Mapping EMC Forms .....	74
EMC Forms .....	74
Pages Used to Define and Map EMC Forms .....	74
Defining EMC forms .....	75
Defining EMC Layouts .....	76
Defining Field Mapping .....	76
Triggering Email Collaboration .....	77
Pages Used to Trigger Email Collaboration .....	77
Updating Approval Transaction Registry to Send Email Approvals .....	77
Configuring Transactions for Email Approval .....	79
Scheduling the Application Engine Program EOAWEMC .....	79
Adding or Modifying Email Addresses for Users .....	79

## Chapter 8

<b>Using EMC Classes .....</b>	<b>81</b>
Understanding EMC Classes .....	81
EmailFormManager Class .....	81
EmailFormManager Class Methods .....	81
addRecipient .....	81
addCC .....	82
addBCC .....	82
addAttachment .....	83
sendEmails .....	84
EmailFormManager Class Properties .....	84
inlineText .....	84
attachmentText .....	84
subject .....	85
submitMessage .....	85

from .....	85
fromEmail .....	85
replyTo .....	85
prependText .....	86
appendText .....	86
deliveryMethod .....	86
EOAW_EMCM:utils Class .....	86
EOAW_EMCM:utils Class Methods .....	86
getAppRS .....	86
getErrorCodesRS .....	87
getPromptsRS .....	88
getRowFromPath .....	88

## Chapter 9

<b>Using the Notification and Escalation Manager .....</b>	<b>91</b>
Understanding Notification and Escalation Manager .....	91
Pages Used to Set Up the Notification and Escalation Manager .....	91
Associating Events to a Server .....	92
Setting Up an Escalation Event .....	92
Setting Up a NEM .....	93

## Chapter 10

<b>Using the Approval Monitor .....</b>	<b>95</b>
Understanding the Approval Monitor .....	95
Understanding Approval Reassignment .....	96
Configuring the Approval Monitor .....	96
Page Used to Configure the Approval Monitor .....	96
Configuring the Approval Monitor .....	96
Using the Approval Monitor .....	98
Pages Used to Use the Approval Monitor .....	98
Using the Approval Monitor Search Page .....	99
Viewing Search Results .....	100
Utilizing the Approval Monitor for a Specific Approval Process .....	101
Using the User Monitor .....	105
Page Used to Use the User Monitor .....	106
Using the User Monitor .....	106

## Chapter 11

<b>Using Approval Framework Base Classes .....</b>	<b>107</b>
Understanding Approval Framework Base Classes .....	107
LaunchManager Class .....	107
LaunchManager Class Methods .....	108
LaunchManager .....	108
DoSubmit .....	109
DoReSubmit .....	109
DoRestart .....	110
PrepareToSubmit .....	110
SetHeader .....	111
Reset .....	112
FindDefinitionID .....	113
TerminateRunningProcess .....	113
LaunchManager Class Properties .....	114
hasTxn .....	114
definition .....	114
hasAppDef .....	114
hasAppInst .....	114
hasEndedAppInst .....	114
EOAW_CORE:DEFN:AppDef appDef .....	115
resubmitEnabled .....	115
submitEnabled .....	115
restartEnabled .....	116
monitorEnabled .....	116
previewEnabled .....	116
EOAW_CORE:ENGINE:AppInst appInst .....	116
EOAW_CORE:DEFN:AWTxn txn .....	116
requester .....	117
Approval Manager Class .....	117
ApprovalManager Class Methods .....	117
ApprovalManager .....	117
DoApprove .....	118
DoApproveRowSet .....	118
DoReassign .....	119
DoReassignAll .....	120
DoDeny .....	120
DoDenyRowset .....	121
DoDenyWithAllowUndeny .....	121
DoHardDeny .....	122
DoLineResubmit .....	123

DoAddNewLine .....	123
GetPending .....	124
DoPushback .....	125
AddComments .....	125
TakeNoAction .....	126
PutOnHold .....	127
PutOnHoldCount .....	127
GetPushedBack .....	128
GetPertinentThreads .....	128
GetStage .....	129
GetPendingSteps .....	129
DoLineTerminate .....	130
GetParticipant .....	130
GetAllActiveParticipants .....	131
RequestInformation .....	131
SetAttributeObject .....	132
ApprovalManager Class Properties .....	132
hasAppInst .....	132
definition .....	133
level .....	133
hasPending .....	133
pushbackEnabled .....	133
EOAW_CORE:ENGINE:AppInst the_inst .....	133
isReviewer .....	134
Approval Event Handler Class .....	134
ApprovalEventHandler Class Methods .....	134
ApprovalEventHandler .....	134
OnProcessLaunch .....	134
OnStepActivate .....	135
OnStepHold .....	136
OnStepReassign .....	136
OnStepComplete .....	137
OnStepPushback .....	137
OnStepReactivate .....	138
OnFinalHeaderDeny .....	138
OnHeaderDeny .....	139
OnHeaderApprove .....	139
OnNoApprovalNecessary .....	139
OnLineDeny .....	140
OnLineApprove .....	140
OnAllLinesProcessed .....	141
OnTerminate .....	141
OnError .....	142
OnStepReview .....	142
OnTimeout .....	142



OnAdHocInsert .....	143
OnAdHocDelete .....	143
OnUserLocked .....	144
OnStepRequestInformation .....	144
OnRequestInformationAdded .....	145
ProcessNotifications .....	145
OnLineTerminate .....	146
ApprovalEventHandler Class Properties .....	146
wlPrefix .....	146
wlBusinessProc .....	146
wlActivityName .....	147
<b>Index .....</b>	<b>149</b>



# Approval Framework Preface

This preface provides an overview of the Approval Framework PeopleBook.

---

## PeopleSoft Approval Framework

Many daily tasks are part of a larger process that involves several steps and people working together. The term *workflow* refers to this process, which could encompass, for example, the approval of a purchase requisition or a job change request form. To facilitate this type of multiuser process, the PeopleSoft product can automatically trigger workflow notifications to inform the next approver in the process of work awaiting them.

The PeopleSoft Approval Framework is a specialized application used for defining and configuring approvals. Approval Framework enables you to configure approval processes using existing components without writing code.

This PeopleBook provides you with information to customize the approval processing of components that leverage the Approval Framework.

Many PeopleSoft applications are delivered with predefined approval processes. However, additional, essential information describing the setup and design of your system resides in companion documentation. The companion documentation consists of important topics that apply to many or all Oracle's PeopleSoft applications across the PeopleSoft product lines. You should be familiar with the contents of the PeopleBooks that apply to your organization contained within the following PeopleSoft product lines:

- *PeopleSoft Financials and Supply Chain Management (FSCM) PeopleBooks*
- *PeopleSoft Human Capital Management (HCM) PeopleBooks*
- *PeopleSoft Customer Relationship Management (CRM) PeopleBooks*
- *PeopleSoft Enterprise Performance Management (EPM) PeopleBooks*

---

## PeopleBooks and the PeopleSoft Online Library

A companion PeopleBook called *PeopleBooks and the PeopleSoft Online Library* contains general information, including:

- Understanding the PeopleSoft online library and related documentation.
- How to send PeopleSoft documentation comments and suggestions to Oracle.
- How to access hosted PeopleBooks, downloadable HTML PeopleBooks, and downloadable PDF PeopleBooks as well as documentation updates.
- Understanding PeopleBook structure.
- Typographical conventions and visual cues used in PeopleBooks.

- ISO country codes and currency codes.
- PeopleBooks that are common across multiple applications.
- Common elements used in PeopleBooks.
- Navigating the PeopleBooks interface and searching the PeopleSoft online library.
- Displaying and printing screen shots and graphics in PeopleBooks.
- How to manage the locally installed PeopleSoft online library, including web site folders.
- Understanding documentation integration and how to integrate customized documentation into the library.
- Application abbreviations found in application fields.

You can find *PeopleBooks and the PeopleSoft Online Library* in the online PeopleBooks Library for your PeopleTools release.

---

## Common Elements Used in This PeopleBook

<b>Business Unit</b>	An identification code that represents a high-level organization of business information. You can use a business unit to define regional or departmental units within a larger organization.
<b>Description</b>	Free-flow text up to 30 characters.
<b>Effective Date</b>	Date on which a table row becomes effective; the date that an action begins. For example, if you want to close out a ledger on June 30, the effective date for the ledger closing would be July 1. This date also determines when you can view and change the information. Pages or panels and background processes that use the information use the current row.
<b>Language or Language Code</b>	The language in which you want the field labels and report headings of the reports to print. The field values appear as you enter them.  Language also refers to the language spoken by an employee, applicant, or nonemployee.
<b>Process Frequency</b> (group box)	Designates the appropriate frequency in the Process Frequency group box:  Process Once executes the request the next time the batch process runs. After the batch process runs, the process frequency is automatically set to Don't Run.  Always Process executes the request every time the batch process runs.  Don't Run ignores the request when the batch process runs.
<b>Process Monitor</b>	This link takes you to the Process List page, where you can view the status of submitted process requests.
<b>Report ID</b>	The report identifier.

<b>Report Manager</b>	This link takes you to the Report List page, where you can view report content, check the status of a report, and see content detail messages (which show you a description of the report and the distribution list).
<b>Request ID</b>	A request identification that represents a set of selection criteria for a report or process.
<b>Run</b>	This button takes you to the Process Scheduler request page, where you can specify the location where a process or job runs and the process output format.
<b>SetID</b>	An identification code that represents a set of control table information or TableSets. A TableSet is a group of tables (records) necessary to define your organization's structure and processing options.
<b>Short Description</b>	Free-flow text up to 15 characters.
<b>Status</b>	the options in this field are <i>Active</i> or <i>Inactive</i> . By linking status and effective date, you can retain historical information and plan future implementation. For auditing purposes, PeopleSoft encourages inactivating data that is no longer in use instead of deleting it.
<b>User ID</b>	The system identifier for the individual who generates a transaction.



## Chapter 1

# Getting Started with Approval Framework

This chapter provides an overview of the PeopleSoft Approval Framework and discusses implementation information.

---

## Approval Framework Overview

Use this PeopleBook for information about the Approval Framework, including:

- Setting up Approval Framework process definitions.
- Defining the approval transaction registry.
- Defining notification templates and users for Approval Framework.
- Defining dynamic approvals.
- Setting up email collaboration.
- Setting up the notification and escalation manager.
- Configuring and use Approval Framework Monitor.
- Using Approval Framework PeopleCode classes.

---

## Approval Framework Implementation

Many PeopleSoft applications are delivered with predefined Approval Framework processes. In order to configure and use predefined processes you will:

### Configuring Approval Processes

Use the following steps to configure predefined approval processes for your application:

<i>Step</i>	<i>Reference</i>
Activate workflow	Refer to the appropriate PeopleBook for your product implementation.

<b>Step</b>	<b>Reference</b>
Define user lists	See and <a href="#">Chapter 5, "Defining Notification Templates and Users for Approval Framework," Defining User Lists, page 56.</a>
Set up approval framework	See and <a href="#">Chapter 3, "Setting Up Approval Framework Process Definitions," Defining the Setup Process Definitions Component, page 17.</a>
Configure the approval user monitor	See and <a href="#">Chapter 10, "Using the Approval Monitor," Configuring the Approval Monitor, page 96.</a>

## Creating New Approval Processes

The approval framework can be used to create new approval processes. Application developers can set up workflow approvals using a transaction-definition component. The following steps are used to create a new approval process.

<b>Step</b>	<b>Reference</b>
Identify the transaction entry component	Identify the component that will be used for transaction entry.
Identify an approval component	Identify or create a component that will be used for transaction approval. This is usually a new component, but can be the same as the transaction component. The approval component will include the necessary push buttons to approve or deny transactions, as well as any other actions that can be taken.
Create an approval cross reference table	The cross reference table must include the subrecord EOAW_XREF_SBR.  See and <a href="#">Chapter 4, "Defining the Approval Transaction Registry," Setting Up the Transaction Registry, page 38.</a>
Create a view of all users that can be an approval participant	The view is used by the approval monitor to display the person's name and contact information.
Develop an approval transaction handler class	Define an application class used to monitor events for the transaction. The application class must extend the ApprovalEventHandler class and enable applications to receive notifications.  See and <a href="#">Chapter 11, "Using Approval Framework Base Classes," ApprovalEventHandler Class Methods, page 134.</a>



<b>Step</b>	<b>Reference</b>
Create code to launch approvals	Create the code to launch Approval Framework. The code must extend the LaunchManager application class and at a minimum define a submit button.  See and Chapter 11, "Using Approval Framework Base Classes," LaunchManager Class Methods, page 108.
Create code to manage approvals	Create the code to manage the approvals using the ApprovalManager application class.  See and Chapter 11, "Using Approval Framework Base Classes," ApprovalManager Class Methods, page 117.
Define notifications	See and Chapter 5, "Defining Notification Templates and Users for Approval Framework," Defining Notification Templates for Approval Framework, page 51.
Create the approvals transaction registry	See and Chapter 4, "Defining the Approval Transaction Registry," page 37.
Configure the approval transactions	See and Chapter 4, "Defining the Approval Transaction Registry," Configuring Approval Transactions, page 43.
Define user lists	See and Chapter 5, "Defining Notification Templates and Users for Approval Framework," Defining User Lists, page 56.
Set up approval framework	See and Chapter 3, "Setting Up Approval Framework Process Definitions," Defining the Setup Process Definitions Component, page 17.
Configure the approval user monitor	See and Chapter 10, "Using the Approval Monitor," Configuring the Approval Monitor, page 96.



## Chapter 2

# Understanding the Approval Framework

This chapter provides an overview of the Approval Framework feature and:

- The Approval Framework process flow.
- Header- and line-level approvals.
- Transaction approval flows.
- Approval features.
- Tasks in the Approval Framework.

---

## Understanding the Approval Framework Feature

Many daily tasks are part of a larger process that involves several steps and people working together. The term *workflow* refers to this process, which could encompass, for example, the approval of a purchase requisition or a job change request form. To facilitate this type of multiuser process, the PeopleSoft product can automatically trigger workflow notifications to inform the next approver in the process of work awaiting them.

A specialized designer is available to address workflow approvals. This designer enables you to configure Approval Frameworks using existing components without writing code.

Three types of users come together to set up approvals. They include application developers, functional business analysts, and users, who include requesters, approvers, and reviewers. Approval Framework brings these roles together to define an approval process workflow.

Application developers set up Approval Framework using a transaction-definition component. Examples of a transaction might include a job change request or leave of absence request. Transactions are made up of an approval process, routing rules and steps, and a set of users who approve and review the transaction.

Using Approval Framework, you can:

- Approve or deny individual line items in a transaction.
- Approve and deny multiple transactions one at a time.
- Include multiple approvers for individual steps.
- Assign additional approvers and reviewers during the approval process.
- Escalate approvals.
- Approve, deny, or push back approvals.

- Reassign approval tasks to another approver.
- Use worklist and email notifications.

---

## Understanding the Approval Framework Process Flow

Approval Frameworks are triggered when requesters originate a transaction, such as a purchase requisition or a job change request, and a set of approvers carry out tasks related to the transaction. The PeopleSoft Approval Framework process is a framework that enables three levels of users to develop, configure, and use transaction approvals that meet their organizational requirements. For example, the process of submitting a job change request and getting it approved requires defining who will approve the request, the order in which they will approve it, and how it will be routed to approvers.

In contrast to the standard PeopleSoft workflow, which requires advanced technical skills in PeopleSoft PeopleTools to create and maintain, the Approval Framework provides an alternative workflow that is much easier to create and maintain. For example, all of the steps in Approval Framework are defined using PeopleSoft pages rather than underlying PeopleSoft PeopleCode, so functional users can design and maintain workflow using these online PeopleSoft pages instead of requiring technical developers to create workflow rules.

Many PeopleSoft products are delivered with integrated Approval Frameworks, consult your application documentation.

To implement the Approval Framework process, the framework for building and interpreting workflow approvals brings together these users:

- Application developers

Application developers adapt applications for approval with minimal coding, using a defined setup process. Making this possible is the Approval Framework, which provides a common implementation that other applications can use. Application developers integrate their applications with the Approval Framework using the Register Transactions page, where they register an application and describe its components, event handler, and records. The register stores the approval process IDs that developers create for applications.

---

**Note.** The PeopleSoft product delivers the transaction registry for delivered Approval Framework processes.

---

After an end user creates an application transaction and submits it for approval, the application hands the transaction over to the Approval Framework, which finds the appropriate approval process definition and launches the Approval Framework.

- Functional business analysts

Functional business analysts design Approval Framework for use with an application. This includes setting up stages, paths, steps, recipients, and notifications for each approval process ID. Analysts identify the application-supplied transaction definition on which to base approval process definitions. They use the Approval Process Definition page to define processes for approving transactions. These processes can be used repeatedly to guide transactions through their approval process.

Many businesses need different approval routings for different types of transactions. To support such variations, you can configure multiple approval process definitions for each transaction within an application. In addition to process IDs, approval process definitions are effective-dated.

- End users

End users create transactions and then use an approval process with approvers and reviewers within an approval flow. Using this process, the different end users can approve or deny requests, monitor transaction statuses, and audit approvals.

---

## Understanding Transaction Approval Flows

After an approval process is defined, validated, and made active, the system can submit a transaction for approval. Each PeopleSoft application typically has a top-level database record that distinguishes one transaction from the next. These top-level records are called header records. When a transaction is submitted for approval, the Approval Framework combines the approval process definition with the header record instance and line records, if line level approval is configured, to create a unique approval process instance. This approval process instance is routed from approver to approver, as configured in the approval process definition.

Approvals use two levels of processing: header and line. Business analysts set up the approval process definition that determines the flow of the approval at both levels. The approval process consists of:

- Stages

A stage is one part of an approval process that can contain multiple parallel paths but must be at the same header or line record level. The system executes stages in sequence where one must complete before the next one begins. A stage can be at either a header level or at a line level. Stages at a line level make it possible for approvers to sign off separately on individual line items for a single transaction. Approval Framework sees each header and each line as individual pieces. A line is a child of the header. A header stage acts on the unique header while a line stage acts on each line. A stage consists of one or more paths.

- Paths

A path contains a sequence of steps. Within a stage, paths execute in parallel. Path entry criteria determines whether or not a path executes for a given transaction or transaction line.

- Steps

A step represents one or more approvers or reviewers. Steps within a path execute in sequence. Separate criteria for each step determines whether or not that step executes. Each step can also have a set of reviewers. Reviewers are notified about transactions that are pending approval by email, through the worklist, or both. However, the workflow proceeds without waiting for reviewers to act.

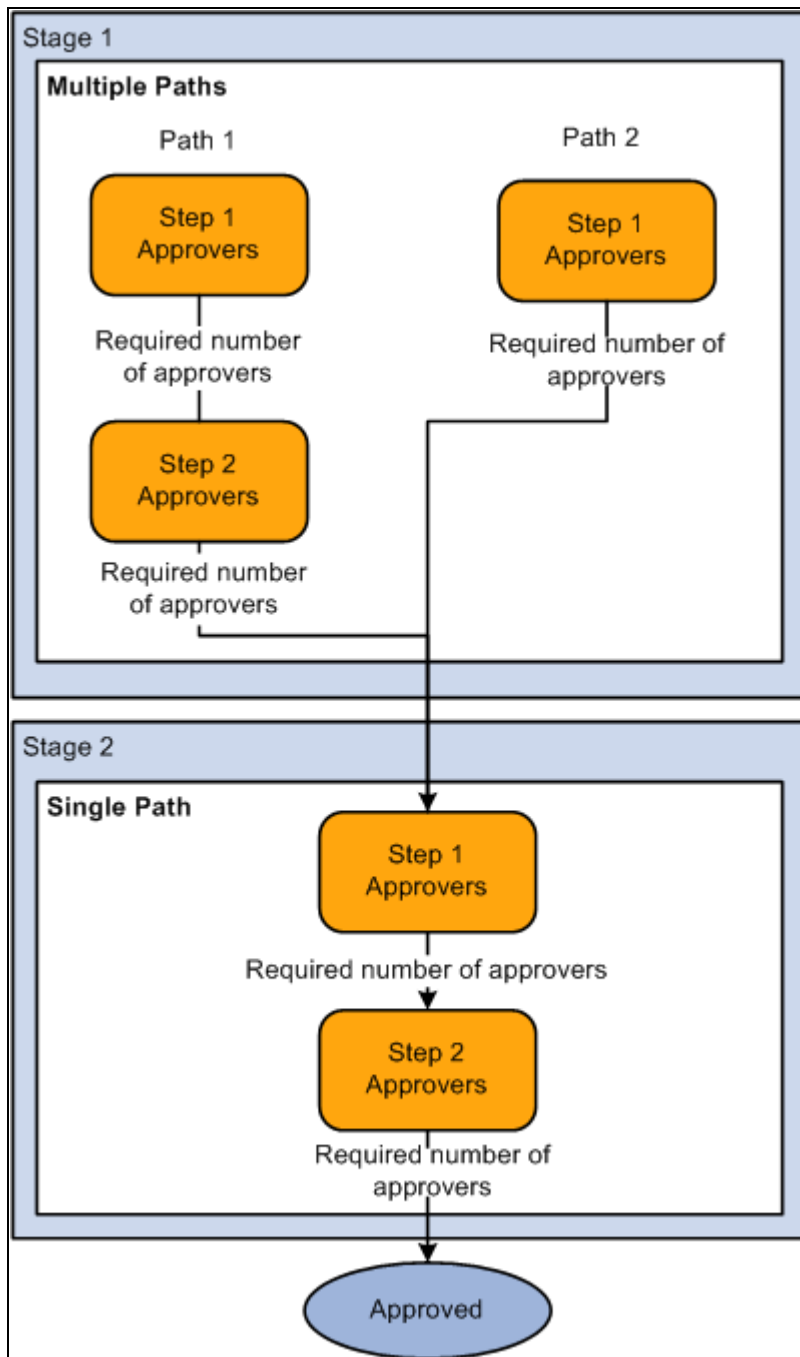
The system notifies approvers by using email, worklist, or both, of pending approval steps, which can require one, all, or a fixed number of approvers. You can specify approvers by roles, queries, fixed lists, or by using custom application classes. Once the required number of approvers have approved a step, the Approval Framework advances to the next step. If the workflow has no further steps, it advances to the next stage.

---

**Note.** While the configuration may require multiple approvers to approve a step before it advances, any single approver can deny a step. The moment an approver denies a step, the transaction stops moving forward in the approval process. If the transaction is at a line level, other lines will continue to move forward. If the denial is at a header level, the approval process terminates, and the entire transaction is denied.

---

This diagram illustrates how the approval process uses stages, paths, and steps for routing approvals:



Example Approval Framework showing stages, paths and step

In this example there are 2 stages. In stage 1 there are 2 paths. Each step within the path is executed in sequence, when the required number of approvers have approved a step, it is advanced to the next step within the path. Paths are executed in parallel. When all paths within the stage are approved, the workflow advances to the first step in the next stage.

---

## Understanding Header- and Line-Level Approvals

Many PeopleSoft transactions have a top-level record (known as a header) with keys that uniquely identify a single transaction in an application. In addition to the header record, a transaction may also contain a more detailed line level record (known as line-level).

The Approval Framework process uses an application's header keys to correlate approval processes and application transactions. For example, when you open an application transaction, the Approval Framework enables you to submit the request for approval only if you haven't already submitted it. This check is possible because the system correlates the application header keys with the approval process instance keys.

If a particular transaction supports line level processing, an analyst can configure the approval process so that different line-items in the same request follow different routes.

You can deny some lines of the request and approve others, making line-level approvals independent for each line. For example, a request can contain multiple lines, and you might want to use special approvers for certain lines based on their area of expertise.

---

**Note.** If a transaction has multiple line items, 1 of which is denied and the rest are approved, the overall transaction is approved.

---

An Approval Framework process might have mixed header and line-level approval routings. For example, department managers might exercise budgetary control over the request as a whole, while commodity approvers still examine only those line-items which fall within their expertise. Final approval requires both types of approvers to sign off on the request.

When a request is approved, the Approval Framework notifies the application, which then takes source end actions:

- End actions.

An approval of one transaction often leads to the creation of another transaction, or triggers another business process. The Approval Framework supports this trigger by providing a call-back mechanism for event notification. For example, when a requisition is approved, it can be sourced—an action follows final approval, which is the end action.

- Line-level versus header-level end actions.

Use line-level approvals to make it possible for an action to be taken on different line items upon their approval, without waiting for the approval of other line items in the requisition. You can source line items as soon as they are approved.

This action is possible only if line-level approval routings are at the end of the process and require no further review. In this case, the application can act on the individual lines as they get approved. The Approval Framework notifies the application of significant approval-related events.

Header actions allow the transaction lines to be grouped together and processed as one unit.

---

## Understanding Criteria for Approval Framework Processes

Criteria is an entity that evaluates to true or false. It uses transaction-specific information to determine if a transaction requires approvals and follows the approval paths and steps that evaluate to true. To set the context for the criteria, the approval process definition provides the transaction keys as bind values.

Criteria is set up at three levels as shown in this table:

<b><i>Set Up Level</i></b>	<b><i>Description</i></b>
Process Definition ID	This criteria is used to determine which Process Definition ID to use to process the approval.
Path	This criteria is used to determine if the approval will follow the path.
Step	This criteria is applied to the individual Approvers defined for the step.

There are 3 different types of criteria you can apply to an approval process.

<b><i>Criteria Type</i></b>	<b><i>Description</i></b>
Always True	No criteria is needed, the approval process will always be triggered.
Application Class	An application class contains the logic used to determine if the workflow approval task evaluates as true.
User Entered	Criteria is based on record and field combinations. The criteria can be either value or monetary-based. A logical operator and a value are used to evaluate the condition.

---

## Understanding Approval Features

Using a combination of features, you can create rules for different types of approvals. This section discusses:

- Ad hoc path approval
- Ad hoc review
- Self-approval
- Route to requester
- Skip prior path
- Auto approval
- Alternate approvers
- Push back



- Approval comments

### ***Ad Hoc Path Approval***

If Ad Hoc Approval Status Monitor is implemented for an approval transaction, then during the approval process, approvers can add other approvers or reviewers to the current or a later stage of the approval process. This action is called *ad hoc approval*, and it only applies to the approval instance in which the addition occurs and does not affect the underlying process definition used for other requests.

You can insert ad hoc approvers and reviewers in serial or parallel with existing approvers:

- For serial approvals, each approval in the process is sequential.

Users can add approvers and reviewers only after the current pending step or later.

- For parallel approvals, the sequence does not matter.

Users can insert an ad hoc step in an ad hoc path in any currently pending or subsequent stage.

You can add ad hoc approvers once the transaction is submitted. The Approval Framework launches the previewed approval process instance if requested by the application developer's code.

If you have an ad hoc approver user list defined in the transaction registry, only the users within that list can be added as an ad hoc approver or reviewer.

See [and Chapter 4, "Defining the Approval Transaction Registry," Registering Approval Transactions, page 38.](#)

### ***Ad Hoc Review***

Ad hoc reviewers are users that an approver or requester would like to review a transaction. Ad hoc reviewers are notified and provided with a link in a worklist entry or email to the transaction, if the process is so configured. Ad hoc reviewers do not approve or deny transactions, they can add comments.

### ***Self-Approval***

A requester can also be an approver. If requesters approve their own transactions, it's called self-approval. A check box setting enables self-approval. If enabled, the requester's approval is assumed, and the process continues. However, you can establish criteria that helps control the requester's approval authority. For example, you can place a limit on the dollar amount for the requester so that if the transaction is over that amount, the requester is not used as an approver.

If self-approval is not enabled, then the requester must manually approve the transaction.

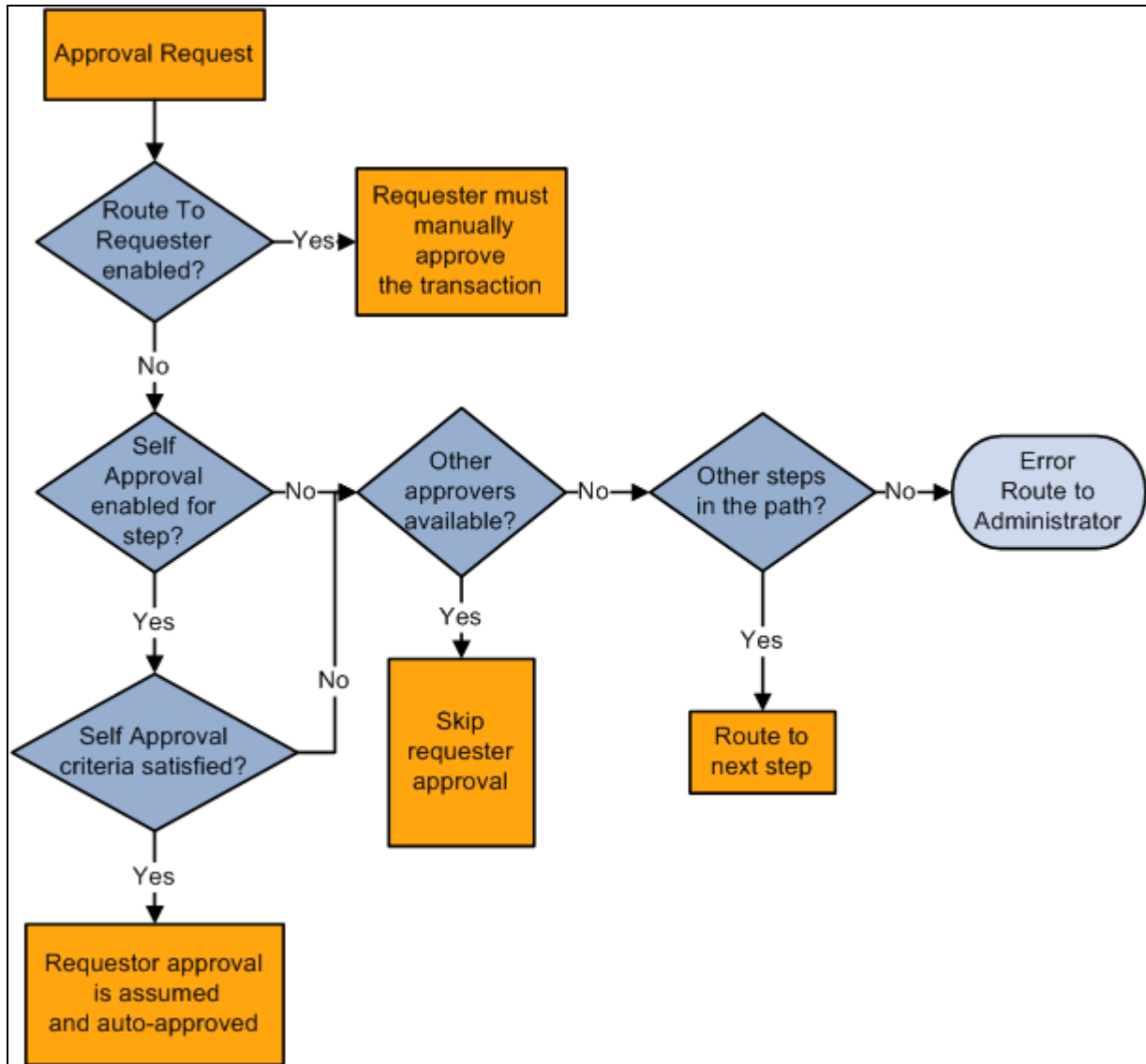
If self-approval is enabled, then the self-approval criteria must be specified. Then, if the requester appears as an approver, the criteria is evaluated. If the criteria are met, then the requester's approval is assumed and the minimum approvers for that step decrements by one.

When a requester is an approver, you can configure the path to skip the steps in that path prior to the requester's step.

Self-approval is configured at the step level.

### Route to Requester

Route to requester is a feature that informs the system that an approval should be sent to a requester. In some cases the requester may already be listed in the approval path. The flow of this option is also dependant on the self-approval feature as displayed in this diagram:



Route to requester and self-approval feature flow

- If route to requester is selected, the requester must always manually approve the transaction regardless of the self-approval setting.

---

**Note.** Even if the self-approval criteria is met, the requester must manually approve the transaction.

---

- If neither route to requester nor self-approval are selected, and the requester is the only approver, the system tries to identify additional steps in the path and routes the transaction to the next step if available. If no additional steps are available, it generates an error and routes the transaction to the administrator.

- If neither route to requester nor self-approval are selected, and there are other approvers in the path, the requester approval is skipped.
- If route to requester is not selected and self-approval is selected, the self-approval criteria are checked. If the requester meets the approval criteria, the approval is assumed and auto-approved.

If the requester does not meet the approval criteria, and other approvers are available, the requester approval is skipped. If self approval criteria are not met and there aren't other approvers in the same step, the system routes the transaction to the next step in the process if applicable, or to the administrator if there are no more steps in the path.

### ***Skip Prior Path***

When skip prior to requester is enabled, the system begins the approval flow at whatever step in which the requester is an acknowledged approver.

For example, if a vice president orders supplies for herself, the system skips application approval steps leading up to the step for which the vice president is an acknowledged approver.

### ***Auto Approval***

When auto approval is enabled, the system remembers an approver's action for that process at the header- or line-level, and applies the same action automatically for any subsequent appearance in the Approval Framework routing.

For example, if the approver approves line one and the line appears again as a required approval for the specific approver, the approval is remembered for the line. In this example, the approval is given only to that specific line in the process. However, if an approval is made at the header level, approval is granted to all of its lines automatically.

As a hierarchical system, the Approval Framework supports auto approval in these situations:

- Header stage that is followed by another header stage.
- Header stage that is followed by a line stage.
- Header stage that is followed by a line stage, which is followed by another header stage.
- Line stage that is followed by another line stage.

Auto approval is not supported in processes where a line stage is followed by a header stage. The reason being that the approver of a line may only have approval privileges to some but not all lines within the same header and therefore should not be given the power to approve the header through the Auto Approval option.

### ***Alternate Approvers***

Alternate workflow approvers are users who are assigned to receive emails and worklist notifications for the primary approver when the primary approver is not available. When you identify an alternate approver, you enter a date range during which you want the alternate to fill in. The system determines if the alternate is in place for the date range.

---

**Note.** Alternate users are defined in the User Profile. Navigate to PeopleTools, Security, User Profiles, User Profile.

You can also use the *FindAlternate* method in the EOAW\_CORE:Utils class to select alternate approver.

---

### **Push Back**

Push back is an optional feature that can be implemented in the Approval Monitor. If implemented, push back takes a currently pending step out of pending status and requeues the previous step to its approvers. The meaning of push back is that the approver is questioning the prior step's approval and is requesting clarification. Push back is only possible within a path, therefore, the first step of a path cannot push back.

---

**Note.** The push back feature ignores auto self-approval.

---

See [Chapter 10, "Using the Approval Monitor," Configure Monitor Approvals, page 97.](#)

### **Approval Comments**

Requesters can add comments to transactions, and approvers can associate their comments with the approval process rather than the request transaction directly. The Approval Framework Monitor provides a mechanism for associating comments with a particular approval process instance, which is tied to a particular application transaction. Approvers can view comments added by another approver, but they cannot change previous comments.

---

## **Understanding Tasks in the Approval Framework**

This section details the steps to implement and use Approval Framework. It describes tasks that application developers, business analysts, and end users perform in conjunction with Approval Framework.

To implement and use Approval Framework:

1. Application developers register information with the Approval Framework by using the Register Transactions page.
2. Functional business analysts define the approval process definition for an application transaction.

Essentially, analysts determine the approval transaction registry entry on which the process definition is to be based and then define the details of the process. The approval process definition includes definitions of stages, paths, steps, and criteria.

3. Requesters submit a transaction for approval.

This action launches the approval process. The Approval Framework reads the approval process definition and queues the transaction for approvers.

4. The system queues an approval task to an approver or reviewer using email notification, worklist entry, or both.

The URL encoded in the worklist entry points to the corresponding approval component.

5. Approvers and reviewers take actions on transactions using the Approval Monitor. Reviewers may only add comments.

When an error or violation of criteria or rules occurs during the approval process, the error is routed to the administrator for resolution, if there are no additional steps in the path.

---

**Note.** The error conditions for static steps are no approvers or not enough approvers at a step.

---



## Chapter 3

# Setting Up Approval Framework Process Definitions

This chapter discusses how to define Approval Framework processes.

---

## Defining the Setup Process Definitions Component

Business analysts use the Setup Process Definition page to define an approval definition process. The process is made up of stages and their paths and steps. The approval steps that you place on the approval path represent the approval levels that are required for a transaction.

This section discusses how to:

- Define Approval Framework processes.
- Define criteria for Approval Framework processes.
- Define paths for Approval Framework processes.
- Define steps for Approval Framework processes.

To set up approval processes, use the Approval Process Setup (EOAW\_PRCS) component.

## Pages Used to Define Approval Framework Processes

<i>Page Name</i>	<i>Definition Name</i>	<i>Navigation</i>	<i>Usage</i>
Setup Process Definitions	EOAW_PRCS_MAIN	Enterprise Components, Approvals, Approvals, Approval Process Setup	Define approval process stages.

<b>Page Name</b>	<b>Definition Name</b>	<b>Navigation</b>	<b>Usage</b>
Criteria Definition	EOAW_CRITERIA	<ul style="list-style-type: none"> <li>Click the Definition Criteria link on the Setup Process Definitions page.</li> <li>Click the Alert Criteria link on the Setup Process Definitions page.</li> <li>Click the Criteria link from the Setup Process Definitions page in the Path section.</li> <li>Click the Criteria link from the Setup Process Definitions page in the Steps section.</li> </ul>	Define criteria for workflow approvals.
Notifications	EOAW_NOTIFY_DEF	Click the Definitions Notifications link from the Setup Process Definitions page.	Define notification options.
Timeout Options	EOAW_TIMEOUTDEF	Click the Timeout Options link from the Setup Process Definitions page.	Define global timeout and escalation settings.
Approval Path Definition	EOAW_PATH_SEC	Click the Details link within the Paths group box on the Setup Process Definitions page.	Set up workflow approval paths.
Approval Step Definition	EOAF_STEP_SEC	Click the Details link within the Steps group box on the Setup Process Definitions page.	Define steps for workflow approvals.

## Defining Approval Framework Processes

Access the Setup Process Definitions page (Enterprise Components, Approvals, Approvals, Approval Process Setup).



### Setup Process Definitions

[Clone Approval Process](#) | 
 [Approval Process Viewer](#) | 
 [Preview Approval Process](#)

**Process ID:** CAApproval  
**Definition ID:** SHARE  
**Effective Date:** 01/01/1900  
**Description:**

**Definition Options**

[Definition Criteria](#) | 
 [Alert Criteria](#) | 
 [Definition Notifications](#) | 
 [Timeout Options](#)

**\*Admin Role:**    
**\*Status:**    
**Priority:**

☐ Default Process Definition  
☒ Take Action on Line Completion  
☒ User Auto Approval  
☐ Route to Requester  
☐ Include Requester

**Stages** Find | View All | First 1 of 4 | Last

**\*Stage Number:**  **Description:**  **Level:**

**Paths** Find | View All | First 1 of 1 | Last

**Description:**  **\*Source:**  [Details](#) | [Criteria](#)

**Steps** Customize | Find | View All | First 1 of 1 | Last

	Description	Approver User List	Details	Criteria			
1	Cash Advance Approvers	Cash Advance Approvers					

[Expand/Collapse All](#)

#### Setup Process Definitions page

Business analysts use this page to define an approval definition process. The process is made up of stages and their paths and steps. The approval steps that you place on the approval path represent the approval levels that are required for a transaction.

You can develop approval processes that:

- Meet organizational or tiered approval limits.
- Use parallel processing to allow multiple transaction entities to be processed at the same time.
- Use staged approvals that require one sequence of processing to complete before another can begin.

Typical approval processes might include:

- Monetary based approval criteria.
- Two different approvers for each step, where both approvers at a step must approve the request for it to advance to the next step.

#### Process ID

The Process ID created on the Register Transactions page.

See and [Chapter 4, "Defining the Approval Transaction Registry," Understanding the Approval Transaction Registry, page 37.](#)

<b>Definition ID</b>	Enter any identification code that provides meaning to you. This identifier is used as a search field on the Monitor Approvals page. <hr/> <b>Note.</b> When upgrading from a previous release, the SetID field is used for the Definition ID. <hr/>
<b>Effective Date</b>	Indicates the date on which this approval process became effective and ready for system use. This value applies to approval processes for a particular approval process ID and definition ID, and it includes PeopleSoft functionality associated with effective-dated entities. For instance, if multiple approval processes are active with the approval process ID, definition ID, and effective-date specification, then the system uses the most current effective dated row.
<b>Description</b>	Enter a description for the approval process.
<b>Clone Approval Process</b>	Click this link to clone the approval process.
<b>Approval Process Viewer</b>	Click this link to access a graphical tool, which enables you to view each stage, path, and step of the approval process. <hr/> <b>Note.</b> An SVG Viewer is required for this feature. <hr/> <b>Note.</b> When you are viewing the graphic tool in the Approval Process Builder page, if you elect to make changes the system will return you to the standard Approval Process Definition page. <hr/>
<b>Preview Approval Process</b>	Click this link to view what a workflow instance would look like if it were running.

### ***Definition Options***

This section of the page is used to define options for the approval process.

<b>Definition Criteria</b>	Click to access the Criteria Definition page, where you can define user and field criteria along with monetary and application class criteria for this process. This page works similar to the other Criteria Definition pages that are used for paths and steps, however this criteria is used to determine which Definition ID is to be used to process the Approval.  See and <a href="#">Chapter 3, "Setting Up Approval Framework Process Definitions," Defining Criteria for Approval Framework Processes, page 23.</a>
----------------------------	---

<b>Alert Criteria</b>	<p>Click to access the Criteria Definition page, where you can define user and field criteria along with monetary and application class criteria for this process. This criteria can be evaluated by applications to highlight conditions of a transaction to be approved. For example, a one-time shipping address used only on this request.</p> <p>Alert Criteria is not used to determine how an approval should route. It is used by the application to determine if messages (Alerts) should be displayed as part of the approval process.</p>
<b>Definition Notifications</b>	<p>Click to access Approval Framework Notifications page, where you can define notification options to the override one or more of the process definition notification options defined in the Configure Transactions component.</p> <p>See <a href="#">and Chapter 4, "Defining the Approval Transaction Registry," Notification Options, page 46.</a></p>
<b>Timeout Options</b>	<p>Click to access Approval Framework Timeout Options page, where you can set global timeout and escalation settings at the Approval Definition level. Path level timeout settings will override the Approval Definition level settings if they differ.</p>
<b>Admin Role</b> (administrative role )	<p>Select the PeopleSoft role used by workflow to route the transaction to all users filling that role in case of an error during approval processing.</p> <hr/> <p><b>Note.</b> The error conditions are no approvers or not enough approvers.</p> <hr/> <p>See <a href="#">and Chapter 5, "Defining Notification Templates and Users for Approval Framework," Defining User Lists, page 56.</a></p>
<b>Status</b>	<p>Select the current state of this approval process. The values are:</p> <p><i>Active:</i> Indicates the approval process is available for use.</p> <p><i>Inactive:</i> Indicates the approval process is not available for use.</p> <p>A transaction that has started with a specific definition continues using that definition, even if the status is <i>Inactive</i>.</p>
<b>Priority</b>	<p>Enter the priority for the definition. Priority 1 is the highest. When a definition is not explicitly passed to the Approval Framework by the calling application, the Approval Framework uses the Definition Level Criteria to determine which definition to use. If multiple definitions return a criteria of true, the definition with the highest priority is used.</p> <hr/> <p><b>Note.</b> Multiple Definition ID's with the same priority may result in inconsistent behavior, in the event that multiple definition ID's match.</p> <hr/>
<b>Default Process Definition</b>	<p>Select to indicate that the system should use this process definition as the default when no other definition ID matches the criteria entered.</p>

**Take Action on Line Completion**

Select to allow each line to continue to the next step of the approval process, without waiting for other lines within the transaction to complete. This setting applies to approval processes that have a line-level stage at the end of the process.

When this check box is selected, the approved lines of a transaction can move forward to the next approval step even though some of the lines in the transaction have not yet completed the approval process. As each line completes the approval process, the header and line status are updated immediately so that the individual line can move forward without having to wait for all the lines of the transaction to complete workflow approval.

When this check box is clear, all lines within a transaction must complete the approval process before any line of the transaction can flow to the next approval step. For example, a requisition line cannot be sourced to a purchase order until all lines of the requisition have completed the approval processing. The header and line statuses remain in a pending status until the last line of the transaction has completed the approval process. Once that last line has had an approval action made against it, then the header status is updated, as well as each line status. An approval action could be a denial as well as an approval. For example, if there are two lines on a transaction, one line is approved and the second line is denied, then the approval process is complete and the approved line can move forward to the next step.

**User Auto Approval**

Select to enable the system to remember an approver's action for this process. The next time this approval process is presented to the approver, the system automatically applies the approver's selections. The automatic application of steps in the process is left in place until you clear the User Auto Approval check box.

This setting applies to the specific line or header the approver had previously approved in this process only. A header approval implies line approvals for all lines.

**Route to Requester**

Select this check box to route this approval to the requester. When this check box is selected and the requester is also an approver, the requester must manually approve the transaction, unless the minimum approvals have already been satisfied.

If this check box is not selected and the requester is also an approver, the system will check if self-approval is active. If self-approval is active, the criteria will be checked.

See [Chapter 2, "Understanding the Approval Framework," Route to Requester, page 12.](#)

**Include Requester**

Select to enable the user to insert the requester in the event the requester is not equal to the originator.

**Stages**

This section of the page is used to define the stages. An approval process can contain multiple parallel paths but they must be at the same header or line record level. The system executes stages in sequence where one must complete before the next one begins.

<b>Stage Number</b>	Enter a number indicating sequence for this stage.
<b>Description</b>	Enter a description for the stage.
<b>Level</b>	Select either <i>Line</i> or <i>Header</i> .

The system executes stages in sequence where one must complete before the next one begins. A stage can be at either a header level or at a line level. Stages at a line level make it possible for approvers to sign off separately on individual line items for a single transaction. The Approval Framework sees each header and each line as individual pieces. A line is a child of the header. A header stage acts on the unique header while a line stage acts on each line. A stage consists of one or more paths.

### **Paths**

This section of the page is used to define the paths for the stage. Use the Details and Criteria links to add the information for each path.

See and [Chapter 3, "Setting Up Approval Framework Process Definitions," Defining Criteria for Approval Framework Processes, page 23.](#) and [Chapter 3, "Setting Up Approval Framework Process Definitions," Defining Paths for Approval Framework Processes, page 30.](#)

### **Steps**

This section of the page is used to define the steps for the path. Use the Details and Criteria links to add the information for each step.

See and [Chapter 3, "Setting Up Approval Framework Process Definitions," Defining Criteria for Approval Framework Processes, page 23.](#) and [Chapter 3, "Setting Up Approval Framework Process Definitions," Defining Steps for Approval Framework Processes, page 32.](#)

## **Defining Criteria for Approval Framework Processes**

Access the Criteria Definition page (click a Criteria link on the Setup Process Definitions page).



The screenshot shows a web interface titled "Criteria Definition". Below the title, there is a label "\*Criteria Type:" followed by a dropdown menu. The dropdown menu is currently displaying the text "Always True".

Criteria Definition page

Use this page to define the different types of criteria you want to apply to an approval process. You can create definitions consisting of a field with a logical operator and a value of definitions consisting of an application class that takes in transaction data to process the approval.

**Criteria Type**

Select one of these options:

- *Always True* informs the system to trigger this approval process. No criteria is needed. The system will only follow paths that evaluate as true.
- *Application Class* requires you to define which specific application class the system uses to determine if the workflow approval task evaluates as true.

---

**Note.** Use the *Application Class* criteria type when the user entered criteria does not contain the necessary level of detail.

---

- *User Entered* requires you to enter all record and field combinations, either value- or monetary-based, that will trigger the workflow to evaluate as true.

**Application Class Criteria (Section)**

Use this section to assign application packages as criteria for the approval process definition. When you define a class, the system uses it along with other criteria you enter to process the approval.

The screenshot shows a web form titled "Criteria Definition". At the top, there is a dropdown menu for "\*Criteria Type:" with "Application Class" selected. Below this is a section titled "Application Class Criteria" with a blue header. Inside this section, there are two input fields: "Root Package ID:" with the value "EX\_APPROVAL\_DEFN\_CRITERIA" and a magnifying glass icon, and "Application Class Path:" with the value "DefinitionCriteria" and a magnifying glass icon. At the bottom of the section, there is a blue italicized text block that reads: "For criteria based on an application class, you will need to extend EOAW\_CRITERIA:CriteriaBase and implement the 'Check' method. If you need further assistance, contact your system administrator."

Criteria Definition page showing criteria type Application Class

**Root Package ID**

Select the primary application package. This selection is the parent class for other packages or for child application classes.

**Application Class Path**

Select a path that describes a specific class within the root package.

**Application Class: Example**

Here is an example of an application class written for approval criteria that extends EOAW\_CRITERIA:CriteriaBase and implements the "Check" method. This example can also be found in EOAW\_CRITERIA:EXAMPLES.

```

import EOAW_CRITERIA:DEFINITION:*;

class CriteriaClassExample extends EOAW_CRITERIA:DEFINITION:CriteriaBase
  1- method CriteriaClassExample(&rec_ As Record);
  2- method Check(&recBind As Record) Returns boolean;
  3- method CheckwithComments(&recBind As Record) Returns array of string;
private
  instance Record &rec;
  method CheckSQL(&recBind As Record) Returns boolean;
end-class;

method CriteriaClassExample
  /+ &rec_ as Record +/
  %Super = create EOAW_CRITERIA:DEFINITION:CriteriaBase(&rec_.EOAWCRTA_ID.Value);
  &rec = &rec_;
end-method;

method Check
  /+ &recBind as Record +/
  /+ Returns Boolean +/
  /+ Extends/implements EOAW_CRITERIA:DEFINITION:CriteriaBase.Check +/
  Local boolean &queryStatus;
  rem %This.CheckQuery(&recBind);
  Local boolean &sqlStatus = %This.CheckSQL(&recBind);

  Return &sqlStatus;

end-method;

method CheckwithComments
  /+ &recBind as Record +/
  /+ Returns Array of String +/
  /+ Extends/implements EOAW_CRITERIA:DEFINITION:CriteriaBase.CheckwithComments +/
  Local array of string &aryComments = CreateArrayRept(" ", 0);

  Return &aryComments;

end-method;

method CheckSQL
  /+ &recBind as Record +/
  /+ Returns Boolean +/
  Local string &sqlDef, &SQLText;
  Local integer &i;
  Local Rowset &rsResults;

  &SQLText = "SELECT 'X' FROM PS_INSTALLATION";
  Local string &output;
  REM ***** Specify a query that uses some or all of the transaction keys as⇒
  input;
  SQLExec(&SQLText, &output);
  If All(&output) Then
    Return True
  Else
    Return False
  End-If;
end-method;

```

Note that:

1. *CriteriaClassExample* is the class name. It should match at all levels. The input is a record that contains the field EOAWCRTA\_ID. The method must have this parameter.

2. The *Check* method is the main method doing the work. It validates if the criteria is true or false. It can call anything you want to add. In this case, it is calling a private method called *CheckSQL*. The *&recBind* parameter is the transaction cross reference record. The cross reference record ties the approval workflow engine to the transaction. The transaction keys are fields on the cross reference record. There is also a field called *RECNAME* so that it is possible to reconstruct the transaction record.
3. The *CheckwithComments* method is nearly identical to the *Check* method. The difference is that it returns any "descriptions" from the criteria page for criteria that was true. This is useful in the area of *Alert Criteria* on the Approval Process Definition page. If there are simple notifications (messages) that the application wants to display, they can be done with the *CheckwithComments* method. One example of this is the *one-time* address check in eProcurement.

Here is an example of how the call is made from *Self Criteria*:

```
If (All(&c) And
    &c.Check(%This.path.thread.rec)) Then
    &self_ok = %This.utils.SELF_OK;
End-If;
```

And this is the constructor for the Criteria object:

```
&SelfAppCriteria = &fact.GetCriteria(&rec.EOAWSELF_CRTA_ID.Value);
```

### ***User Entered Criteria (Section)***

Use this section to define additional criteria for the approval.



Criteria Definition

\*Criteria Type:

User Entered

☒ All Criteria Needed to Satisfy

User Entered Criteria

Find | View All | First | 1 of 1 | Last

Description:

HRS\_JOB\_OPENING\_ID

Field Criteria

Record: HRS\_JOB\_OPENING\_ID | Field Name: HRS\_JOB\_OPENING\_ID

Personalize | Find | | First | 1-2 of 2 | Last

	*Criteria Operator	Value		
1	Greater Than	10000	+	-
2	Less Than	9900	+	-

Monetary Criteria

Amount Record:

Amount Field:

Currency Field:

Operator:

Greater Than

Amount:

0.000

Currency Code:

Rate Type:

Criteria Definition page showing criteria type User Entered

Description

Enter purpose of the alert.  
For example, if you are using a one-time ship-to address, create a description that indicates that a one-time ship-to address is attached to the requisition.

All Criteria Needed to Satisfy

Select to indicate that all criteria defined on the Criteria Definition page must be met in order to trigger the workflow to evaluate as true.

Field Criteria (Section)

Use this section to select a record and field on which to control and filter ranges of data or types of data placed in the file you want to use in the approval process. As you add rows in this section to build your approval criteria, they get translated into a long SQL string by the approval framework, each of them connected by the OR operator. For example, the *where* clause of the SQL generated for the criteria that is specified in the screenshot above looks like this:

"where HRS\_JOB\_OPENING\_ID > '10000' OR HRS\_JOB\_OPENING\_ID < '9900'"

---

**Important!** If the *Not Equal To* criteria operator is used in a row, subsequent rows will be connected using the AND operator instead of the OR operator. Here is an example of an SQL constructed for field criteria that contains the *Not Equal To* operator:

```
"where HRS_JOB_OPENING_ID <> '10500' AND HRS_JOB_OPENING_ID > '10000' AND
HRS_JOB_OPENING_ID < '11000'"
```

The approval framework *does not* support the insertion of parentheses around specified criteria values in SQLs.

---

<b>Record Name</b>	<p>Select a record that you want to use in defining approval criteria.</p> <p>For example, if you are indicating a one-time address, identify <i>PO_ADDR_REQ_VW</i> as the record.</p>
<b>Field Name</b>	<p>Select a field you want to use to define approval criteria. The values you define in the remaining field criteria grid are those that are used in determining the approval criteria.</p>
<b>Criteria Operator</b>	<p>Select the operator to build the approval criteria.</p> <p>These operators are available:</p> <ul style="list-style-type: none"> <li>• <i>Between:</i> The specified field has a value that is within the range identified using the Value fields.</li> <li>• <i>Equals:</i> The specified field has the same value that is entered in the Value field.</li> <li>• <i>Greater Than:</i> The specified field has a value that is greater than the value entered in the Value field.</li> </ul> <hr/> <p><b>Note.</b> The <i>greater than or equal to</i> operator (as typically represented by the <math>\geq</math> symbol) is not supported.</p> <hr/> <ul style="list-style-type: none"> <li>• <i>Is Blank:</i> The specified field has no value. When selected, the Value field is not displayed.</li> <li>• <i>Is Not Blank:</i> The specified field has a value. When selected, the Value field is not displayed.</li> <li>• <i>Less Than:</i> The specified field has a value that is less than the value entered in the Value field.</li> </ul> <hr/> <p><b>Note.</b> The <i>less than or equal to</i> operator (as typically represented by the <math>\leq</math> symbol) is not supported.</p> <hr/> <ul style="list-style-type: none"> <li>• <i>Not Equal To:</i> The specified field has a value that is either greater than or less than the value entered in the Value field.</li> </ul>
<b>Value</b>	<p>Use the two Value fields to define a range upon which you want the operator criteria to evaluate. The second value is only used when the selected criteria operator is <i>Between</i>.</p>

## Monetary Criteria (Section)

Use this section to establish approval criteria for requisition amounts. The system uses the values you define to determine the routing for approving the requisition. When the system evaluates the criteria for an approval process or a step or path within the process, it uses monetary values you define in this section.

The screenshot shows the 'Criteria Definition' page. At the top, the '\*Criteria Type:' is set to 'User Entered'. Below this, the checkbox 'All Criteria Needed to Satisfy' is checked. The 'User Entered Criteria' section is expanded, showing a 'Description:' field with the text 'Step Criteria Definition for Line Level'. Below the description is the 'Field Criteria' section, which includes a 'Record:' field and a 'Field Name:' field. A table below these fields shows the criteria configuration:

	*Criteria Operator	Value	Value
1	Greater Than	500.000	

Below the table is the 'Monetary Criteria' section, which includes the following fields:

- Amount Record:** PO\_APPR\_LN\_VW
- Amount Field:** MERCHANDISE\_AMT
- Currency Field:** CURRENCY\_CD
- Operator:** Greater Than
- Amount:** 500.000
- Currency Code:** USD
- Rate Type:** CRRNT

Criteria Definition page showing criteria type User Entered for monetary amount

The system uses values from fields in this section in conjunction with the Operator field to determine whether to run a step.

- Amount Record** Select the record name containing the amount in the original transaction to be used when comparing the minimum amount required to trigger the step.
- Amount Field** Select the field within the amount record to be used when comparing the requisition to the minimum amount required to trigger the step. The system uses the value you select to evaluate the Amount field.
- Currency Field** Select the currency field that corresponds to the currency code you select in the Currency Code field.
- Operator** Select a value that determines how the system processes the values in the Amount fields. Values include *Between*, *Greater Than*, and *Less Than*.

**Amount**

Use the Amount fields to define an amount range for use with the Operator field.

In the first field, enter the minimum amount required on the transaction in order to trigger the step. The system identifies all lines in the transaction that meet the criteria defined in the Amount Record and Amount Field fields. The amounts on these lines are totaled based on the Amount Record and Amount Field specified. If the requisition total is higher than this minimum amount, the criteria is met. If no amount is specified, zero is considered the minimum.

---

**Note.** If you select Operator with a value of *Between*, a second Amount field becomes active.

---

**Currency Code**

Select the monetary unit you want to use for the approval.

**Rate Type**

Select how the system arrives at the currency value, such as the current rate or a financial rate.

## Defining Paths for Approval Framework Processes

Access the Approval Path Definition page (click the Path Details button or the Details link on the Setup Process Definitions page).

**Approval Path Definition**

Criteria

Approval Path: 1

\*Step Source: Dynamic

Description: Cash Advance Reviewer

Long Description:

☐ Notify Admin on No Approvers

☐ Skip Prior Steps for Requester

☐ Check Authorization

☐ Skip Unauthorized Users

Timeout Options						
Customize   Find       1 of 1						
	*Escalate Option	Hours	Days	Reassign To	User List	Use Proxy
1	Notify Participant					<input checked="" type="checkbox"/>

Approval Path Definition page

Use this page to set up additional parameters that determine how the system processes an approval path. Use the escalations feature to define time elements for when an approver takes too long to approve or deny a pending request.

<b>Criteria</b>	Click to access the Criteria Definition page where you can define user and field criteria along with monetary and application class criteria.
<b>Approval Path</b>	Displays the path name that you are creating or updating. The path provides the sequence of approvers of a request, usually from a single reporting (or other) hierarchy.
<b>Step Source</b>	<p><i>Static:</i> Select this source to indicate that you want the system to use the individual user-defined steps when it processes an approval.</p> <p><i>Dynamic:</i> A dynamic path definition contains only one step. When begun, the single step definition could yield any number of instances in sequence.</p> <p>When using the <i>Dynamic</i> source, the system uses the user list on the step definition to initialize the steps in the path. The single step definition is repeatedly run, until the step's user list returns no more approvers. All these instances are queued in sequence.</p> <p>See and <a href="#">Chapter 5, "Defining Notification Templates and Users for Approval Framework," Defining User Lists, page 56.</a></p>
<b>Notify Admin on No Approvers</b> (notify administrator on no approvers)	<p>Select to indicate that the administrator is to be notified if the system does not find an approver for the path. This option is only available when the selected step source is <i>Dynamic</i>.</p> <hr/> <p><b>Note.</b> This is the default behavior for static paths.</p> <hr/>
<b>Skip Prior Steps for Requester</b>	Select to indicate that if one of the approvers in this path is also the requester, then the system is to skip all steps in this path prior to that approver's step.
<b>Check Authorization</b>	<p>This option is only available when the selected step source is <i>Dynamic</i>.</p> <p>Select to bypass the existing criteria and use the Authorize Approvers component and criteria definitions.</p> <p>Check Authorization enforces an exit point for dynamic paths.</p>
<b>Skip Unauthorized Users</b>	<p>This option is only available when the selected step source is <i>Dynamic</i>.</p> <p>Select to bypass an approver that is not qualified to approve the transaction. and exits when an approver is qualified to approve a transaction.</p> <p>For example, if the first approver can approve transactions less than 500 and the second approver can approve transactions less than 5000 and the transaction is for 675, the first approver would be skipped and the transaction would go directly to the second approver. Since the second approver is qualified, no further approvals are necessary.</p>

## Timeout Options

### Escalate Option

*Advanced Approval:* Skip the current approver. Advance Approval will not advance if it is on the last step of the last stage.

*Notify Participant:* Sends an email, or whatever notification is defined in the transaction registry, to the individual.

*Reassign Approval:* Reassigns to a user ID or a user list.

---

**Note.** If you select *Advanced Approval* and defined a User List, a notification is sent to the user list members.

---

### Hours

Enter the number of hours a transaction can remain at one workflow step before being escalated. This field is combined with number of days to determine the total time an approver has to take action on an approval request.

### Days

Enter the number of days a transaction can remain at one workflow step before being escalated. This is the length of time an approver has to do something such as approve or deny a transaction.

### Reassign To

If you have selected *Reassign* as the option, you can enter a user name or a specific user list.

---

**Note.** A user list will reassign to the first user in the list that does not match the current user.

---

### User List

Select the list of users the workflow should be routed to.


### Use Proxy


Select this check box to indicate that the escalation should be routed from the perspective of the current approver (proxy). If the check box is not selected, the escalation is routed from the perspective of the original approver (delegator).

## Defining Steps for Approval Framework Processes

Access the Approval Step Definition page (click the Details link within the Steps group box on the Setup Process Definitions page).

### Approval Step Definition

 Criteria

 Self-Approval Criteria

Sequence Number: 1

Description: Cash Advance Approvers

Approvers

Approver User List: Cash Advance Approvers

Approver Role Name:

Approver Requirements

☒ All Approvers Required

☐ Some Approvers Required

Number of Approvers Needed: 1

☐ Self Approval

☐ External Approver

☐ Route to Requester

☐ Filter Requester

Reviewers

Reviewer User List:

Approval Step Definition page

Use this page to set up additional parameters for the step definition.

Criteria	Click to access the Criteria Definition page, where you can define field criteria along with monetary and application class criteria.
Self-Approval Criteria	Click to access the Criteria Definition page, where you can set up self-approval criteria for a user, including field criteria and monetary and application class criteria.
Sequence Number	Displays the sequence number in which the approval is routed. Each step typically represents a routing to an approver. However, it is possible to route to multiple approvers or reviewers within a step.
Approver User List	Select the type of approver you want to use for this step based on the user list.
Approver Role Name	In addition to a User List, a role can be added to check for additional authorization checking. Select a role that specifies the authority that a user has. The Approval Framework filters approvers returned by the user list for this role. It also enforces the role at the time the approver acts. If the role assignment changes, such as the approver is no longer in the role, the approver is blocked from acting on the requisition.
All Approvers Required	Select to indicate that all approvers at this step are required to approve the transaction at this step. You can select to have all approvers or some approvers approve the transaction at this step.

<b>Some Approvers Required</b>	Select to indicate that it's not required for all approvers to sign off on a transaction. If you select this option, you can define the number of approvers required in the Number of Approvers Needed field.
	<b>Note.</b> After the number is met, the approval advances to the next step.
<b>Number of Approvers Needed</b>	Enter the minimum number of approvers you want to sign off for a requisition at this step. When an approval process is launched and this number can't be met, the system notifies the approval <i>Admin Rolename</i> .
<b>Self Approval</b>	<p>Select to indicate that requesters can also approve their own requisition. This setting only applies if the requester also appears as an approver in the step. You can establish criteria that controls the requester's approval authority by using the Self-Approval Criteria link. If the associated criteria evaluate to true, then self-approval is acceptable. For example, you can place a limit on the dollar amount for the requester so that if the transaction is over that amount, the requester is not used as an approver.</p> <p>If you select self-approval and the criteria is not met, the Approval Framework requires that there be at least one more step after this one in the path. This does not apply to ad hoc steps. Clearing the check box means that self approval is never acceptable.</p>
	<b>Note.</b> If the criteria is not met and no later step exists, the system inserts an additional step. This selection is then routed to the administrator.
<b>Route to Requester</b>	<p>Select this check box to route this approval to the requester. When this check box is selected, the requester must always manually approve the transaction.</p> <p>Route to requester works in conjunction with the self-approval flag.</p> <p>See <a href="#">and Chapter 2, "Understanding the Approval Framework," Route to Requester, page 12.</a></p>
<b>External Approver</b>	<p>Select to indicate that an external approver is supplied the email information. This feature is used to notify someone that is not part of the Employee Portal.</p> <p><b>Note.</b> Notification information for the external approver must be set up on the Configure Transaction page.</p> <p>See <a href="#">and Chapter 4, "Defining the Approval Transaction Registry," Configuring Approval Transactions, page 44.</a></p>
<b>Filter Requester</b>	This check box is only visible if Route to Requester is not selected. When this check box is selected, requester's user Id is filtered from the result set whenever it is encountered.
<b>Reviewer User List</b>	Select the type of reviewer you want to use for this step. Use a user list to map users to certain functional roles. The system then uses the list and its users to run automated business processes. The list defines the user sources who can be used in approval and review steps.



See [and Chapter 2, "Understanding the Approval Framework," Understanding Approval Features, page 10.](#)

**See Also**

*PeopleTools: Workflow Technology PeopleBook*, User List Roles



## Chapter 4

# Defining the Approval Transaction Registry

This chapter provides an overview of the approval transaction registry, lists prerequisites, and discusses how to:

- Register approval transactions.
- Configure approval transactions.

---

## Understanding the Approval Transaction Registry

The approval transaction registry is the interface application that developers use to register an application with the Approval Framework. Transactions that require approvals are candidates for being linked to Approval Framework. You use the Register Transactions page to link the components, event handler, records, and classes that you created into the approval process for an application transaction, such as a requisition or purchase order. Application developers register the main records and components that make up the transaction, then functional business analysts select the approval transaction on which to base the approval process definition.

---

**Note.** Any PeopleSoft delivered approvals will already have the Approval Transaction Registry populated. No additional configuration is typically needed.

---

---

## Prerequisites

Before defining the transaction registry:

1. Create a Transaction Handler Application class that extends an approved event handler class delivered by Approval Framework.
2. Create transaction data sources, as needed.
3. Create views on transaction tables that will serve as criteria sources.

## Setting Up the Transaction Registry

Use the Register Transactions component to register an approval transaction. The transaction definition is the metadata that describes the transaction setup to the Approval Framework.

### Page Used to Set Up the Transaction Registry

Page Name	Definition Name	Navigation	Usage
Register Transactions	EOAW_TXN	Enterprise Components, Approvals, Approvals, Transaction Registry	Register the approval transaction.

### Registering Approval Transactions

Access the Register Transactions page (Enterprise Components, Approvals, Approvals, Transaction Registry).

### Register Transactions

**Process ID:** CAApapproval  
**\*Description:** Cash Advance Approval Process  
**Object Owner ID:** Expenses  
**\*Cross Reference Table:** EX\_ADV\_AW  
**Worklist Prefix:**

**Notification Options**  
**\*Enable Notifications:** Enable Email and Worklist  
**\*Notification Strategy:** Online Processing  
**Use Email Approvals:** ☒  
**Form Generator Package Root:** EX\_APPRVL EMC  
**Form Generator Class Path:** OUTBOUND:CAFormGenerator

**Internal URL Definition**  
**Internal URL Base:**  
**Internal Portal Name:**  
**Internal Node Name:**

Register Transactions page (1 of 3)

<b>External URL Definitions</b>	
External URL Base:	<input type="text"/>
External Portal Name:	<input type="text"/>
External Node Name:	<input type="text"/>
<b>Default Approval Component</b>	
*Menu Name:	APPROVE_EXPENSE_TRANSACTIONS
*Approval Component:	EX_ADV_APPR
<b>Approval Event Handler Class</b>	
Root Package ID:	EX
Class Path:	Approval:Callbacks:CAApprovalCallBac

Register Transactions page (2 of 3)

<b>Approval Status Monitor</b>			
Adhoc Package:	<input type="text"/>	Adhoc Class:	<input type="text"/>
Thread Package:	<input type="text"/>	Thread Class:	<input type="text"/>
<b>Transaction Approval Levels</b>			
	*Level	*Record (Table) Name	
1	Header	EX_CA_HDR_AW_VW	<input type="button" value="+"/> <input type="button" value="-"/>
2	Line	EX_CA_LN_AW_VW	<input type="button" value="+"/> <input type="button" value="-"/>
<b>Level Record Key Field Label IDs</b>			
	Record (Table) Name	Field Name	*Field Label ID
1	EX_CA_HDR_AW_VW	ADVANCE_ID	ADVANCE_ID
2	EX_CA_LN_AW_VW	ADVANCE_ID	ADVANCE_ID
3	EX_CA_LN_AW_VW	LINE_NBR	LINE_NBR
<a href="#">Expand/Collapse All</a>			

Register Transactions page (3 of 3)

Application developers use this page to register a PeopleSoft application, such as eProcurement or job offer, with the Approval Framework. Using this page, you can define how the system interacts with portions of the application that you have defined for approvals. The transaction definition is the metadata which describes the transaction make up to the approval framework. In some cases, you might add a transaction to enhance an existing transaction or make changes to a transaction.

Use this page to define:

- Worklist approvals.
- Approval event handler class.
- Transaction approval levels.

- Email notifications.
- Ad hoc approver class logic.

<b>Process ID</b>	Enter a name the system uses to track this Approval Framework process for a transaction. You can also enter a description for the approval process.
<b>Object Owner ID</b>	Select the PeopleSoft application to which this object belongs.
<b>Cross Reference Table</b>	<p>Select the table used to manage application specific transaction records and associate them with the approval process run time instances. Each time a request launches an approval process, the system tracks the process by the header and line-level records of the application. To relate the approval process instance to the transaction instance, the cross-reference table holds the correspondence.</p> <p>For a given application transaction record, this cross-reference information helps you determine the pending Approval Framework process and to define to the application which transaction is being approved or denied.</p> <p>Application developers must create a record containing the applications keys, and include the Approval Framework-supplied subrecord EOAW_XREF_SBR. Developers must also build the underlying table.</p>
<b>Worklist Prefix</b>	<p>Enter a prefix to identify the worklist to a specific application. Every application using the Approval Framework shares the same business process and activities, which can cause a problem identifying specific worklist instances. This prefix will allow worklists to be sorted by application. The prefix should be used:</p> <ul style="list-style-type: none"> <li>• If Enterprise Portal is used and users have worklist items from multiple applications. For example Expenses and HCM in the same portal.</li> <li>• If the application creates dashboards, the dashboard needs to be able to differentiate between the different application's approvals.</li> </ul>

### ***Notification Options***

Identify whether you will be using email or worklist notification, or both.

<b>Enable Notifications</b>	<p>Determine what type of notifications your company will use. The options include:</p> <ul style="list-style-type: none"> <li>• <i>Disable Email and Worklist</i></li> <li>• <i>Email Notification Only</i></li> <li>• <i>Enable Email and Worklist</i></li> <li>• <i>Worklist Notification Only</i></li> </ul>
-----------------------------	--

<b>Notification Strategy</b>	It allows the email to be processed immediately ( <i>Online Processing</i> ) or offline ( <i>Offline Processing</i> ) through NEM (Notification and Escalation Manager).
<b>Use Email Approvals</b>	Defines that you are going to use email approvals with workflow.
<b>Form Generator Package Root</b>	This package root reads the threads provided by the Form Generator Class, and creates a form from an existing email collaboration definition.
<b>Form Generator Class Path</b>	Calls the From Generator Class which receives a list of threads to be approved and a language code. This class returns a runtime class, which will add the appropriate recipients and send the emails.

### ***Internal URL Definition***

Use this section to define the internal URL to identify the URL base, portal and node. Internal URL is used to send email to a user either in batch or Integration Broker process. When the process is run online, the built-in properties %Portal and %Node are used.

### ***External URL Definition***

Use this section to define the external URL to identify the URL base, portal and node. External URL is used to notify any step that has a been set to *External*.

### ***Default Approval Component***

Identifies the default component that users should go to when selecting a worklist entry.

<b>Menu Name</b>	Select the menu name that contains the component you want to register for the Worklist.
<b>Approval Component</b>	Select the component on which the approval is going to be based.

### ***Approval Event Handler Class***

Use this section to define the application class used to monitor events for this transaction. Each time an event occurs, the Approval Framework notifies the application. For applications to receive the notifications, application developers must extend the event handler class, ApprovalEventHandler.

When a transaction results in an action from the Approval Framework, the event handler class you specify how to proceed with the transaction.

The event handler base class defines the handler methods that you can override by extending classes. The extending class must have a no-argument constructor, since the system instantiates the class with no arguments.

This table explains some of the various event handler methods for which the system passes arguments to provide the specific context for each event, and gives examples of how an application, in this case PeopleSoft eProcurement, may act:

<b>Event</b>	<b>Parameters</b>	<b>Possible Application Actions</b>
PROCESS_LAUNCHED	<ol style="list-style-type: none"> <li>Header record</li> <li>Approval Process Instance</li> </ol>	<ul style="list-style-type: none"> <li>Disable edits of the application transaction.</li> <li>Display status information.</li> </ul>
HEADER_DENIED	Header record	<ul style="list-style-type: none"> <li>Delete transaction.</li> <li>Disable resubmission.</li> <li>Log the event on the transaction, possibly highlighting previous denial if the system allows resubmission.</li> </ul>
LINE_DENIED	Line-level record	Deduct the line amount from the header, and delete or otherwise deactivate the line item.
HEADER_APPROVED	Header record	<ul style="list-style-type: none"> <li>Source the transaction if it's a requisition.</li> <li>Reimburse the employee if it's an expense report.</li> </ul>
LINE_APPROVED	Line level record	Source the line item if it's configured for sourcing.
PROCESS_TERMINATED	Header record	Log the event on the application, possibly highlighting changes since the previous submission. This might be useful for approvers who acted on the previous submission of this request.
ALL_LINES_PROCESSED	Header record	<p><b>Note.</b> The system calls this method only if the last stage is at the line-level, and the analyst has configured the process to trigger LINE_APPROVED end calls as individual lines are approved.</p> <p>The action the system takes depends on how the application developer defines the line sourcing. If the lines are sourced as they are approved, then nothing has to be done when all the lines are processed. This event is distinct from HEADER_APPROVED, and having a distinct notification simplifies the process.</p>

**Root Package ID**

Select the parent application class through which events are exposed. This defines the action to take based on events.

**Path**

Select a path that uses a specific class within the root package.

See *PeopleTools: PeopleCode Developer's Guide PeopleBook*

See and [Chapter 11, "Using Approval Framework Base Classes," page 107.](#)

**Approval Status Monitor**

Use this section to control how the system processes ad hoc approvers. Any approver can add or remove ad hoc approvers.



<b>Adhoc Package</b>	Select the ad hoc application class package that you want to use for ad hoc approvals.
<b>Adhoc Class</b>	Select the ad hoc application class path. Adding approvers and reviewers is handled by the class you define here. If no class is specified, then the system default class is used. If the transaction has further restrictions an application developer needs to create a class that will be defined here.
<b>Thread Package</b>	<p>The package here defines how the transaction details are displayed in the system in the status monitor. Behind the scene approvals are defined with a sequence number, this allows for a user friends display.</p> <hr/> <p><b>Important!</b> Leave this field blank if the transaction is set up on the Configure Transactions page to send notifications to participants with a URL pointing to an alternate page other than the default destination (as specified in the Events section). For example, you might configure the transaction to generate a URL within each notification that takes an approver to the default component page to approve or deny any given request, and generate another URL for other participants to access a separate page (as specified for that group of users in the Notifications section) and perform other actions pertaining to the request. To ensure that the URLs are created correctly for all approval participants, do not enter a thread package.</p> <hr/>
<b>Thread Class</b>	Enter the specific class within the thread description package and the worklist description that sets the display details.

### ***Transaction Approval Levels***

Use this section to define if the transaction is to be approved at the header or line level and what level the application supports.

<b>Level</b>	Select <i>Header</i> or <i>Line</i> , which determines the levels that are enabled by the application for approvals. The first row will always be the header level.
<b>Record (Table) Name</b>	Select the database table that represents this transaction level.
<b>Level Record Key Field Label IDs</b>	Use this section to identify the label IDs that are used when viewing the Monitor Approvals page. All key field names appear for each record (table) name that is listed.

---

## **Configuring Approval Transactions**

Use the Configure Transactions page to select and define elements that determine what triggers a notification, who receives the notification, and the content of the notification. Notifications are mapped to work with the approval transaction registry and include menus and components and SQL definitions.

## Page Used to Configure Approval Transactions

<i>Page Name</i>	<i>Definition Name</i>	<i>Navigation</i>	<i>Usage</i>
Configure Transactions	EOAW_TXN_NOTIFY	Enterprise Components, Approvals, Approvals, Transaction Configuration	Use the Configuration Transactions page to configure how the system uses the particular implementation of approval triggers.

## Configuring Approval Transactions

Access the Configure Transactions page (Enterprise Components, Approvals, Approvals, Transaction Configuration).

## Configure Transactions

**Process ID:** CAApapproval

**Ad Hoc Approver Options**

**\*Approval User Info View:**

**Ad Hoc User List:**

**Notification Options**

☐ **Send Email Approvals to All**

**Email Approval User List:**

**\*Delivery Method:**

☐ **Perform Sent-To Security Check**

**User Utilities**

**User Utilities Package:**

**User Utilities Path:**

**Events**
Find | View All | First 1 of 6 Last

**\*Event:**  **\*Level:**

**Menu Name:**

**Approval Component:**

**Page Name:**

**Menu Action:**

**SQL Object Identifier:**

**Notifications**
Customize | Find | View All | First 1 of 1 Last

Main
Template Details
Frequency

	*Participant	Channel	User List	Template Name		
1	<input type="text" value="User List"/>	<input type="text" value="Email"/>	<input type="text" value="Expense Submitter ID"/>	<input type="text" value="Cash Advance Approved"/>		

### Configure Transactions page

Use this page to select and define elements that determine what triggers a notification, who receives the notification, and the content of the notification. Notifications are mapped to work with the approval transaction registry and include menus and components and SQL definitions. The events for which the system sends notifications include:

- Launch of the approval process on a transaction.
- Queue of approval step to an approver.
- Queue of a review step to a reviewer.
- Denial of a line or header.

- Approval of a line or header.
- Completion of the approval process.

Recipients of notifications include requesters, approvers, and reviewers, who can receive their notifications through either worklist entries or email notification. When using email notifications, business analysts must create email templates.

### ***Ad Hoc Approver Options***

**Approver User Info View** Provides details about which view a user sees when using the Approval Monitor.

---

**Note.** Data in this view dictates what is displayed in the approver links.

---

**Ad Hoc User List** This is a filter used to display only a list of users who can be ad hoc approvers.

### ***Notification Options***

This section appears when the Use Email Approvals check box is selected for the Process ID on the Register Transactions page.

**Send Email Approvals to All** Select to send email notifications to all approvers.

**Email Approval User List** Specify exactly which users should be allowed to do their approval by using email.

---

**Note.** If the user receiving the notification also falls into the email approval user list, then they receive an email approval rather than a standard email notification.

---

**Delivery Method** Define whether you wish the users to receive their email approvals as text within the email, or as attachments.

**Perform Sent-To Security Check** Selecting this check box informs the system that you want it to verify the security of the person the notification is sent to.

---

**Note.** This security check is only performed on new approvals.

---

### ***User Utilities***

User Utilities are the mechanism that the user changes to modify the behavior of delegation and reassignment.

**User Utilities Package** Select the parent application class through which alternate users are selected.

**User Utilities Path** Select a path that uses a specific class within the root package.

**Events**

Use the events section to define event parameters to trigger workflow notification.

**Level**

Select *Header* or *Line* to determine the level at which you want a notification sent for an event. For each of these events to be notified, you must select the level of the transaction.

**Event**

Select the event for which you want to send a notification. Participants will be notified when the selected event is triggered.

Event values include:

*Ad Hoc Delete:* Send a notification when a step is removed from the approval process through the status monitor.

*Ad Hoc Insert:* Send a notification when a step is added to the approval process through the status monitor.

*Hold Step:* Send a notification when a thread is placed on hold.

*Locked Out:* Send a notification when the application workflow engine (AWE) encounters a user whose account has been locked out.

*No Approver Necessary:* Send a notification when a thread does not route because no approval is necessary.

*On Error:* Send a notification when a thread encounters a routing error.

*On Escalate:* Send a notification when a thread is escalated through the NEM process.

*On Final Approval:* Send a notification when a thread is approved and there are no more steps to process.

*On Final Denial:* Send a notification when a thread is denied and there are no more steps to process.

*On Process Launch:* Send a notification when an approval process is submitted.

*On Reactivate:* Send a notification when a step is reactivated for a thread.

*On Reassign:* Send a notification when a thread is reassigned to a new approver.

*On Step Complete:* Send a notification when a step no longer has pending threads.

*On Terminate:* Send a notification when a thread is terminated.

*Processing Complete:* Send a notification when an approval process is complete.

*Push Back:* Send a notification when a thread is pushed back from one step to the prior step.

*Request Information:* Send a notification when the Request Information action is called by the application.

*Request Information Added:* Send a notification when a comment is added for a thread and that thread has been placed on hold.

*Route for Approval:* Send a notification when a thread is routed to an approver.

*Route for Review:* Send a notification when a thread is routed to a reviewer.

---

**Note.** The *Lock Out*, *On Process Launch*, and *Processing Complete* events are for header level only.

---

<b>Menu Name</b>	Select the menu name that contains the component you want the notification recipient to link to. This identifies where the person should go upon notification. If you do not enter values, the recipient is sent to the same menu and component that is defined for the Worklist Approval component.
<b>Approval Component</b>	Select the component you want to make available to the notification recipient.
<b>Page Name</b>	The page defined is the page approvers are redirected to from the URL sent within the email notification.
<b>Menu Action</b>	This is the action of the page users see when directed to the page from the URL sent within the email notification.
<b>SQL Object Identifier</b> (structured query language object identifier)	Select the SQL definition identifier you want to use to get content for the email. The SQL must accept bind inputs equal to the number of keys at the notification level. For example, header or line keys.

## Notifications

Use the Notifications section to define who to notify and how to notify them in addition to the defaults determined in the Events section of this page.

<b>Participant</b>	<p>Define the user who is notified when this event takes place.</p> <ul style="list-style-type: none"> <li>• <i>A-Delegate</i>: the approver that the approval was originally assigned to.</li> <li>• <i>A-Proxy</i>: the approver who performed the actual approval.</li> <li>• <i>Admin</i></li> <li>• <i>Approvers</i></li> <li>• <i>Dynamic</i></li> <li>• <i>External</i></li> <li>• <i>R-Delegate</i>: the person who created the request for someone else.</li> <li>• <i>R-Proxy</i>: the person who requested the transaction to be created.</li> <li>• <i>Requester</i></li> <li>• <i>Reviewers</i></li> <li>• <i>User List</i></li> </ul>
--------------------	---

<b>Channel</b>	<p>Defines how the participant will be notified. Values are:</p> <ul style="list-style-type: none"> <li>• <i>Both</i></li> <li>• <i>Email</i></li> <li>• <i>None</i></li> <li>• <i>User</i></li> <li>• <i>Worklist</i></li> </ul> <hr/> <p><b>Note.</b> Routing preferences can also be set up in PeopleTools, Security, User Profiles, Workflow. From there you have two options. You can select <i>Worklist User</i> and or <i>Email User</i>.</p> <hr/>
<b>User List</b>	Select either <i>Dynamic</i> or <i>User List</i> as the participant. The option becomes active when you select one of these values.
<b>Template Name</b>	<p>Select the generic template you want to use for the email content of this notification. You define the contents of the email using the Generic Template page.</p> <p>See and <a href="#">Chapter 5, "Defining Notification Templates and Users for Approval Framework," Entering Generic Template Definitions, page 51.</a></p>
<b>Menu Name, Approval Comment, Page Name, Menu Action, and SQL Object Identifier</b>	All of these fields have the same definition as the corresponding fields in the Events section of this page.
<b>Number of Hours</b>	Enter a number that determines how many hours between notifications.
<b>Max Notification</b> (maximum notification)	Enter a number that determines the maximum number of notifications sent. If the approver does not take action, an escalation is sent to the administrator.



## Chapter 5

# Defining Notification Templates and Users for Approval Framework

This chapter discusses how to:

- Define notification templates for Approval Framework.
- Define users for Approval Framework.

---

## Defining Notification Templates for Approval Framework

This section discusses how to enter generic template definitions.

### Pages Used to Define Notification Templates for Approval Framework

<i>Page Name</i>	<i>Definition Name</i>	<i>Navigation</i>	<i>Usage</i>
Generic Template Definition	WL_TEMPLATE_GEN	PeopleTools, Workflow, Notifications, Generic Templates	Enter generic template definitions.
URL Maintenance	URL_TABLE	PeopleTools, Utilities, Administration, URLs	<p>Use this page to identify the URL that the notification process places within the email. The user then uses this URL to navigate back into their system to perform the required task.</p> <p><b>Note.</b> An example of the format to use is <i>http://servername/psp/employeeportaldomain/.</i></p>

### Entering Generic Template Definitions

Access the Generic Template Definition page (PeopleTools, Workflow, Notifications, Generic Templates).

Generic Template Definition

Blackberry Email Responses

Template:

Cash Advance Approved

\*Description:

Cash advance approved

Instructional Text:

Type names or email addresses in the To, CC, or BCC fields, using a semi-colon as a separator. Click LOOKUP RECIPIENT to search for a name. Click DELIVERY OPTIONS to view or change the

Priority:

2-Medium

\*Sender:

User

Email ID:

Subject:

Cash advance for %8 %9 has been approved.

Message Text:

The following cash advance has been approved:

Employee ID: %2

Employee Name: %3

Submission Date: %4

Advance Description: %6

Advance ID: %5

Business Purpose: %7

Below is the list of available variables for this template.

You can use template variables within your subject or message text.

The following variables can also be used:

%Date, %DateTime, %Time, %ServerTimeZone, %EmailAddress, %NotificationPriority, %NotificationToList, %NotificationCCList

Template Variables

*Value	*Description		
%1	URL	+	-
%2	EMPLID	+	-

Generic Template Definition page

You use generic templates to establish common formats for ad hoc notifications.

For approvals, the first bind variable is used to store the URL that appears in the email.

### See Also

*PeopleTools: Workflow Technology PeopleBook*, Using Notification Templates

## Defining Users for Approval Framework

This section discusses how to:

- Attach workflow roles to users.
- Define workflow for user profiles.
- Define user lists.

## Pages Used to Define Users for Approval Framework

Page Name	Definition Name	Navigation	Usage
User Profiles - Roles	USER_ROLES	PeopleTools, Security, User Profiles, User Profiles, Roles	Attach workflow roles to users.
User Profiles - Workflow	USER_WORKFLOW	PeopleTools, Security, User Profiles, User Profiles, Workflow	Define workflow for user profiles.
User List Definition	EOAW_USER_LIST	Enterprise Components, Approvals, Approvals, User List Setup	Define user lists.

## Attaching Workflow Roles to Users

Access the User Profiles - Roles page (PeopleTools, Security, User Profiles, User Profiles, Roles).

The screenshot shows the 'User Profiles - Roles' page. At the top, there are tabs: General, ID, Roles, Workflow, Audit, Links, and User ID Queries. The 'Roles' tab is selected. Below the tabs, the 'User ID' is 'BIS1' and the 'Description' is 'Wood,Nathan'. On the left side, there is a 'Dynamic Role Rule' section with a search box, a 'Test Rule(s)' button, a 'Refresh' button, an 'Execute Rule(s)' button, and links for 'Process Monitor' and 'Service Monitor'. The main part of the page is a table titled 'User Roles'. The table has columns: Role Name, Description, Dynamic, Route Control, and View Definition. The table lists several roles: Billing Manager, EOPP\_USER, PAPP\_USER, PeopleSoft User, SUPERVISOR, Sales Manager, and UPG\_BIPNLS. Each role has a 'Route Control' link and a 'View Definition' link. There are also '+' and '-' buttons at the end of each row.

### User Profiles - Roles page

Use this page to attach workflow roles to users. A role is a class of users who perform the same type of work, such as clerks, buyers, or managers. A role describes how people fit into workflow.

Role user IDs determine how to route worklist items to users and how to track the roles that users play in the workflow.

<b>Role Name</b>	Select a role to assign to this user. Role users are the people who participate in automated business processes.
<b>Dynamic</b>	<p>Appears if the definition of this role is dynamic. This value is driven by PeopleSoft PeopleCode. Dynamic users can obtain membership in a role programmatically. You can run a batch process that executes predefined role rules and assigns roles to user profiles according to these rules. This approach is called dynamic membership, and users who become role users of a particular role programmatically are dynamic role users.</p> <p>Static role users, on the other hand, obtain their membership through an administrator adding a role to their user profile manually.</p>
<b>Route Control</b>	<p>Select to access the User Route Control Profiles page, where you can select a route control profile for the workflow.</p> <p>The PeopleSoft Workflow Administrator enables you to define route controls. For example, suppose you want to route requisitions to different buyers, depending on which vendor supplies the ordered items, which business unit is requesting the items, and which department needs the items. You can define a route control for each factor—vendor ID, business unit, and department—and specify a range of values for each buyer.</p> <hr/> <p><b>Note.</b> Not used with eProcurement Approval Framework.</p> <hr/>
<b>View Definition</b>	Select to access the Roles - General page, where you can change the role name definition. You can also view the user ID of the role member to ensure that you selected the appropriate definition for inclusion in the role.
<b>See Also</b>	<i>PeopleTools: Security Administration PeopleBook</i> , Administering User Profiles

## Defining Workflow for User Profiles

Access the User Profiles - Workflow page (PeopleTools, Security, User Profiles, User Profiles, Workflow).

General	ID	Roles	Workflow	Audit	Links	User ID Queries
User ID: BIS1 Description: Wood,Nathan						
Workflow Attributes						
Alternate User ID:		<input type="text"/>		<div>Routing Preferences</div> <input checked="" type="checkbox"/> Worklist User <input checked="" type="checkbox"/> Email User		
From Date:		<input type="text"/>				
To Date:		<input type="text"/>				
Supervising User ID:		<input type="text" value="BIVP1"/>				
Reassign Work						
<input type="checkbox"/> Reassign Work To:		<input type="text"/>				
Total Pending Worklist Entries:				0		

### User Profiles - Workflow page

Use this page to define alternate users who are part of the workflow process. You can define alternate users to handle approvals during the absence of the primary approver and supervisor.

#### Alternate User ID

Select a user to receive worklist items when this user ID is temporarily unavailable. In situations where the current role user is unavailable, the system automatically forwards the current role user's new work items to the specified alternate role user. The system doesn't reassign items already in the user's worklist.

**Note.** For workflows that are built using the Approval Framework, if the current role user is unavailable for assignment, the system checks for a alternate user recursively until it either creates a cycle or finds a valid substitute. In other words, the system sends routings down multiple levels of alternate user IDs if necessary. For example, suppose that user A specifies user B as his alternate user, and user B specifies user C as his alternate user. If both user A and user B are out of the office the same time, the system sends workflow routings originally intended for user A to user C.

#### From Date and To Date

Enter a date range when the current user ID is not going to be available. The system uses these values to forward routings to the alternate user.

#### Supervising User ID

Select the user ID of this user's supervisor. The system uses this value when forwarding information to the user's supervisor and uses the PERSONAL\_DATA record to determine the supervisor.

**Note.** If you're using PeopleSoft Human Capital Management (PeopleSoft HCM) applications, this field shouldn't appear. If it does, you must set your workflow system defaults.

- Worklist User

Select to indicate that this role user can receive approval routings. Clear the check box if the user is not a PeopleSoft user. You can select Worklist User, Email User, or both.
- Email User

Select to specify that this role user can receive email. Clear the check box if email is not available.
- Reassign Work To

This option is not used in Approval Framework. It will reassign the worklist, but not the underlying data in Approval Framework.

Defining User Lists

Access the User List Definition page (Enterprise Components, Approvals, Approvals, User List Setup).

User List Definition

User List:

Cash Advance Approvers

\*Description:

Cash Advance Approver List

User List Source

☐ Role

☐ SQL Definition

☐ Query

☒ Application Class

Root Package ID:

EX

Application Class Path:

Approval>List:CAApproverList

☒ Include Users as Input

☒ Transaction Keys as Input

User List Attributes

Customize | Find | First 1 of 1 Last

	Attribute Name	Attribute 1	Attribute 2		
1				+	-

User List Definition page

Use this page to define user sources for use with steps in the approval process. The PeopleSoft product delivers a set of default user list roles corresponding to the roles within an organization.

When you select a user list source type, you must also select a corresponding value. You can add a new user list or change a current list.

**Note.** You can only select one user list source for a user list.

---

**Note.** The SQL Definition, Query, and Application Class user list sources are dynamic, and can use input values to help identify users.

---

<b>Role</b>	Select to use a role as the source for this user list. A role is a list of users who perform the same type of work, such as buyers or managers. Each role has a set of parameters that determine the limits of the role in the organization and in the workflow process.
<b>SQL Definition</b> (structured query language definition)	Select to use an SQL definition as the source for this user list. The SQL must return OPRID field.
<b>Query</b>	Select to use a query as the source for this user list. When a source is defined as a query, the system determines who should receive a work item based on the field values on the page that triggers the routing.
<b>Application Class</b>	Select to use an application class as the source for this user list.  When you select an application class, the system passes the originator of the transaction and then determines the approver for that originator. For subsequent approval steps, the system passes the approver from the previous step.
<b>Include Users as Input</b>	Select to indicate that the system uses the originator of a transaction as the first step in each path. For subsequent steps in each path, the system uses the approver from the previous step.  This field is not available when the User List Source is User List.
<b>Transaction Keys as Input</b>	Select to have the system base the approval routing on transaction keys. Transaction keys are key fields in a database table. System actions depend on the approval level at which a user list is being used. If the approval is at the header level, the system uses transaction record keys from the header record.  This field is not available when the User List Source is User List.
<b>User List Attributes</b>	This section is only available when the User List Source is Application Class.

---

**Note.** UserList Attributes are used for application class User Lists and Route Control is used for role based user lists.

---

### ***Route Attributes***

If User List Source is Role, the Route Attributes section is displayed. This option enables the Route Control filters.

**Route Control Profile**                      Select the route control profile to use.

<b>Record Name</b>	Record the user passes in that stores the values used in the route control. <hr/> <b>Note.</b> Route control checks against fields, Approval Framework needs to know which record contains the field. <hr/>
<b>Route Control Type</b>	Route control attribute. <hr/> <b>Note.</b> Route control attribute is defined on the User Profile Roles page in the route control link. <hr/>
<b>Field Name</b>	Field on the record mapped to the attribute.

**See Also**

*PeopleTools: PeopleCode Developer's Guide PeopleBook*



## Chapter 6

# Defining Dynamic Approvals

This section provides overviews of dynamic paths and dynamic approval authorizations and discusses how to:

- Define user lists for dynamic authorizations.
- Set up approval authorizations.
- Define dynamic approval paths.

---

## Understanding Dynamic Paths

To use dynamic approvals, create one approval step that determines a list of approvers without setting up every step individually within the path.

Workflow processes are defined in stages, paths, and steps. The system looks at the stage to determine if the trigger for the workflow is recognized at the header or line level. Within each stage, there is a minimum requirement of one path. The path contains steps, which define the workflow triggers and the action to take if the criteria is met. Without dynamic paths, the administrator creates a step for every possible approver. With dynamic workflow, the administrator creates a single path where the system uses a user list for the approval hierarchy.

---

**Note.** When self-approval is used and the transaction creator is on the list of authorized approvers, that role is counted as one approval.

---

### See Also

Chapter 5, "Defining Notification Templates and Users for Approval Framework," Defining User Lists, page 56

---

## Understanding Dynamic Approval Authorizations

PeopleSoft applications can define Approval Framework paths to be either static or dynamic. Static path approvals define every approval in step-by-step fashion. Dynamic approvals enable you to create a single step that systematically identifies every potential approver, searches to find out if that approver has enough authority to complete the approval, and creates a visual path for users to view of all necessary approvers in the process. You can configure a dynamic path allowing supervisor roles to approve or deny a transaction, and stop the approval path when the system has determined that all criteria has been satisfied. The administrator creates a user list that the system uses to select the appropriate supervisory approvers for a transaction and then checks for authorization. The dynamic path takes the prior approver into consideration.

To configure the dynamic approval authorization, the administrator must:

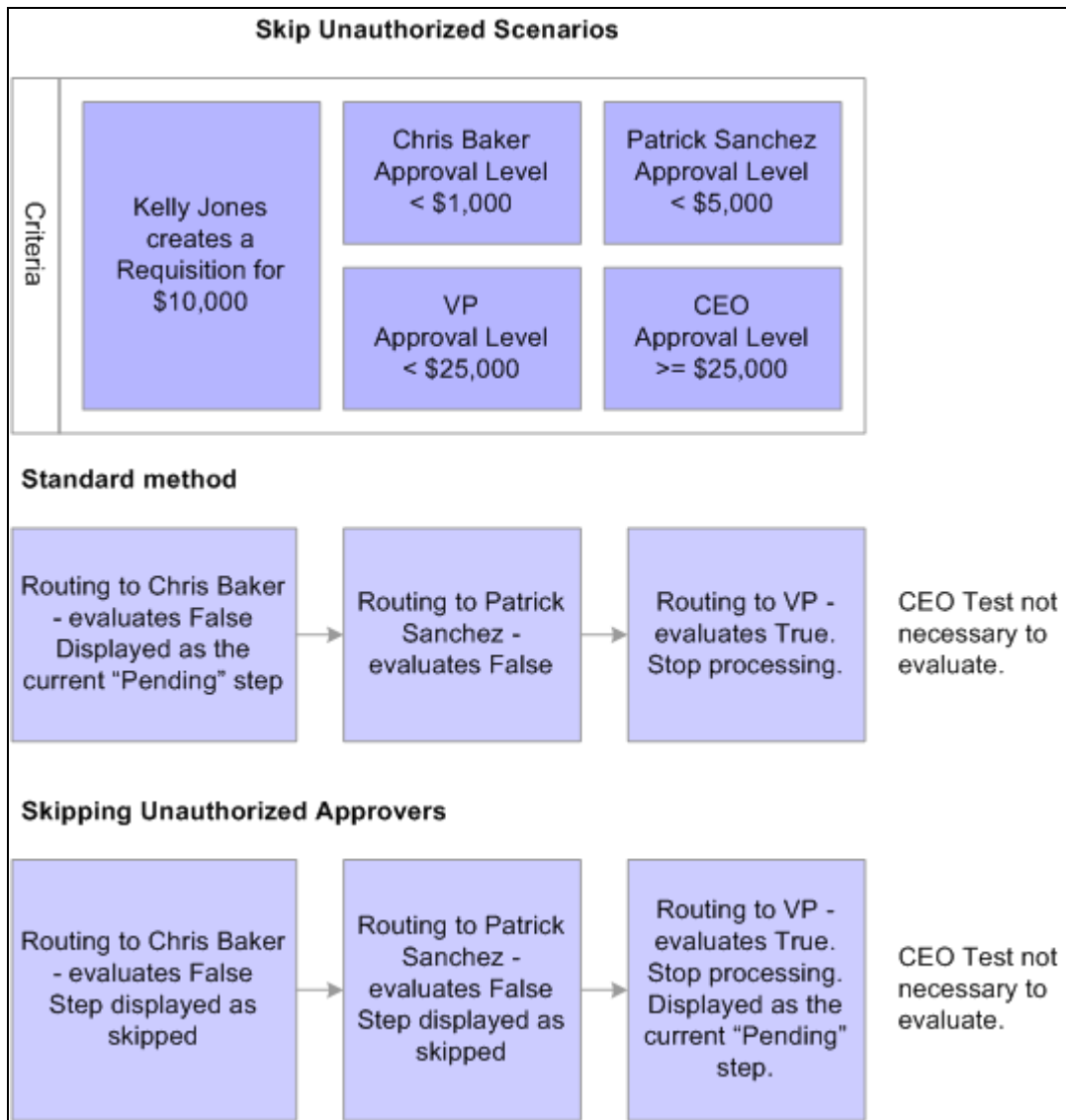
- Define user lists.
- Create an approver authorization.
- Define a dynamic approval path.

Two keys to creating approval authorizations for dynamic paths are the system's ability to:

- Check authorizations.
- Skip unauthorized users.

The system looks at the user list and the approval authorization to determine which users are required to complete the authorization. The system displays the non-required users as a skipped step instead of a pending step in the event that Skip Unauthorized is selected.

This diagram illustrates an example of a workflow routing setup for the standard method and a workflow routing that skips unauthorized users:



Example of Skip Approval scenario

In the example, the criteria for the workflow approval path is set up for Chris Baker to have approval authority for less than 1,000.00 USD, Patrick Sanchez to have authority for less than 5,000.00 USD, VP to have approval for less than 25,000.00 USD, and CEO to have approval for a requisition equal to or more than 25,000.00 USD.

Kelly Jones creates a requisition for 10,000.00 USD.

If the system is not set up to skip unauthorized users, the system displays Chris Baker, Patrick Sanchez, and the VP as pending steps in the approval path.

If the system is set up to skip unauthorized users, the system displays the approval path with the VP as the only listed approver, and will display Chris Baker and Patrick Sanchez as skipped.

**See Also**

Chapter 5, "Defining Notification Templates and Users for Approval Framework," Defining User Lists, page 56

---

## Understanding Approval Authorizations

You can identify the approval authorization by role or user in conjunction with a dynamic step. To accomplish this, the Approval Framework selects the appropriate supervisory approver from the user list and verifies that the approver meets the criteria for authorization.

You establish approval authorizations for each transaction. The authorization can accommodate approvals by role or user ID.

You can set authorization across Definition IDs, which are defined on the Setup Process Definition page.

For each authorization, the system checks the specific user ID to see if that individual can authorize the transaction. If found, it checks the authorization criteria. If criteria are met, the user has authorization.

If no authorization is found for a specific user ID, then the system looks for role-based authorizations using the approval hierarchy.

For approval hierarchy, the system first looks for authorization by Definition ID. If no authorization is found, the system then seeks authorization for rows without a Definition ID. If no authorization approval criteria is matched, the system process is deemed Not Authorized.

You can establish dynamic authorizations for either the header or line level, but not both.

When workflow is initiated for a change order or requisition, the system compares the approval authorization data to the user list to verify the approval process. To verify the approval, the system:

1. Checks the user list and assigns the first approver to the first user that is returned.
2. Looks at the roles that are established for the user ID.
3. Identifies the approval limits that are set for that user ID.
4. Routes the requisition status to the first approver if the amount is satisfied for the requisition and the approver list is complete.
5. Continues to look for additional approvers until all conditions are met.
6. Routes the approval to the administrator if the approver criteria is never met.

**See Also**

Chapter 5, "Defining Notification Templates and Users for Approval Framework," Defining User Lists, page 56

## Defining Dynamic Approvals

This section includes:

- Define user lists for dynamic authorizations.
- Setting up approval authorizations.
- Define dynamic approval paths.

## Pages Used to Define Dynamic Approvals

<i>Page Name</i>	<i>Definition Name</i>	<i>Navigation</i>	<i>Usage</i>
Approval Authorization	EOAW_AUTH	Enterprise Components, Approvals, Approvals, Authorize Approvers	Authorize roles and approvers for dynamic paths.
Criteria Definition	EOAW_CRITERIA	Click the Criteria link on the Approval Authorization page.	Define criteria for the workflow approver.
Setup Process Definitions	EOAW_PRCES_MAIN	Enterprise Components, Approvals, Approvals, Approval Process Setup	Define approval process stages.
Approval Path Definition	EOAW_PATH_SEC	Click the Path Details button or the Details link on the Setup Process Definitions page.	Set up workflow approval paths.
User List Definition	EOAW_USER_LIST	Enterprise Components, Approvals, Approvals, User List Setup	Set up list of users for workflow approval.
User Profile - General	USER_GENERAL	PeopleTools, Security, User Profiles, User Profiles, General	Set up user IDs and assign roles.

## Defining User Lists for Dynamic Authorizations

Access the User List Definition page (Enterprise Components, Approvals, Approvals, User List Setup).

User List Definition

User List:Supervisor by UserId

\*Description:Supervisor by UserID

User List Source

Role

SQL Definition

Query

Application Class

SQL Object Identifier:EOAW\_SUPERVISOR\_BY\_OPRID

☒ Include Users as Input

☐ Transaction Keys as Input

User List Definition page

See Also

Chapter 5, "Defining Notification Templates and Users for Approval Framework," Defining User Lists, page 56

Setting Up Approval Authorizations

Access the Approval Authorization page (Enterprise Components, Approvals, Approvals, Authorize Approvers).

### Approval Authorization

Process ID: Credit Invoice

Approval Authorization Find | View All | First 1 of 1 Last

Effective Date: 01/01/1900 Status as of Effective Date: Active

Criteria Definition Customize | Find | View All | First 1-4 of 4 Last

Definition ID	Role Name	User ID				
Credit Invoic	Billing Manager		<a href="#">Criteria</a>	<a href="#">Self-Approval Criteria</a>		
Credit Invoic	Billing Specialist		<a href="#">Criteria</a>	<a href="#">Self-Approval Criteria</a>		
Credit Invoic	CFO		<a href="#">Criteria</a>	<a href="#">Self-Approval Criteria</a>		
Credit Invoic	VICE PRESIDENT		<a href="#">Criteria</a>	<a href="#">Self-Approval Criteria</a>		

Approval Authorization page

If you don't specify a Definition ID, the authorization is generic. To create an approval authorization for specific definition IDs, you must add a line for each Definition ID.

Select either a role name or User ID. For each role or user, you can configure the criteria and self-approval criteria, using the links provided.

---

**Note.** If you activate self-approval on the Approval Authorization page, it replaces the self-approval on static path steps.

---

## Defining Dynamic Approval Paths

Access the Approval Path Definition page (click the Path Details button or the Details link on the Setup Process Definitions page).

### Approval Path Definition

Criteria

Approval Path: 1

\*Step Source: 

Dynamic

Description: Chain of Supervisros

Long Description:

☐ Notify Admin on No Approvers

☐ Skip Prior Steps for Requester

☒ Check Authorization

☐ Skip Unauthorized Users

Timeout Options

Customize | Find | First 1 of 1 Last

	*Escalate Option	Hours	Days	Reassign To	User List	Use Proxy	
1	<div>Notify Participant</div>				<div></div>	<input type="checkbox"/>	<div>+ -</div>

Approval Path Definition page

- Step Source

Select *Dynamic* for a dynamic approval path.
- Notify Admin on No Approvers (notify administrator on no approvers)

Select to indicate that the administrator is to be notified if the system does not find an approver for the path. This option is only available when the step source is *Dynamic*.
- Skip Prior Steps for Requester

Select to indicate that if one of the approvers in this path is also the requester, then the system is to skip all steps in the path prior to that approver's step.
- Check Authorization

Select to enable the approval authorization. The data set up in Authorize Approvers component is used.
- Skip Unauthorized Users

Select to enable the approval process to skip users within the user list if the system determines that they can't satisfy all of the criteria for approval.

**Note.** When creating criteria within the path that will trigger the approval process to activate, be certain that you set up the final approver as *Greater Than* so that no gaps occur.



## Chapter 7

# Using Email Collaboration

This chapter provides an overview of email collaboration and discusses how to:

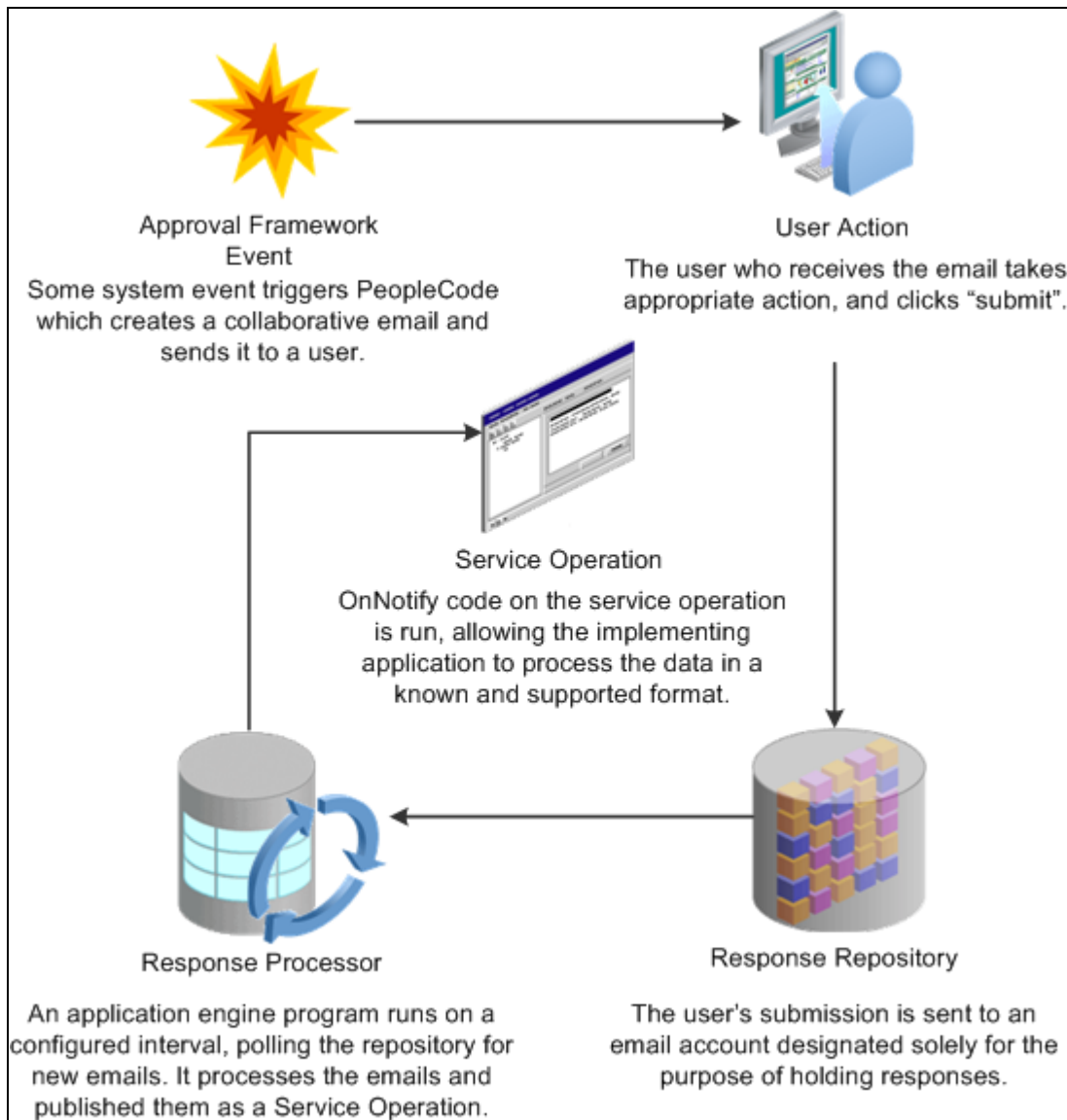
- Set up the PSFT\_EMG\_GETMAIL node.
- Define Integration Broker message and service operation.
- Define and map EMC forms.
- Trigger email collaboration.

---

## Understanding Email Collaboration

The Email Collaboration Framework (EMC) allows applications to send, receive, and process emails with interactive content. You can send an HTML form to a user, and they do not need to log into their system to perform tasks.

This diagram shows the email collaboration flow:



#### Email Collaboration flow

1. A system event triggers PeopleSoft PeopleCode, which creates a collaborative email and sends it to a user.
2. The user who receives the email takes appropriate action and clicks Submit.
3. The user's submission is sent to an email account that is designated for holding responses.
4. An application engine program runs on a configured interval, polling the repository for new emails. It processes the emails and publishes them as service operations.
5. The service operation runs, allowing the implementing application to process the data in a known and supported format.

## Email Collaboration Support Matrix

This table covers the general email clients supported for email collaboration.

<b>Client</b>	<b>Inline Support</b>	<b>Attachment Support</b>	<b>Enhanced Attachment Support</b>
Outlook 2003 <b>Note.</b> Outlook 2003 uses Internet Explorer to render the HTML message.	Y	Y	Y
Outlook 2007 <b>Note.</b> Outlook 2007 uses an internal tool to render the HTML. The actionable fields (buttons, check boxes, and so on) are rendered as text.	N	Y	Y
Lotus Note	Y	Y	Y
Thunderbird	Y	Y	Y
Hotmail <b>Note.</b> The HTML actions are controlled by the Hotmail Client. Since Hotmail is a web client, they circumvent the form submission calls that are in the HTML client. The user may be required to download the attachment prior to opening.	N	Y	Y
Gmail <b>Note.</b> Similar to Hotmail.	N	Y	Y
Yahoo <b>Note.</b> Similar to Hotmail.	N	Y	Y
Internet Explorer (IE) <b>Note.</b> Internet Explorer is listed because it responds differently based on the version of Outlook. IE 7 does not support the basic attachment when used with Outlook 2007. The email is created, but the body is not populated. If using IE 6, there is no issue.	N	Y	Y

<b>Client</b>	<b>Inline Support</b>	<b>Attachment Support</b>	<b>Enhanced Attachment Support</b>
FireFox  <b>Note.</b> Listed to compare to Internet Explorer. No issues have been reported with the way the email is populated using either attachment type.	N	Y	Y

### **Setting Up Email Collaboration**

To set up email collaboration, you need to:

- Set up the PSFT\_EMG\_GETMAIL node.
- Create Integration Broker message.
- Create Integration Broker service operation.
- Define EMC forms, EMC Layout and fields mapping.
- Update an Approval Process notification option.
- Schedule the application engine program EOAWEMC.

---

## **Setting Up the PSFT\_EMG\_GETMAIL Node**

Email collaboration uses the Integration Broker framework to retrieve new emails from the response repository. The node PSFT\_EMG\_GETMAIL is used to retrieve these emails. Integration Broker must be configured and pub/sub must be enabled.

See *PeopleTools: PeopleSoft Integration Broker Administration PeopleBook*, Administering Messaging Servers for Asynchronous Messaging.

### **Pages used to Set Up the PSFT\_EMG\_GETMAIL Node**

<b>Page Name</b>	<b>Definition Name</b>	<b>Navigation</b>	<b>Usage</b>
Node Definitions	IB_NODE	PeopleTools, Integration Broker, Integration Setup, Nodes	Set up the PSFT_EMG_GETMAIL node.
Node Properties	IB_NODEPROP	Click the Properties link on the Node Definitions page.	Modify the node properties required for email collaboration to function.

## Setting Up the PSFT\_EMG\_GETMAIL Node

To set up the PSFT\_EMG\_GETMAIL node:

1. Access the Node Definitions page (PeopleTools, Integration Broker, Integration Setup, Nodes), activate the *PSFT\_EMG\_GETMAIL* node.
2. Click the Properties link to access the Node Properties page.
3. Enter the email address of the email repository in the EMC\_REPOSITORY\_EMAILADDRESS property.
4. (Optional) On the Node Properties page, enter appropriate values for EMC\_BCC\_LIST and EMC\_CC\_LIST properties to automatically send a copy of all collaborative emails to the address specified.
5. Access the Connector page.

This node uses the GETMAILTARGET target connector. You will need to configure the GETMAILTARGET properties.

See *PeopleTools: PeopleSoft MultiChannel Framework*, Configuring the Email Channel, Configuring PeopleSoft Integration Broker for the Email Channel.

6. Save the node.

---

## Defining Message and Service Operation

This section provides discusses how to:

- Define Integration Broker message.
- Define service operation.

## Pages Used to Define Message and Service Operation

<i>Page Name</i>	<i>Definition Name</i>	<i>Navigation</i>	<i>Usage</i>
Message Definition	IB_MESSAGE_BUILDER	PeopleTools, Integration Broker, Integration Setup, Messages, Message Definition	Define a message.
Services	IB_SERVICEDEFN	PeopleTools, Integration Broker, Integration Setup, Services, Services	Define or update a service.
Service Operation Definition - General	IB_SERVICE	PeopleTools, Integration Broker, Integration Setup, Service Operations, General	Define or update a service operation.

<b>Page Name</b>	<b>Definition Name</b>	<b>Navigation</b>	<b>Usage</b>
Service Operation Definition - Handler	IB_SERVICEHDLR	PeopleTools, Integration Broker, Integration Setup, Service Operations, Handler	Specify handler name, the handler type, and the implementation method for the handler.
Handler Details	IB_SERVICEHDLR_SEC	Click the Details link on the Service Operation Definition - Handler page.	Specify the handler details.

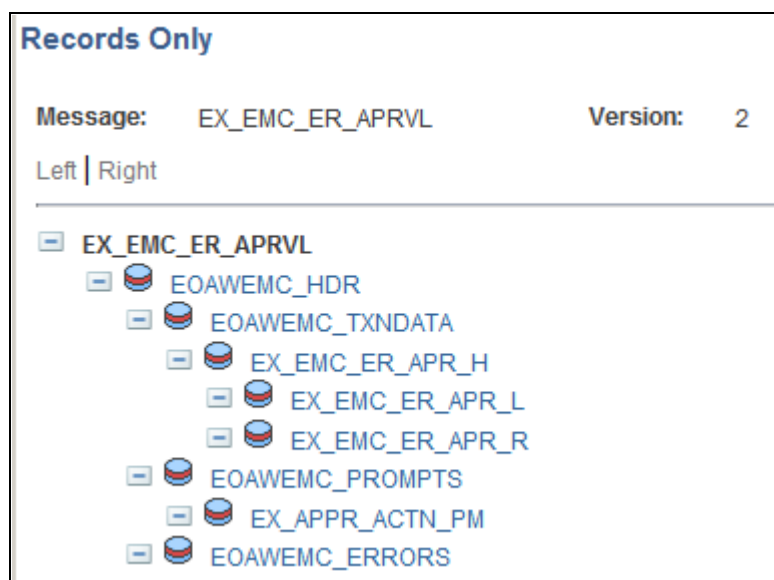
## Defining Integration Broker Message

The Integration Broker message created for EMC needs to be a rowset-based message that includes the EMC records listed in this table:

<b>Record</b>	<b>Level</b>	<b>Description</b>
EOAWEMC_HDR	0	Header record for EMC.
EOAWEMC_TXNDATA	1	Transactional data record for EMC.
EOAWEMC_PROMPTS	1	Prompting record for EMC. Application will add records for the specific prompt action.
EOAWEMC_ERRORS	1	Error record for EMC.

The application transaction record structure is added at level 2 under EOAWEMC\_TXNDATA.

This example shows the Record Only view for a message created to approve expense reports. In this example, the application records that contain transactional data start with EX\_EM\_C.



Records Only view of a message used for EMC

## Defining Service

All service operations must be associated with one or more services. You can create a new service operation for an existing service or create a new service with which to associate your service operation.

See *PeopleTools: PeopleSoft Integration Broker PeopleBook*, Managing Services, Adding and Configuring New Service Operations for Services.

## Defining Service Operation

You will need to create an asynchronous one-way service that uses the EMC message structure.

---

**Note.** The service operation must have the same name as the message.

---

See *PeopleTools: PeopleSoft Integration Broker PeopleBook*, Managing Service Operations, Adding Service Operation Definitions.

You will also need to create an application class that will process the EMC form, to be used as the handler.

The application class created to process inbound subscriptions needs to extend the EOAW\_EMC:API:emailFormManager and EOAW\_EMC:utils classes.

When creating this object, you will need to pass it 2 parameters:

1. Your full instantiated and populated service operation.
2. The language code of the user(s) that will receive this email.

---

**Note.** EMC engine will handle transaction of things like field labels , however you will need to translate any values contained within your message yourself.

---

See [Chapter 8, "Using EMC Classes," page 81.](#)

### **EMC Runtime-Inbound**

EMC inbound processing takes all the values retrieved from the user's submission and merges them with the values sent out to the user (overriding outbound values with inbound values). It then inserts those values into the same service operation used to define your form, and publishes it. You will need add a piece of subscription code to catch the published message and process the data.

Using the utils class method `getErrorCodesRS()`, you can get a list of error codes inserted by the EMC engine while the inbound message was being processed. The list of codes is as follows:

Error Code	Description
0	No Error.
1	Duplicate Response Received - The user submitted the form more than once.

<b>Error Code</b>	<b>Description</b>
2	Sent-To Check Failed - The user to whom the email was sent is not the user from whom the response was received. Keep in mind when dealing with this error that some users have more than one email alias.
3	Invalid date value error.
4	Invalid number value error.

The fields in the error codes rowset are ERROR\_CODE, ROW\_PATH, RECNAME, FIELDNAME, and RECEIVED\_VALUE. Not all fields are used for all errors, though. ROW\_PATH, RECNAME and FIELDNAME for instance are only used when an invalid date or number value is received.

## Defining and Mapping EMC Forms

This section provides an overview of EMC forms and discusses how to:

- Define EMC form.
- Define EMC form layout.
- Define field mapping.

## EMC Forms

EMC allows applications to send, receive and process emails with interactive content. The EMC forms are created based on service operations. The service operation defines the messages structure and the handler necessary to process the EMC forms.

## Pages Used to Define and Map EMC Forms

<b>Page Name</b>	<b>Definition Name</b>	<b>Navigation</b>	<b>Usage</b>
Form Element Designer	EOAWEMC_ELEMENTS	Enterprise Components, Approvals, Email Collaboration, EMC Form	Define metadata used as system data when you deliver an email collaboration.
Form Layout Designer	EOAWEMC_LAYOUT	Enterprise Components, Approvals, Email Collaboration, EMC Layout	Define the layout of the email.
Field Mapping	EOAWXLAT_SYMBOL	Enterprise Components, Approvals, Email Collaboration, Field Mapping	Map field values to the form.



## Defining EMC forms

Access the Form Element Designer page (Enterprise Components, Approvals, Email Collaboration, EMC Form).

To create a new form:

1. Select the Add a New Value page.
2. Enter the Message name.

---

**Note.** The message name must be the same as the service operation name.

---

Upon entering this component, a grid will auto-populate with all of the fields in your message definition , except those that are part of the EMC required records and any that are not marked *include* in your message definition.

<b>Rebuild List</b>	If you change your message definition, use this button to regenerate the field list. Changes are not automatically updated. This will also remove any Form Layout Definitions associated with this message.
<b>Element Type</b>	Select the element type.
<b>Label ID</b>	Label ID prompts against a field's list of available labels. If you leave this field blank, the field will appear on your form without a label.

### ***Element Types***

This table lists the valid element types:

<b><i>Element Type</i></b>	<b><i>Description</i></b>
Blank	A value may be stored in this field by the application, however it will not be transmitted in the email form, and thus may be used to store sensitive data.
Input	Will be represented as a standard text input field.
Output	Will be represented as plain text. This will be the application developers method for providing contextual data to the user.
Secret	Similar to an input field, however it provides superficial security, as the contents of the field appear as a set of masked characters, such as bullets, asterisks, question marks, and so on.
Select	This type of field may be represented in one of two ways: check boxes or multi-select boxes. It gives the user the ability to select multiple values for a single field.

<i>Element Type</i>	<i>Description</i>
Select1	<p>Similar to the Select, it may be represented in multiple ways: radio buttons or drop-down lists. It gives the user the ability to select a single value for a field from a list of available values.</p> <p><b>Note.</b> While radio buttons are available as a design option, some email servers do not enforce mutual exclusion of radio button selections in HTML.</p>
Textarea	This field type gets represented as a long edit box, useful for areas where longer strings of text must be entered or displayed to the user.

## Defining EMC Layouts

Access the Form Layout Designer page (Enterprise Components, Approvals, Email Collaboration, EMC Layout).

The layout rules are fairly rigid on this page. Grids will be automatically created for each new level and fields may not move outside their current grid. Inserting a line break at the header level has the expected result of simply wrapping information to a new line. Should the line break be inserted inside a grid, however, the effect will be like having a "special grid row." That is, the fields after the line break will not be shown as grid columns, but rather as label:field, as in level 0.

On this page, you can change grid labels, field sizes, and number of columns for check boxes and radio buttons. You can also add line breaks and move fields up or down.

Use the Preview button to see a preview of your form layout.

## Defining Field Mapping

Access the Field Mapping page (Enterprise Components, Approvals, Email Collaboration, Field Mapping).

Field Name		EOAWEMC_ELEM_TYPE					
		*Field Value	Long Name	Symbolic Name	Content Name		
1	I	Input	input			+	-
2	O	Output	output			+	-
3	S	Secret	secret			+	-
4	M	Select	select			+	-
5	N	Select1	select1			+	-
6	T	Textarea	textArea			+	-

Field Mapping page

Use the field mapping page to define values for fields that are used with drop-down lists in your form.

## Triggering Email Collaboration

This section discusses how to:

- Update approval transaction registry to send email approvals.
- Configure transaction for email approval.
- Schedule the application engine program EOAWEMC.
- Add or modify email addresses for users.

## Pages Used to Trigger Email Collaboration

<i>Page Name</i>	<i>Definition Name</i>	<i>Navigation</i>	<i>Usage</i>
Register Transaction	EOAW_TXN	Enterprise Components, Approvals, Approvals, Transaction Registry	Create and update the transaction registry.
Configure Transactions	EOAW_TXN_NOTIFY	Enterprise Components, Approvals, Approvals, Transaction Configuration	Configure how a specific approval process uses email notification options.  See <a href="#">Chapter 4, "Defining the Approval Transaction Registry,"</a> <a href="#">Configuring Approval Transactions</a> , page 43.
Recurrence Definition	PRCSRECURDEFN	PeopleTools, Process Scheduler, Recurrences	Define how often you want the process scheduler to run processes.
Application Engine Request	AE_REQUEST	PeopleTools, Application Engine, Request AE	Set up the Request AE for EOAWEMC.
Email Addresses	USER_EMAIL	PeopleTools, Security, User Profiles, User Profiles, General  Select the Email Addresses link on the User Profiles - General page.	Modify email addresses.

## Updating Approval Transaction Registry to Send Email Approvals

Access the Register Transactions page (Enterprise Components, Approvals, Approvals, Transaction Registry).

**Register Transactions**

Process ID: ERApproval

\*Description: Expense Rpt Approval Process

Object Owner ID: Expenses

\*Cross Reference Table: EX\_SHEET\_AW

Worklist Prefix:

**Notification Options**

\*Enable Notifications: Enable Email and Worklist

\*Notification Strategy: Online Processing

Use Email Approvals: ☒

Form Generator Package Root: EX\_APPRVL EMC

Form Generator Class Path: OUTBOUND:ERFormGenerator

Register Transactions page

### Use Email Approvals

Select this check box to use Email Approvals.

### Form Generator Package Root

Enter the package name that contains the class that contains the form generator application class for this transaction, or use the Lookup button to search for and select one.

### Form Generator Class Path

Enter the application class path that contains the form generator application class for this transaction, or use the Lookup button to search for and select one.

## Application Class for Form Generator

The application class for the form generator must be an extension of EOAW EMC:API:formGeneratorBase. When the approvals engine determines that it needs to send out an email approval, it will create an instance of the class you specify on this page. It will pass to it 2 parameters:

- &threads as array of EOAW CORE:ENGINE:Thread - This is an array of Approval Engine Threads.
- &userID as String - The userid of the individual receiving the email.

The base class provided takes the parameters passed in and makes them protected properties. Immediately after instantiating your object, the approvals engine will call the only method defined in the base class: returnEFM(). In this method you should take the threads and userid, create an instance of the emailFormManager, and return it. The approvals engine will then call the sendEmails() method on the object you return.

### See Also

[Chapter 8, "Using EMC Classes," page 81](#)

## Configuring Transactions for Email Approval

Access the Configure Transactions page (Enterprise Components, Approvals, Approvals, Configure Transactions).

**Configure Transactions**

Process ID: ERApproval

**Ad Hoc Approver Options**

\*Approval User Info View: EX\_APPR\_INFO\_VW

Ad Hoc User List:

**Notification Options**

☐ Send Email Approvals to All

Email Approval User List: Expense Report Email Approvers

\*Delivery Method: Inline - HTML Email

☐ Perform Sent-To Security Check

Configure Transactions page

The Notification Options section is only available if the *Use Email Approvals* check box is selected for this transaction on the Transaction Registry page.

See and [Chapter 4, "Defining the Approval Transaction Registry," Notification Options, page 46.](#)

## Scheduling the Application Engine Program EOAWEMC

To schedule the application engine program EOAWEMC:

1. (Optional) Access the Recurrence Definition page (PeopleTools, Process Scheduler, Recurrences), and specify an interval for the process scheduler recurrence.
2. Select PeopleTools, Process Scheduler, Processes for the application engine program EOAWEMC and up the process definition.
3. Access the Process Definition Option page and specify the process schedule server and the recurrence.

See *PeopleTools: PeopleSoft Process Scheduler PeopleBook*, Defining PeopleSoft Process Scheduler Support Information, Defining Process Definitions.

## Adding or Modifying Email Addresses for Users

Access the Email Addresses page (PeopleTools, Security, User Profiles, User Profiles, General, then click the Edit Email Addresses link on the General page).

To update your own email address, select System Profile.

Email Addresses

User ID: VP1

Customize | Find | View All | First 1 of 1 Last

Primary Email Account	Email Type	Email Address		
<input checked="" type="checkbox"/>	Business	peoplesoft@peoplesoft.com		

Email Addresses page

## Chapter 8

# Using EMC Classes

This chapter discusses the EMC classes:

- EmailFormManager
- Utils

---

## Understanding EMC Classes

EMC classes provide the functions to send and retrieve email approvals. They are included in the application package EOAW\_EMC.

---

## EmailFormManager Class

This class provides the utility methods to send collaborative emails.

---

## EmailFormManager Class Methods

This section describes the public methods of interest to application developers for emailFormManager, in alphabetical order.

### addRecipient

#### Syntax

**addRecipient**(*&emailAddress*, *&userID*)

#### Description

Users added via this method will be allowed to act on the email form.

## Parameters

<i>Parameter</i>	<i>Description</i>
<code>&amp;emailAddress</code>	Email address, as string. Required.

<i>Parameter</i>	<i>Description</i>
<code>&amp;userID</code>	User ID, as string. Optional.

## Returns

None.

## addCC

### Syntax

`addCC(&emailAddress)`

### Description

Email addresses added via this method will be copied on every email sent out.

## Parameters

<i>Parameter</i>	<i>Description</i>
<code>&amp;emailAddress</code>	Email address, as string. Required.

## Returns

None.

## addBCC

### Syntax

`addBCC(&emailAddress)`



## Description

Email addresses added via this method will be blind copied on every email sent out.

## Parameters

<i>Parameter</i>	<i>Description</i>
&emailAddress	Email address, as string. Required.

## Returns

None.

## addAttachment

### Syntax

**addAttachment**(&filePath, &filePathType,&fileName, &fileTitle)

## Description

Adds an attachment to be sent along with the email.

## Parameters

<i>Parameter</i>	<i>Description</i>
&filePath	A string consisting of the complete path to the file and the filename itself.
&filePathType	A number representing type of file path used in first parameter. Use tools system variables such as %FilePath_Relative.
&fileName	A string representing the name of the file being attached.
&fileTitle	A string consisting of the title of the file. The titles appear near the attachment icons in place of the fully qualified filename.

## Returns

None.

## sendEmails

### Syntax

**sendEmails**(*&sentToCheckOnReturn*)

### Description

Does the actual send of the email once all other information is ready.

### Parameters

<i>Parameter</i>	<i>Description</i>
<i>&amp;sentToCheckOnReturn</i>	If set to true, the EMC will verify that the user to whom the email was sent is the same as the user who sent the response. If this is not true, an error will be inserted into the error stack allowing the implementing application to handle it as they see fit.

---

## EmailFormManager Class Properties

This section describes the emailFormManager class properties.

### inlineText

#### Description

A string of text that the user will see if the deliveryMethod property is set to inline but their mail client does not support HTML emails. A default is provided. inlineText and attachmentText will never be delivered in the same email.

### attachmentText

#### Description

A string of text that the user will see if the deliverMethod property is set to attachment. This text will generally be used as instructional text telling them to detach the html form and submit it. A default is provided. inlineText and attachmentText will never be delivered in the same email.

## subject

### Description

The subject of the email to be sent. Default is *PeopleSoft Collaborative Email Routing*. Keep language considerations in mind when using this property.

## submitMessage

### Description

A string of text that the user will see if the `deliveryMethod` property is set to `inline` but their mail client does not support HTML emails. A default is provided. `inlineText` and `attachmentText` will never be delivered in the same email.

## from

### Description

The name of the person or account sending this email.

## fromEmail

### Description

The email address of the person or account sending this email.

## replyTo

### Description

The address users should reply to if there is a problem. This defaults to the same email address that form responses are fielded from.

## prependText

### Description

Use this variable to include any text you would like inserted before the form being emailed.

## appendText

### Description

Use this variable to include any text you would like inserted after the form being emailed.

## deliveryMethod

### Description

Specifies whether to send the emails as inline html or attachments. Attachments will include a better user interface as the browser used to view the html attachment is likely to support more JavaScript and DHTML than email clients like Lotus Notes. Valid values are *I* for Inline and *A* for attachment.

---

## EOAW\_EMC:utils Class

This class contains methods that support the main functionality of EMC.

---

## EOAW\_EMC:utils Class Methods

This section describes utils class methods.

## getAppRS

### Syntax

**getAppRS**(*&msgRS*)

## Description

This method provides a single point from which to retrieve the applications rowset from the message definition rowset, as the message definition should have the application rowset starting at level 2. Given the rowset object representing your entire message, this rowset will return just the part of the overall rowset representing your transaction.

## Parameters

<i>Parameter</i>	<i>Description</i>
&msgRS	The rowset object representing your entire message.

## Returns

Rowset.

## getErrorCodesRS

### Syntax

`getErrorCodesRS(&msgRS)`

## Description

This method provides a single point from which to retrieve the error codes rowset from the message definitions rowset.

## Parameters

<i>Parameter</i>	<i>Description</i>
&msgRS	The rowset object representing your entire message.

## Returns

Rowset.

## getPromptsRS

### Syntax

**getPromptsRS**(*&msgRS*)

### Description

This method provides a single point from which to retrieve the prompts rowset from the message definition's rowset, as the message definition should have the prompts rowset starting at level 2.

### Parameters

<i>Parameter</i>	<i>Description</i>
<i>&amp;msgRS</i>	The rowset object representing your entire message.

### Returns

Rowset.

## getRowFromPath

### Syntax

**getRowFromPath**(*&startingRS*,*&path*,*&createIfNull*)

### Description

This method is useful for error processing. When your message expects a date or number field to be returned by the user, but the EMC cannot cast the value it gets back into one of those data types, it will put an error code in the exceptions stack. Along side that code in the stack, will be a row path. A row path is the path you need to follow to get to a specific row in a rowset. It follows the format (n)SCROLLNAME[(n)...] where n is a row number and SCROLLNAME is the name of the record comprising the rowset you want to retrieve in that row. When you get an error of one of these 2 types, you can call this method, passing in your message rowset, the rowpath in the error stack, and *false*. You will be returned the exact row where the error occurred (the name of the record and field are also in the error stack).

## Parameters

<i>Parameter</i>	<i>Description</i>
&startingRS	The rowset from which to start searching for a row, as rowset.
&path	Path to the desired row, as string. Syntax - (n)[SCROLLNAME(n)...] where n is the row in the scroll preceding it. The rowset preceding the first n is &startingRS.
&createIfNull	If set to true, the method will insert as many rows as necessary to retrieve the row successfully. If set to false and the row is not present, an exception will be thrown.





## Chapter 9

# Using the Notification and Escalation Manager

This chapter provides an overview of notification and escalation and discusses how to:

- Associate events to a server.
- Set up an escalation event.
- Set up a notification and escalation manager (NEM).

---

## Understanding Notification and Escalation Manager

Notification and Escalation Manager (NEM) is a mechanism used to process notifications and escalations on a specified interval.

For example, escalations are used when an approver has not responded within a specified time period to a transaction that is pending approval. You can specify the time period (timeout) and you can specify alternate approvers to whom to notify and escalate the approval for further action. Timeout options are defined on the Approval Path Definition page.

## Pages Used to Set Up the Notification and Escalation Manager

<i>Page Name</i>	<i>Definition Name</i>	<i>Navigation</i>	<i>Usage</i>
Event Types	EOAW_NEM_EVENTS	Enterprise Components, Approvals, Notification and Escalations, Events	Associate events to a server.
Notification and Escalations	EOAW_NEM_SETUP	Enterprise Components, Approvals, Notification and Escalations , Notifications and Escalations	Set up an escalation event and define the evaluation and action details.
Event Status	EOAW_NEM_STATUS	Enterprise Components, Approvals, Notification and Escalations, Status	Check status of notification.
Schedule JobSet Definitions	SCHDLDEFN	PeopleTools, Process Scheduler, Schedule JobSet Definitions	Set up a NEM to define the job to run, and how often you want it to run.

Associating Events to a Server

Access the Event Type page (Enterprise Components, Approvals, Notification and Escalations, Events).

Event Type:

APPROVAL ACTIVITY EMAIL

Server Name:

PSNT

Event Types Description:

Offline email processing for notification events triggered by the event handler

Event Type page

- Event Type

Select an event type. PeopleSoft applications deliver some event types, such as *ESCALATION\_EVENT* and *APPROVALACTIVITYEMAIL*.
- Server Name

Select the server on which to run the notification.

Setting Up an Escalation Event

Access the Notification and Escalations page (Enterprise Components, Approvals, Notification and Escalations, Notification and Escalations).

Event ID:

APPROVALACTIVITYEMAIL

\*Event Type:

APPROVAL ACTIVITY EMAIL

☒ Active

Event Types Description:

Offline email processing for notification events triggered by the event handler

Recurrence: 15 Min

Repeat Time:

\*Evaluation Type:

PeopleCode

Package:EOAW\_EMAIL

Class:EmailEvaluation

\*Action Type:

PeopleCode

Package:EOAW\_EMAIL

Class:EmailSender

Notification and Escalations page

- Event Type

The server information that was entered on the Event Type page.

<b>Event Types Description</b>	The value entered in the Description field on the Events Type page.
<b>Active</b>	Select to enable the escalation process.
<b>Recurrence</b>	Enter a time interval at which to run the evaluation process.
<b>Repeat Time</b>	Enter a time period to limit the number of times the action step is run.
<b>Evaluation Type</b>	<p>Select a method for evaluation. Possible values are:</p> <ul style="list-style-type: none"> <li>• <i>PeopleCode</i>: Select if you are using a custom application package or class written using PeopleSoft Application Designer.</li> <li>• <i>Query Object</i>: Select if you are using a query set up using the PeopleSoft Query Manager tool.</li> <li>• <i>SQL View</i>: Select if you are using a record object created using the PeopleSoft Application Designer.</li> </ul> <hr/> <p><b>Note.</b> For escalations, the evaluation type should be <i>SQL View</i>.</p> <hr/>
<b>Name</b>	Displays the name of the query object or SQL view, depending on which is selected as the evaluation type.
<b>Action Type</b>	<p>Select an action:</p> <ul style="list-style-type: none"> <li>• <i>PeopleCode</i>: Select if you are using a custom application package or class written using the PeopleSoft Application Designer.</li> <li>• <i>Email</i>: Enter an email address and notification template.</li> </ul> <hr/> <p><b>Note.</b> For escalations, the action type should be <i>PeopleCode</i>.</p> <hr/>
<b>Package</b>	<p>Select the application package that contains the escalation utility.</p> <hr/> <p><b>Note.</b> For escalations, the package should be <i>EOAW_CORE</i>.</p> <hr/>
<b>Class</b>	For escalations, select <i>Escalator</i> .

## Setting Up a NEM

Access the Schedule JobSet Definitions page (PeopleTools, Process Scheduler, Schedule JobSet Definitions).

Schedule JobSet Definition		Schedule JobSet Items	Schedule JobSet Requests
Schedule Name:	SAC_NEM		<a href="#">JobSet Report</a> <a href="#">Report Manager</a> <a href="#">Process Monitor</a>
Job Name:	NEM_MAIN		
<b>Schedule Information</b>			
User ID:	PS		
*Description:	SAC_NEM	*Status:	Inactive ▼
*Run Control ID:	SAC_NEM_MAIN	*Priority:	Medium ▼
<b>Time Information</b>			
*Begin Date:	10/11/2010	*Time:	4:46:25PM
		*Time Zone:	PST
Recurrence Name:			<input type="button" value="Run Now"/>
<b>Server Information</b>			
*Server Run Option:	Any Server ▼		
Primary Server:		Operating System:	▼

Schedule JobSet Definition page

**Schedule Name** Select *EOAW\_NEM* for the notification and escalation manager.

**Job Name** Select *NEM\_MAIN* for the notification and escalation manager.

**Status** Select *Active* for the notification and escalation manager.

**Run Control ID** Enter the run control that has the run configuration desired.

**Recurrence Name** Enter a value that specifies how often the process runs.

See *PeopleTools: PeopleSoft Process Scheduler PeopleBook*, Defining Jobs and JobSets.

## Chapter 10

# Using the Approval Monitor

This chapter provide an overview of the Approval Monitor discusses how to:

- Configure the Approval Monitor.
- Use the Approval Monitor.
- Use the User Monitor.

---

## Understanding the Approval Monitor

The Approval Monitor gives administrators a view into all approvals to which they have access, as well as the ability to take necessary actions on pending approvals. Administrators are provided access to Approval Transactions based on the Role defined as System Administrator on the Approval Process Definition page. Actions available for the administrator are:

<b>Reassignment</b>	Allows the system administrator to reassign pending approvals to a new approver based on search criteria.
<b>Approve</b>	Allows the administrator to act on behalf of the assigned approver. The approval is initiated for a specific user, wherever that user may be pending within a specific transaction. Once the administrator takes action, the approval resumes the approval process.
<b>Denial</b>	Allows the administrator to act on behalf of the assigned approver. The denial is initiated for a specific user, wherever that user may be pending within a specific transaction.
<b>Ad Hoc</b>	Allows the administrator to add a reviewer or approver to a specific transaction.
<b>Resubmit</b>	Allows the administrator to resubmit a completed transaction to all approvers in the approval path.
<b>Push Back</b>	Allows the administrator to send the process back to the previous approver.

## Understanding Approval Reassignment

The administrator can reassign pending tasks to another approver, or an administrator can reassign all tasks that belong to a specific approver to another approver. Use reassignment in the following situations:

- The approver chooses to redirect the task to another approver, thus delegating a specific task (step) to another approver.
- The administrator decides to reassign all pending tasks within a step that belong to an approver to another approver.

This reassignment usually occurs when an approver is unexpectedly absent and the administrator reassigns all pending tasks to another.

When the administrator redirects a workflow task to another approver, the administrator can modify the approval process map.

---

**Note.** The Approval Framework is set up for administrative reassignment and escalations only.

---

## Configuring the Approval Monitor

This section discusses how to configure the Approval Monitor.

You can configure the Approval Monitor to display the information necessary for an administrator to approve a transaction when the original approver is not available. You can also configure the actions that can be performed by the administrator. Each process ID can be configured.

### Page Used to Configure the Approval Monitor

<i>Page Name</i>	<i>Definition Name</i>	<i>Navigation</i>	<i>Usage</i>
Approval Monitor Configuration	EOAW_MONDIS_CONFIG	Enterprise Components, Approvals, Approvals, Monitor Configuration	Configure the approval monitor.

## Configuring the Approval Monitor

Access the Approval Monitor Configuration page (Enterprise Components, Approvals, Approvals, Monitor Configuration).

Approval Monitor Configuration

Process ID: SP\_WORKORDER

Configure Monitor Approvals

☒ Include in Search

☐ Display Only

☒ Allow Approve

☒ Allow Resubmit/Restart

☒ Allow Mass Approve

☒ Allow Deny

☒ Allow Reassign

☒ Allow Mass Deny

☒ Allow Pushback

☒ Allow Mass Reassign

Configure User Monitor

Find | View All | First | 1 of 1 | Last

User Role: SP\_WO\_APPROVER

Description:

☒ Allow Mass Approve

☒ Allow Mass Deny

Transaction Display Level

Find | View All | First | 1 of 1 | Last

Display Level: Header

Record Name: SPF\_WORDERREC

Customize | Find | View 100 | First | 1 of 103 | Last

	Field Name	Field Label ID	Display		
1	BUSINESS_UNIT	BUSINESS_UNIT	Yes	↓	↑

Approval Monitor Configuration page

Configure Monitor Approvals

Use this section to control the actions available in the Monitor Approvals page for this approval process.

- Include in Search

Select to have the approval process available on the Monitor Approvals page.
- Display Only

Select to have approval process available in display only mode.

**Allow Approve, Allow Deny, Allow Pushback, Allow Resubmit/Restart, Allow Reassign, Allow Mass Reassign, Allow Mass Approve, and Allow Mass Deny**

Select each action that will be available in the Approval monitor for this approval process.

See [Chapter 10, "Using the Approval Monitor," Utilizing the Approval Monitor for a Specific Approval Process, page 101.](#)

### **Configure User Monitor**

Use this section to assign user roles and approval functions for monitoring this approval process.

### **Transaction Display Level**

Use this section to select which fields at the header and line level should be displayed in the User Monitor for this approval process.

---

## **Using the Approval Monitor**

This section discusses how to:

- Use the Approval Monitor search page.
- View search results.
- Utilize the Approval Monitor for a specific approval process.

The approval monitor gives administrators a view into all approvals to which they have access, as well as the ability to take necessary actions on pending approvals.

---

**Warning!** Due to the complex rules used by PeopleSoft Expenses, the Monitor Approvals page should not be used to approve or deny expense transactions. To approve and deny expense transactions use the PeopleSoft Expenses approval pages. For more information, refer to *PeopleSoft Expenses PeopleBook*, Managing Approvals in PeopleSoft Expenses.

---

## **Pages Used to Use the Approval Monitor**

<b>Page Name</b>	<b>Definition Name</b>	<b>Navigation</b>	<b>Usage</b>
Monitor Approvals	EOAW_ADM_MON_SRC	<ul style="list-style-type: none"> <li>• Enterprise Components, Approvals, Approvals, Monitor Approvals</li> <li>• Enterprise Components, Approvals, Approvals, User Monitor</li> </ul>	Use the Monitor Approvals page as a system administrator to search approval processes and perform mass reassignments.



Page Name	Definition Name	Navigation	Usage
Monitor Approvals	EOAW_ADM_MON_ACT	From the Monitor Approvals Search page, select the link for the approval step you want to modify.	Use this page to perform an action on a specific approval process.

## Using the Approval Monitor Search Page

Access the Monitor Approvals page (Enterprise Components, Approvals, Approvals, Monitor Approvals).

**Monitor Approvals**

▼ Search Criteria

Approval Process:  Definition ID:

Header Status:  Last Modified:

Approver:

Approver Status:

Originator:

Requester:

▼ Administrative Actions

The selected approver does not currently have an alternate approver selected in their user profile. You must manually select an alternate approver to reassign pending tasks.

Approver's Oper ID:  ☐ Allow Auto Approval

Reassign To:  ☐ Allow Self-Approval

Comment:

Monitor Approvals-Search Criteria page

### Approval Process

Select an approval process. The list of approval processes available is determined by the administrator role associated with the approval process definition. If a user is associated with the role specified in the Admin Role field on the Setup Process Definitions page, they can view or act on that process ID within the Approval Monitor.

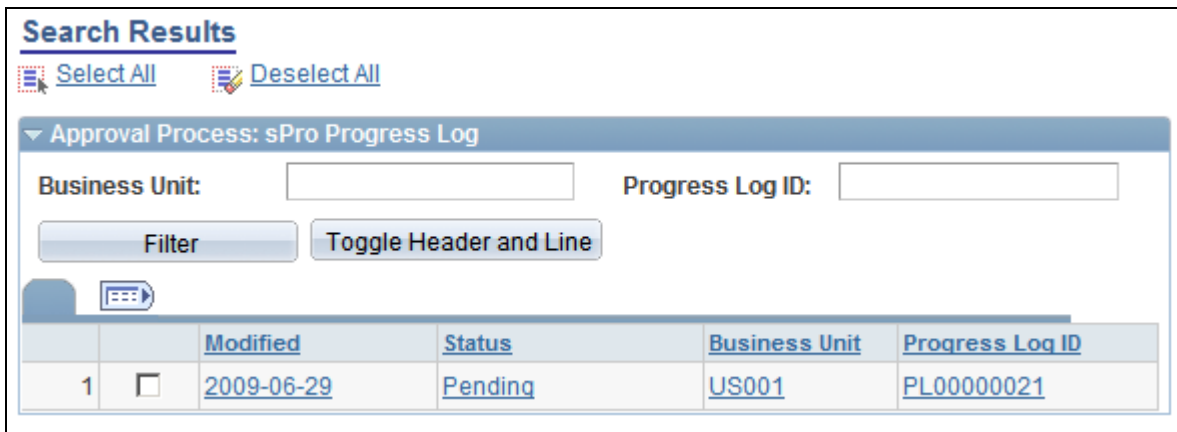
<b>Definition ID</b>	Select the process definition that is determined on the Setup Process Definition page.
<b>Header Status</b>	<p>Select a status in this field to display the selected status. Choices are:</p> <ul style="list-style-type: none"> <li>• <i>Approved</i></li> <li>• <i>Complete</i></li> <li>• <i>Denied</i></li> <li>• <i>Hard Deny</i></li> <li>• <i>Initial</i></li> <li>• <i>Not Active</i></li> <li>• <i>Pending</i></li> <li>• <i>Pending Denial</i></li> <li>• <i>Suspended/Pending Denial</i></li> <li>• <i>Terminated</i></li> </ul>
<b>Approver</b>	Select an approver. In order to view or take action on an approval processes for a specific approver this field is required.
<b>Approver Status</b>	Select a status from those available. This field is only available when a specific approver is selected in the Approver field. The statuses available for selection are based on the statuses of that specific approver in the cross-reference table associated with the approval process ID.
<b>Originator</b>	Select the user who entered the transaction that started the approval process.
<b>Requester</b>	Select the user who requested the transaction that started the approval process. The originator is not always the requester.

### ***Reassign Pending Tasks***



Use this section to reassign approvals to another user. This section is only available if the process allows reassignment.

## **Viewing Search Results**

After entering the selection criteria on the Monitor Approvals page, click the Search button.



**Search Results**

 [Select All](#)  [Deselect All](#)

▼ **Approval Process: sPro Progress Log**

**Business Unit:**  **Progress Log ID:**

		<a href="#">Modified</a>	<a href="#">Status</a>	<a href="#">Business Unit</a>	<a href="#">Progress Log ID</a>
1	<input type="checkbox"/>	<a href="#">2009-06-29</a>	<a href="#">Pending</a>	<a href="#">US001</a>	<a href="#">PL00000021</a>

Approval Monitor Search results

For the approval process selected, specific fields will be displayed based on the transaction display level configured for the approval process.

See and [Chapter 10, "Using the Approval Monitor," Transaction Display Level, page 98.](#)

**Filter** Use this button to filter the result set based on criteria entered.

**Toggle Header and Line** If the approval process contains header and line approvals, you can use this button to toggle the display to view the details.

## Utilizing the Approval Monitor for a Specific Approval Process

Access the Monitor Approvals page (Enterprise Components, Approvals, Approvals, Monitor Approvals, then select the link for the approval step you want to modify).

Monitor Approvals

Approval Process:SP\_PLOG

Definition ID:US001

Business Unit:US001

Progress Log ID:PL00000021

Approver:

Comment:

Line Details

	Business Unit	Progress Log ID	Line Number
1	US001	PL00000021	1

Reassign Pending Tasks

The selected approver does not currently have an alternate approver selected in their user profile. You must manually select an alternate approver to reassign pending tasks.

Reassign To:

☐ Allow Self-Approval

☐ Allow Auto Approval

Reassign

Administrative Approve/Deny

Act on behalf of the approver selected above by selecting the Approve/Deny buttons below. This action will apply to all tasks pending for the approver selected within the context of this transaction.

Approve

Deny

Pushback

Restart

Progress Log Header Amount

Progress Log ID : PL00000021:Pending

View/Hide Comments

Request Information

Start New Path

Progress Log Header Amount

Skipped

Skipped

Pending

Morgan, David

sPro BU default Administrator

→

Kenneth Schumacher

Services procurement approvals

→

Multiple Approvers

Error Step

+

Monitor Approvals page for a specific approval process

This page will reflect the options selected when you configured the approval monitor for the process. The Reassign Pending Tasks section and the buttons available in the Administrative Approve/Deny section will display as configured on the Approval Monitor Configuration page.

**Warning!** Due to the complex rules used by PeopleSoft Expenses, the Monitor Approvals page should not be used to approve or deny expense transactions. To approve and deny expense transactions use the PeopleSoft Expenses approval pages. For more information, refer to *PeopleSoft Expenses PeopleBook*, Managing Approvals in PeopleSoft Expenses.

- Approver

Select an approver. All approvers associated with pending steps within the approval process are listed.
- Comment

Enter text to appear under the approval graphic in the Approval Comment History section.

### ***Reassign Pending Tasks***

<b>Reassign To</b>	Select an approver to whom to reassign all pending steps within the approval process.
<b>Reassign</b>	<p>Click the button to initiate task reassignment. At the end of the procedure, all pending task of the corresponding approval process instance is assigned to the user who is specified in the Reassign To field.</p> <p>Reassignment history is captured as comments and viewable in the approval graphic at the bottom of this page.</p>
<b>Allow Self-Approval</b>	Select to enable self-approval. When it is enabled, the approval is assumed and the process continues.
<b>Allow Auto Approval</b>	Select to enable auto-approval. When it is enabled, the system remembers an approver's action for that process at the header or line level, and applies the same action automatically for any subsequent appearance in the Approval Framework routing.

### ***Administrative Approve/Deny***

<b>Approve</b>	Click the button to act on behalf of the selected approver. This action applies to all tasks pending for the approver selected within the context of the approval process.
<b>Deny</b>	Click the button to act on behalf of the selected approver. This action will apply to all tasks pending for the approver selected within the context of the approval process.
<b>Pushback</b>	<p>Click to requeue the previous step to its approver. This button is only available at a step that is greater than one.</p> <p>For example, a transaction has three approvers. The first approver has approved the transaction, therefore, the transaction is pending at step two. The administrator needs additional information from the requester and, therefore, pushes the transaction back to the requester.</p>
<b>Restart</b>	Click to restart a pending transaction to all approvers in the approval path. This button is only available when the transaction is pending.
<b>Resubmit</b>	Click to resubmit a completed transaction to all approvers in the approval path. This button is only available when the transaction is complete. The transaction can only be resubmitted in its current state and cannot be modified before resubmitting for approval.
<b>View/Hide Comments</b>	Click to expand the Comments section and view comments regarding reassignment or text that is captured in the Comment field when any of the action buttons is clicked.

**Request Information**

Click to request additional information about this transaction from another user in the system. This selected user might not be an approver of this transaction, but he or she must respond to the request before the approval process can move forward to the next level. The system displays a new approval graphic, showing that more information has been requested as well as the user responsible for the request. As the requester, you can use the comments section to clarify the type of information you are requesting.

***Requesting Additional Information About the Transaction***

When you click the Request Information link in the approval graphic, the system:

1. Places the current transaction on hold. Doing so prevents anyone from coming in and approving the transaction before the approver (who requested the information) gets the chance to review the returned information.
2. Inserts the requester as a reviewer of the transaction for that step.
3. Triggers the *Request Information* event. Any processing in the applications event handler is triggered; any notifications that is defined in the transaction configuration is sent as well.

The approver (who requested the information) can act on the transaction by clicking any action button at any time.

Here is a use case of the Request Information link:

1. Jane Doe submits a requisition request to purchase a monitor and a hard drive.  
The request routes to three group managers at step 1. Two of the three managers' approvals are required.
2. Manager Jen Smith wants to know why Jane needs to purchase such a large monitor, so she clicks the Request Information link on the approval graphic trying to gather more information about this requisition.  
The requisition transaction is now on hold and awaiting her action. The other two managers can still issue their approvals at any time, but the transaction will not move until the hold is lifted.
3. Depending on the how the application implemented the approval feature, Manager Jen Smith can enter comments in the Comments field, which will render in the Status Monitor, or seek information by any other means of communication (for example, by phone, by email, walk to my office and ask).  
Similarly, depending on the implementation, Jane Doe can respond by either entering comments through the application (this will trigger the Request Information Added event), or communicating to Jen Smith through other means (for example, by phone, by email, or other).
4. Jen Smith can either approve or deny the transaction after (or before) she receives a response. Normal processing resumes.

See [Chapter 11, "Using Approval Framework Base Classes," ApprovalManager Class Methods, page 117](#) and [Chapter 11, "Using Approval Framework Base Classes," ApprovalEventHandler Class Methods, page 134](#).

### ***Reassigning Tasks Assigned to You***

To reassign your tasks to another approver, select a step assigned to you as an approver and request that the step be reassigned to an alternate. You must have an administrator role to perform this task.

The Approval Framework reassigns that step to the newly appointed approver, and deletes the original approver's worklist entry. The system creates a new worklist entry for the new approver, and notifies the new approver.

The Approval Framework adds a comment to the approval thread to log the reassignment.

### ***Reassigning Tasks as an Administrator***

To reassign a specific approver's pending tasks to another approver:

1. Filter the display to present pending approval processes for the specific approver.
2. Indicate the steps to be reassigned and the users affected.

The system submits a request to the Approval Framework to reassign all of the pending steps.

Once you have reassigned the pending tasks to a new approver, the approval path is updated and the approval transaction is routed to the new approver.

---

**Note.** You can create reassignments through the user profile. However, workflow reassignments through the user profile don't alter the actual approval process. Reassigning using the Monitor Approvals component performs the reassignment and creates the worklist for the new user.

---

The administrator can also take action instead of reassigning.

### ***Comment History***

Expand the Comment History section to view the previous workflow streams. This section is available when the system has resubmitted the transaction due to a change in the transaction.

For example, a requisition is approved by two out of three approvers. Then the requisition is changed before the third approver sees it. The system resubmits the requisition and begins the approval path from the beginning. The Comment History section retains the original stream, indicating that the first two approvers had approved the original requisition.

---

## **Using the User Monitor**

This section discusses how to use the User Monitor.

The User Monitor provides a generic approval page that is very similar to the Approval Monitor. The only difference is that it is designed to only display approval items where the signed on user is either the approver or the requester.

## Page Used to Use the User Monitor

Page Name	Definition Name	Navigation	Usage
User Monitor - Monitor Approvals	EOAW_ADM_MON_SRC	Enterprise Components, Approvals, Approvals, User Monitor	Approve and review transactions.

## Using the User Monitor

Access the User Monitor - Monitor Approvals page (Enterprise Components, Approvals, Approvals, User Monitor).

**Monitor Approvals**

**Search Criteria**

Approval Process:  Definition ID:

Header Status:  Last Modified:

Current Role:

Approver Status:

Originator:

Requester:

**Search Results**

[Select All](#) [Deselect All](#)

**Approval Process: Requisition**

Business Unit:  Requisition ID:

		Modified	Status	Business Unit	Requisition ID
1	<input type="checkbox"/>	<a href="#">2009-01-23</a>	<a href="#">Pending</a>	<a href="#">US100</a>	<a href="#">0000000044</a>
2	<input type="checkbox"/>	<a href="#">Never</a>	<a href="#">Pending</a>	<a href="#">US100</a>	<a href="#">0000000045</a>

User Monitor - Monitor Approvals page for current role Approver

Select your current role, either Approver or Requester and click the search button to display the transactions.

See and [Chapter 10, "Using the Approval Monitor," Using the Approval Monitor Search Page, page 99.](#)



## Chapter 11

# Using Approval Framework Base Classes

This chapter provides an overview of Approval Framework base classes and discusses:

- How to import Approval Framework type classes.
- The LaunchManager class.
- The ApprovalManager class.

---

## Understanding Approval Framework Base Classes

Approval Framework Base Classes provide functions to launch workflow, manager approvals, display approval monitor. In order to build custom Approval Framework processes, you will need to extend these class in your PeopleCode.

### *How to Import Approval Framework Base Classes*

The Approval Framework base classes are not built-in classes, like Rowset, Field, Record, and so on. They are application classes. Before you can use these classes in your PeopleCode program, you must import them to your program.

An import statement names either all the classes in a package or one particular application class. For importing Approval Framework base classes, Oracle recommends that you import the functions class in the application package that is specific to your needs.

The function classes are stored in EOAW\_CORE application packages:

You should use construct your import statement to include the classes necessary for your code. Some examples are:

```
import EOAW_CORE:*;  
import EOAW_CORE:LaunchManager;  
import EOAW_CORE:ApproverManager
```

---

## LaunchManager Class

This class provides the utility methods to launch Approval Framework.

## LaunchManager Class Methods

Application developers should instantiate this class as a component-scoped variable. The best place to initialize this class is in the component post-build event. This section describes the LaunchManager class methods. The methods are discussed in alphabetical order.

### LaunchManager

#### Syntax

**LaunchManager**(*&awprcs\_id*,*&hdr\_*, *&requester\_*)

#### Description

Launches the Approval Framework process.

#### Parameters

<i>Parameter</i>	<i>Description</i>
<i>&amp;awprcs_id</i>	The approval Process ID that has been defined in the Transaction Registry, as string.
<i>&amp;hdr_</i>	The header record defined for the approval processes in the Transaction Approval Levels in the Transaction Registry, as record.
<i>&amp;requester_</i>	The Operator ID of the user initiating the approval process, as string.

#### Returns

None.

#### Example

This is an example to launch Approval Framework for VoucherApproval.

```
import EOAW_CORE:*;
Component EOAW_CORE:LaunchManager &launchMgr;

If VOUCHER.VCHR_APPRVL_FLG = "W" Then
    &vchrRecord = CreateRecord(Record.VCHR_AF_HDR_VW);=>
    GetLevel0()(1).GetRecord(Record.VOUCHER).CopyFieldsTo(&vchrRecord);
    &launchMgr = create EOAW_CORE:LaunchManager("VoucherApproval", &vchrRecord,&
    %OperatorId);
```

## DoSubmit

### Syntax

**DoSubmit()**

### Description

Used to submit a new transaction for approval. An exception is thrown if the property submitEnabled is not true.

### Parameters

None.

### Returns

None.

### Example

This is an example of submitting the transaction.

```
import EOAW_CORE:*;  
  
Component EOAW_CORE:LaunchManager &launchMgr;  
  
&launchMgr.DoSubmit();
```

## DoReSubmit

### Syntax

**DoReSubmit()**

### Description

Used to resubmit a transaction which was previously submitted, and the previous approval processes have completed (approved, denied or terminated). An exception is thrown if the property resubmitEnabled is false.

### Parameters

None.

## Returns

None.

## Example

This is an example to resubmit a process:

```
import EOAW_CORE:*;  
  
Component EOAW_CORE:LaunchManager &launchMgr;  
  
&launchMgr.DoReSubmit();
```

## DoRestart

### Syntax

**DoRestart()**

### Description

Used to restart a currently running approval process. An exception is thrown if the property restartEnabled is false.

For example, in eProcurement, a requisition can be modified after has been approved. Since it was approved, Approval Framework sees it as closed (ended) and will not allow it to be resubmitted. The only option is to restart the transaction. Restart will reset all approval actions and route to the first approver regardless of the previous approvals.

### Parameters

None.

### Returns

None.

## PrepareToSubmit

### Syntax

**PrepareToSubmit()**

## Description

This method instantiates an approval process, but does not save or launch it. Repeated calls to this method do nothing, if an approval process has been instantiated (or if an approval process is running), this method simply returns. The point is that the requester can add ad-hoc approvers while previewing which will be included in the process launched by DoSubmit(). The DoSubmit(), DoResubmit(), DoPreview() and DoRestart() methods all call this method. So any approval process instantiated by DoPreview() will be reused by the above methods. When this behavior is not desired, application developers use the Reset() method.

---

**Note.** Repeated calls to prepare to submit can result in performance problems. Typically, this method is only used when ready to submit or when previews are necessary.

---

## Parameters

None.

## Returns

None.

## SetHeader

### Syntax

**SetHeader**(&hdr\_)

## Description

When creating a new application transaction, some applications do not fully populate the key fields until it is saved. For instance, eProcurement uses a generated id number for requisitions, but does not populate the id field until a new transaction is saved. Since applications need to initialize the LaunchManager object in the component post-build event code, this poses a challenge. In such situations, application developers can safely call this method to reset the header record before submitting a new transaction for approvals. It is always safe to call this method as long as the header record being passed in is valid. However when this method will re-instantiate the approval process, which eliminates any performance improvements achieved by caching previewed instances.

## Parameters

<i>Parameter</i>	<i>Description</i>
&hdr_	The header record defined for the approval processes in the Transaction Approval Levels in the Transaction Registry, as record.

## Returns

None.

## Example

```
&reqRecord = CreateRecord(Record.PV_REQHDR_AW_VW);  
GetLevel0()(1).GetRecord(Record.REQ_HDR).CopyFieldsTo(&reqRecord);  
&launchMgr.SetHeader(&reqRecord);  
&launchMgr.DoSubmit();
```

## Reset

### Syntax

**Reset()**

### Description

This method clears the approval process instance cached by the DoPreview() method. Approval process instances are cached between DoPreview() and DoSubmit() calls because a requester can add ad-hoc approvers to the previewed process before submitting. If ad-hoc approvers are added, using DoSubmit() to regenerate the approval process instance would cause the ad-hoc approvers to be lost. However, application developers must take care to call Reset() if the requester changes any significant fields between calls to DoPreview() and DoSubmit(). Since the Approval Framework does not know which fields are significant, this responsibility falls to the application developer. If concerned about catching all such situations, developers can adopt the strategy of always calling Reset() before calling DoSubmit() or any of its cousins. But this runs the risk of losing any ad-hoc approvers added by the requester during preview. Finally, if the application allows users to change the requester (for instance, when one user submits a transaction on behalf of another), the LaunchManager property *requester* needs to be updated. Updating the requestor property automatically calls the Reset() method.

### Parameters

None.

### Returns

None.

## FindDefinitionID

### Syntax

**FindDefinitionID()**

### Description

This method will return all definition IDs that have criteria matches. A developer can set the definition ID using the definitionID property. If a method such as DoSubmit or DoResubmit is called before this property is set, Approval Framework will use the first definition ID returned based on priority order.

### Parameters

None.

### Returns

Array of records.

## TerminateRunningProcess

### Syntax

**TerminateRunningProcess()**

### Description

Terminates a running process.

### Parameters

None.

### Returns

None.

---

## LaunchManager Class Properties

This section describes the LaunchManager class properties.

### hasTxn

#### Description

True if the application's process id valid.

### definition

#### Description

Set the definition and re-initialize the AppDef.

### hasAppDef

#### Description

True if an approval process has been defined.

### hasApplInst

#### Description

True if an approval process is currently running on this application transaction.

### hasEndedApplInst

#### Description

True if an approval process has previously been run on this application transaction.



## EOAW\_CORE:DEFN:AppDef appDef

### Description

The approval process definition currently active on this application.

## resubmitEnabled

### Description

Used to determine if this transaction can be re-submitted for approval. This will be true when:

1. A transaction registry entry exists.
2. An approval process has been defined.
3. One or more approval processes were launched on this transaction, but none are currently running (they were either approved, denied, or terminated).

### Example

```
If &launchMgr.resubmitEnabled = True Then
    &launchMgr.DoReSubmit();
End-If;
```

## submitEnabled

### Description

Used to determine if the submit action (which triggers launching of the Approval Framework) is currently enabled on this application transaction. Submit will be enabled when:

1. A transaction registry entry exists.
2. An approval process has been defined.
3. No approval process has previously been launched on this transaction.

### Example

```
If (&launchMgr.submitEnabled) Then
    &launchMgr.DoSubmit();
End-If;
```

## restartEnabled

### Description

Used to determine if this transaction's approval process be restarted. This will be true if an approval process is currently running.

## monitorEnabled

### Description

Used to determine if the Approval Monitor is enabled. The monitor is enabled if an approval process is running.

## previewEnabled

### Description

Preview is enabled if a process is not currently running on this transaction. The DoPreview() method only presents a preview of the approval process to be launched going forward. It is not meant for a review of prior processes.

---

**Note.** DoPreview does not display completed approval processes for this transaction.

---

## EOAW\_CORE:ENGINE:ApplInst applInst

### Description

The approval process instance currently running. Can be null.

## EOAW\_CORE:DEFN:AWTxn txn

### Description

The application transaction registry entry.

This property is read-only.

## requester

### Description

Use this property to get or set the requester on whose behalf this transaction is to be submitted for approval. Note that this property is not read-only. Applications may change the requester even after constructing the `LaunchManager`, but with certain caveats.

---

## Approval Manager Class

This class provides the methods to manage the approval process.

---

## ApprovalManager Class Methods

The approval manager object is the application developers interface to the Approval Framework. Most developers will not find it necessary to directly access any other Approval Framework objects. Application developers should instantiate this class as a component-scoped variable. The best place to initialize this class is in the component post-build event. The class can be instantiated with information readily available to developers at component post-build time. Developers should examine the boolean-valued properties of this object to know whether or not the user entering the approval component has any pending approvals work. If there are any pending approvals, the `GetPending()` method will identify what is pending, and the action methods such as `DoApprove()`, or `DoDeny()`, will enable them to implement the full approval functionality.

This section lists the Approval Manager class methods in alphabetical order.

## ApprovalManager

### Syntax

**ApprovalManager**(*&awprcs\_id,&hdr\_,&approver\_*)

### Description

Developers need to know their transaction id, and need to pass in an instance of their application's main record, which describes the request being submitted for approval. They also need to explicitly tell the Approval Framework who the approver is.

## Parameters

<i>Parameter</i>	<i>Description</i>
&awprcs_id	The approval Process ID that has been defined in the Transaction Registry, as string.
&hdr_	The header record defined for the approval processes in the Transaction Approval Levels in the Transaction Registry, as record.
&approver_	The Operator ID of the user approving the transaction, as string.

## Example

```
import EOAW_CORE::*;
Component EOAW_CORE:ApprovalManager &approvalMgr;
&vchrRecord = CreateRecord(Record.VCHR_AF_HDR_VW);
GetLevel0()(1).GetRecord(Record.VCHR_FS).CopyFieldsTo(&vchrRecord);
&approvalMgr = create EOAW_CORE:ApprovalManager("VoucherApproval", =>
&vchrRecord, %UserId);
```

## DoApprove

### Syntax

**DoApprove**(&rec)

### Description

Use this method to approve the transaction for the record.

### Parameters

<i>Parameter</i>	<i>Description</i>
&rec	The record defined for the approval processes in the Transaction Approval Levels in the Transaction Registry, as record.

## DoApproveRowSet

### Syntax

**DoApproveRowSet**(&rs)

## Description

Use this method to approve transactions for the entire rowset. If this is a line-level record, then all paths which had routed this record to the approver are advanced.

For example, if an expense report has 3 lines and you want to approve all 3 lines, you would insert the records into a rowset and pass that rowset to `DoApproveRowset`. Approval Framework will approve each line in that rowset. This avoids repeated calls to `Approve`.

## Parameters

<i>Parameter</i>	<i>Description</i>
<code>&amp;rs</code>	Rowset to approve, as rowset.

## Returns

None.

## DoReassign

### Syntax

**DoReassign**(*&rec, &reassignee, &bAllowAutoApprove, &bAllowSelfApprove, &comment*)

## Description

Use this method to reassign the record instance to a different approver.

## Parameters

<i>Parameter</i>	<i>Description</i>
<code>&amp;rec</code>	The transaction record to be reassigned, as record.
<code>&amp;reassignee</code>	The user to reassign the approval to, as string.
<code>&amp;bAllowAutoApprove</code>	Indicate whether or not auto approval is active, as boolean.
<code>&amp;bAllowSelfApprove</code>	Indicate whether or not self approval is active, as boolean.
<code>&amp;comment</code>	Include any comments, as string.

## Returns

String.

## DoReassignAll

### Syntax

**DoReassignAll**(*&reassignee, &bAllowAutoApprove, &bAllowSelfApprove, &comment*)

### Description

---

**Note.** Use this method to reassign all line items for the transaction to a different user for approval.

---

### Parameters

<i>Parameter</i>	<i>Description</i>
<i>&amp;reassignee</i>	User ID to reassign the transactions to, as string.
<i>&amp;bAllowAutoApprove</i>	Indicate whether or not auto approval is active, as boolean.
<i>&amp;bAllowSelfApprove</i>	Indicate whether or not self approval is active, as boolean.
<i>&amp;comment</i>	Include any comments, as string.

## Returns

String.

## DoDeny

### Syntax

**DoDeny**(*&rec*)

### Description

Use this method to deny approval on a record instance. If the record is a header record, then the approval process ends. If the record is a line-level record, that particular line's processing ends, while the other records in that transaction continue.

## Parameters

<i>Parameter</i>	<i>Description</i>
<code>&amp;rec</code>	The record name defined for the approval processes in the Transaction Approval Levels in the Transaction Registry, as record.

## Returns

None.

## Example

```
&approvalMgr.DoDeny(&reqRecord);
```

## DoDenyRowset

### Syntax

**DoDenyRowset**(*&rs*)

### Description

Use this method to deny approval on a rowset. This is used to enable Approval Framework to deny multiple lines at once rather than looping through each line.

## Parameters

<i>Parameter</i>	<i>Description</i>
<code>&amp;rs</code>	Name of the rowset, as rowset.

## Returns

None.

## DoDenyWithAllowUndeny

### Syntax

**DoDenyWithAllowUndeny**(*&rs*)

## Description

Use this method to deny approval on a rowset, where resubmits are not allowed.

## Parameters

<i>Parameter</i>	<i>Description</i>
<code>&amp;rs</code>	Name of the rowset, as rowset.

## Returns

None.

## DoHardDeny

### Syntax

**DoHardDeny**(*&rec*)

## Description

Use this method to deny approval on a record instance and not allow re-submits. If the record is a header record, then the approval process ends. If the record is a line-level record, that particular line's processing ends, while the other records in that transaction continue.

## Parameters

<i>Parameter</i>	<i>Description</i>
<code>&amp;rec</code>	The record name defined for the approval processes in the Transaction Approval Levels in the Transaction Registry, as record.

## Returns

None.



## DoLineResubmit

### Syntax

**DoLineResubmit**(*&rec, &start*)

### Description

Allows the line being passed in to be resubmitted to a running transaction. The user decides if the line is restarted from the current stage or the beginning. If it starts from the beginning, all history after the first header stage is lost.

### Parameters

<i>Parameter</i>	<i>Description</i>
<i>&amp;rec</i>	The record defined for the approval processes in the Transaction Approval Levels in the Transaction Registry, as record.
<i>&amp;start</i>	Use <i>True</i> to start at the beginning of the approval process or <i>False</i> to start at the current stage.

### Returns

None.

### Example

```
&approvalMgr.DoLineResubmit(&reqLnRecord, True);
```

## DoAddNewLine

### Syntax

**DoAddNewLine**(*&rec, &start*)

### Description

Use to allow the line being passed in to be added to a running transaction. The user decides if the line is restarted from the current stage or the beginning. If it starts from the beginning, all history after the first header stage is lost.

## Parameters

<i>Parameter</i>	<i>Description</i>
<code>&amp;rec</code>	The record defined for the approval processes in the Transaction Approval Levels in the Transaction Registry, as record.
<code>&amp;start</code>	Use <i>True</i> to start at the beginning of the approval process or <i>False</i> to start at the current stage.

## Returns

None.

## GetPending

### Syntax

**GetPending()**

### Description

This method returns a rowset in the form of the applications header or line record that includes any header row or line row that is pending.

### Parameters

None.

### Returns

Rowset containing pending transactions.

### Example

```
&LINERS = CreateRowset(Record.PV_REQLIN_AW_VW) ;  
&LINERS = &approvalMgr.GetPending() ;
```

## DoPushback

### Syntax

**DoPushback**(*&rec*)

### Description

Use this method to send the process back to the previous approver. This is used to give the approver a means of requesting a prior approver to make a clearer case for approving the transaction.

---

**Note.** Pushback only works to push back the workflow to a prior step in the same path. Calling Pushback() on the first step its path does nothing.

---

### Parameters

<i>Parameter</i>	<i>Description</i>
<i>&amp;rec</i>	The record defined for the approval processes in the Transaction Approval Levels in the Transaction Registry, as record.

### Returns

None.

### Example

```
&approvalMgr.DoPushback(&reqRecord);
```

## AddComments

### Syntax

**AddComments**(*&username, &rec, &comments*)

### Description

Use this method to add comments in the approval process.

## Parameters

<i>Parameter</i>	<i>Description</i>
&username	The user name for the current user, as string.
&rec	The record defined for the approval processes in the Transaction Approval Levels in the Transaction Registry, as record.
&comments	Comments for the approval transaction, as string.

## Returns

None.

## Example

```
&approvalMgr.AddComments(getUserName(%OperatorId), =>  
    &reqRecord, PV_REQ_APPPG_WK.COMMENTS_2000);
```

## TakeNoAction

### Syntax

**TakeNoAction**(&rec)

### Description

Use this method to update the user's status as no action taken. If all approvers at this step take no action, then the step will advance and the current step will store the status for No Action Taken.

## Parameters

<i>Parameter</i>	<i>Description</i>
&rec	The record defined for the approval processes in the Transaction Approval Levels in the Transaction Registry, as record.

## Returns

None.

## PutOnHold

### Syntax

**PutOnHold**(*&rec*)

### Description

Used to put the record instance passed in on hold for this step. If this is a line-level record, then all paths which had routed this record to the approver are advanced.

### Parameters

<i>Parameter</i>	<i>Description</i>
<i>&amp;rec</i>	The record defined for the approval processes in the Transaction Approval Levels in the Transaction Registry, as record.

### Returns

None.

## PutOnHoldCount

### Syntax

**PutOnHoldCount**(*&rec*)

### Description

This method is used to count how many approvers have put an approval transaction on hold. For example, if there are 10 approvers and 5 approvals are needed, checking to see that 3 approvers put it on hold will tell you that there are still 7 approvers that have not looked at the transaction yet.

### Parameters

<i>Parameter</i>	<i>Description</i>
<i>&amp;rec</i>	The record defined for the approval processes in the Transaction Approval Levels in the Transaction Registry, as record.

**Returns**

Array of string.

**GetPushedBack****Syntax**

**GetPushedBack()**

**Description**

Use this method to retrieve all the rows in a transaction that were pushed back.

**Returns**

Rowset.

**GetPertinentThreads****Syntax**

**GetPertinentThreads**(*&checkApprover, &userType*)

**Description**

Use this method to determine what threads are pending review or approval.

**Parameters**

<i>Parameter</i>	<i>Description</i>
<i>&amp;checkApprover</i>	As string.
<i>&amp;userType</i>	As string.

**Returns**

Rowset.

## GetStage

### Syntax

GetStage(&rec)

### Description

Use this method to retrieve the current stage of the approval process for a transaction.

### Parameters

<i>Parameter</i>	<i>Description</i>
&rec	The record defined for the approval processes in the Transaction Approval Levels in the Transaction Registry, as record.

### Returns

EOAW\_CORE:ENGINE:StageInst

## GetPendingSteps

### Syntax

GetPendingSteps()

### Description

Use this method to determine all pending steps for an approval process. This method returns a list of step-instance objects pending for this transaction.

### Parameters

None.

### Returns

Array of EOAW\_CORE:ENGINE:UserStepInst.

## DoLineTerminate

### Syntax

**DoLineTerminate**(*&LineRec*)

### Description

Use this method to terminate the given line level approval.

### Parameters

<i>Parameter</i>	<i>Description</i>
<i>&amp;LineRec</i>	The line level record for the approval process.

## GetParticipant

### Syntax

**GetParticipant**(*&username*)

### Description

Use this method to determine whether or not the user is a participant in the approval process. This method returns the participation status for a user that is currently pending or has taken an action.

### Parameters

<i>Parameter</i>	<i>Description</i>
<i>&amp;username</i>	User ID, as string.

### Returns

Returns a string to determine user participation in the approval process:

- *N*- if the user is not a participant in the approval.
- *AA*- if the user is the actual approval.
- *OA*- if the user is the original approval.



- *RR*- if the user is a reviewer.
- *O*- if the user is the originator.
- *R*- if the user is the requestor.

## GetAllActiveParticipants

### Syntax

GetAllActiveParticipants()

### Description

Use this method to retrieve a list of all users that have either been flagged as Approvers, Reviewers, requestor or initiator.

### Parameters

None.

### Returns

Array of string.

## RequestInformation

### Syntax

RequestInformation(*&user, &rs*)

### Description

Use this method to put a step on hold until the specified user provides more information.

### Parameters

<i>Parameter</i>	<i>Description</i>
<i>&amp;user</i>	UserID to insert as a reviewer and request information from, as string.
<i>&amp;rs</i>	Rowset to push the reviewer onto the step, as rowset.

**Returns**

None.

**SetAttributeObject****Syntax**

**SetAttributeObject**(&attrObj\_)

**Description**

Use this method to set the Approval Attribute class on the Appinst class.

**Parameters**

<i>Parameter</i>	<i>Description</i>
&attrObj	As EOAW_CORE:ENGINE:AppAttributes.

**Returns**

None.

---

**ApprovalManager Class Properties**

This section describes the ApprovalManager class properties.

**hasAppInst****Description**

True if the current transaction (header record) has a currently pending Approval Framework instance.

## definition

### Description

Get the current definition ID set on the App Def.

## level

### Description

The level of the pending approval work--*0* for header, *1* for line-level. The Approval Framework architecture guarantees that header and line-level approval work cannot be pending at the same time.

This property is read-only.

## hasPending

### Description

True if the user has any pending approval tasks for the current transaction.

---

**Note.** Typically this will be the signed on user, but it does not have to be.

---

## pushbackEnabled

### Description

This method is used to determine if there is any path that is pending with a step number greater than one. If there is a step greater than one, *True* is returned and push back is enabled.

This is a read-only property.

## EOAW\_CORE:ENGINE:ApplInst the\_inst

### Description

The current approval process instance.

## isReviewer

### Description

True if the user is a reviewer in the approval process.

This is a read-only property.

---

## Approval Event Handler Class

This class provides methods to monitor events and notify the approval framework.

---

## ApprovalEventHandler Class Methods

The ApprovalEventHandler class is used to perform actions that are required at the specific points in the process. For instance, if a new Employee has been approved, then the required action would be to activate that user. The purpose of this class is to move the check for specific statuses or actions from the Component to Approval Framework. This sections lists the ApprovalEventHandler class methods.

## ApprovalEventHandler

### Syntax

ApprovalEventHandler()

### Description

This method is the constructor for the ApprovalEventHandler class. Its only purpose is to create an object that can be used to access the other methods.

## OnProcessLaunch

### Syntax

OnProcessLaunch(*&appInst*)

## Description

This method is called with the newly-launched approval process instance, after the launch operation is successful.

## Parameters

<i>Parameter</i>	<i>Description</i>
<code>&amp;appInst</code>	As EOAW_CORE:ENGINE:AppInst.

## Returns

None.

## Example

```
method OnProcessLaunch
    /+ &appInst as EOAW_CORE:ENGINE:AppInst +/
    /+ Extends/implements EOAW_CORE:ApprovalEventHandler.OnProcessLaunch +/
    &appInst.thread.SetAppKeys(&vndrRecord);
    %This.updateProcessFlag(&PENDING);
end-method;
```

## OnStepActivate

### Syntax

**OnStepActivate**(*&stepinst*)

### Description

Use this method when a step instance is activated, the approvers and reviewers associated with the step will receive a worklist entry, if they do not already have one). The activated step instance itself is passed in as the argument.

## Parameters

<i>Parameter</i>	<i>Description</i>
&stepinst	The activated step instance, as EOAW_CORE:ENGINE:StepInst.

## Returns

None.

## OnStepHold

### Syntax

**OnStepHold**(&userinst)

### Description

Use this method to put this step on hold when a user performs an action. On Hold means that the user plans to take action at a later time.

## Parameters

<i>Parameter</i>	<i>Description</i>
&userinst	As EOAW_CORE:ENGINE:UserStepInst.

## Returns

None.

## OnStepReassign

### Syntax

**OnStepReassign**(&userinst,&origApprover)

### Description

Use this method to indicate the action when a step is reassigned.

## Parameters

<i>Parameter</i>	<i>Description</i>
&userinst	As EOAW_CORE:ENGINE:UserStepInst.
&origApprover	As string.

## Returns

None.

## OnStepComplete

### Syntax

**OnStepComplete**(&stepinst)

### Description

Use this method to indicate the action to take place when the step is complete.

## Parameters

<i>Parameter</i>	<i>Description</i>
&stepinst	The active step instance, as EOAW_CORE:ENGINE:StepInst.

## OnStepPushback

### Syntax

**OnStepPushback**(&userinst)

## Parameters

<i>Parameter</i>	<i>Description</i>
&userinst	As EOAW_CORE:ENGINE:UserStepInst.

## OnStepReactivate

### Syntax

**OnStepReactivate**(*&stepins*)

### Description

Use this method when a step is reactivated.

### Parameters

<i>Parameter</i>	<i>Description</i>
<i>&amp;stepinst</i>	The current step instance, as EOAW_CORE:ENGINE:StepInst.

## OnFinalHeaderDeny

### Syntax

**OnFinalHeaderDeny**(&appinst As EOAW\_CORE:ENGINE:AppInst);

### Description

Use this method for the final denial.

### Parameters

<i>Parameter</i>	<i>Description</i>
<i>&amp;appinst</i>	Application instance, as EOAW_CORE:ENGINE:AppInst.

### Returns

None.



## OnHeaderDeny

### Syntax

**OnHeaderDeny**(*&userinst*)

### Description

Use this method for header denial.

### Parameters

<i>Parameter</i>	<i>Description</i>
<i>&amp;userinst</i>	As EOAW_CORE:ENGINE:UserStepInst.

## OnHeaderApprove

### Syntax

**OnHeaderApprove**(*&appinst*)

### Description

Use this method to indicate the action to take when the header is approved.

### Parameters

<i>Parameter</i>	<i>Description</i>
<i>&amp;appinst</i>	Application instance, as EOAW_CORE:ENGINE:AppInst.

## OnNoApprovalNecessary

### Syntax

**OnNoApprovalNecessary**(*&appinst*)

## Description

Use this method when no approval is necessary.

## Parameters

<i>Parameter</i>	<i>Description</i>
&appinst	Application instance, as EOAW_CORE:ENGINE:AppInst.

## OnLineDeny

### Syntax

**OnLineDeny**(*&userstep*)

## Description

Use this method to indicate the action to take when a line is denied.

## Parameters

<i>Parameter</i>	<i>Description</i>
&userstep	User step, as EOAW_CORE:ENGINE:UserStepInst.

## OnLineApprove

### Syntax

**OnLineApprove**(*&appinst, &thread*)

## Parameters

<i>Parameter</i>	<i>Description</i>
&appinst	Application instance, as EOAW_CORE:ENGINE:AppInst.
&thread	As EOAW_CORE:ENGINE:Thread.

## OnAllLinesProcessed

### Syntax

**OnAllLinesProcessed**(*&appinst, &approved, &denied*)

### Description

Use this method when all lines are processed.

### Parameters

<i>Parameter</i>	<i>Description</i>
<i>&amp;appinst</i>	Application instance, as EOAW_CORE:ENGINE:AppInst.
<i>&amp;approved</i>	As array of EOAW_CORE:ENGINE:Thread.
<i>&amp;denied</i>	As array of EOAW_CORE:ENGINE:Thread.

## OnTerminate

### Syntax

**OnTerminate**(*&appinst As EOAW\_CORE:ENGINE:AppInst*);

### Description

Use this method when the application instance is terminated.

### Parameters

<i>Parameter</i>	<i>Description</i>
<i>&amp;appinst</i>	Application instance, as EOAW_CORE:ENGINE:AppInst.

## OnError

### Syntax

**OnError**(*&stepinst*)

### Description

Use this method when an error occurs in a step.

### Parameters

<i>Parameter</i>	<i>Description</i>
<i>&amp;stepinst</i>	Step instance, as EOAW_CORE:ENGINE:StepInst.

## OnStepReview

### Syntax

**OnStepReview**(*&stepinst*, *&reviewers*)

### Description

Use this method to indicate the action to take when the step instance is reviewed.

### Parameters

<i>Parameter</i>	<i>Description</i>
<i>&amp;stepinst</i>	Step instance, as EOAW_CORE:ENGINE:StepInst.
<i>&amp;reviewers</i>	UserIds for the reviewers, as an array of string.

## OnTimeout

### Syntax

**OnTimeout**(*&userinst*, *&notify*)

## Parameters

<i>Parameter</i>	<i>Description</i>
&userinst	As EOAW_CORE:ENGINE:UserStepInst.
&notify	As string.

## OnAdHocInsert

### Syntax

**OnAdHocInsert**(*&stepinst, &approver*)

### Description

Use this method when new adhoc reviewers are inserted.

## Parameters

<i>Parameter</i>	<i>Description</i>
&stepinst	As EOAW_CORE:ENGINE:AdHocStepInst.
&approver	UserID for approvers, as an array of string.

## OnAdHocDelete

### Syntax

**OnAdHocDelete**(*&stepinst*)

### Description

Use this method when an adhoc step is deleted.

## Parameters

<i>Parameter</i>	<i>Description</i>
&stepinst	As EOAW_CORE:ENGINE:AdHocStepInst.

## OnUserLocked

### Syntax

**OnUserLocked**(*&userinst*)

### Description

Use this method to indicate the action to be taken if a user is locked out. If a user is locked out, then the administrator should be notified. This acts more as a warning because the system will still route the approval to that person.

## Parameters

<i>Parameter</i>	<i>Description</i>
&userinst	As EOAW_CORE:ENGINE:UserStepInst.

## OnStepRequestInformation

### Syntax

**OnStepRequestInformation**(*&userinst*)

### Description

Use this method when there is a request for information.

## Parameters

<i>Parameter</i>	<i>Description</i>
&userinst	As EOAW_CORE:ENGINE:UserStepInst.

## OnRequestInformationAdded

### Syntax

**OnRequestInformationAdded**(*&appinst* As *EOAW\_CORE:ENGINE:AppInst*, *&level* As *number*, *&notifyList* As *array of string*);

### Description

Use this method when request information has been added.

### Parameters

<i>Parameter</i>	<i>Description</i>
<i>&amp;appinst</i>	Application instance, as EOAW_CORE:ENGINE:AppInst.
<i>&amp;level</i>	Approval level, as number.
<i>&amp;notifyList</i>	UserIds to notify, as array of string.

## ProcessNotifications

### Syntax

**ProcessNotifications**(*&appinst*)

### Description

Use this method to process notifications.

### Parameters

<i>Parameter</i>	<i>Description</i>
<i>&amp;appinst</i>	Application instance, as EOAW_CORE:ENGINE:AppInst.

## OnLineTerminate

### Syntax

**OnLineTerminate**(*&appinst,&thread*)

### Description

Use this method when a line is terminated.

### Parameters

<i>Parameter</i>	<i>Description</i>
<i>&amp;appinst</i>	Application instance, as EOAW_CORE:ENGINE:AppInst.
<i>&amp;thread</i>	As EOAW_CORE:ENGINE:Thread.

---

## ApprovalEventHandler Class Properties

This section lists the ApprovalEvent Hander Class properties.

### wlPrefix

#### Description

Worklist prefix.

### wlBusinessProc

#### Description

Business process.



## **wlActivityName**

### **Description**

Activity name.



# Index

## A

- ad hoc
  - approvals 11
  - review 11
- alternate user ID, setting 55
- alternate workflow approvers 13
- application class criteria for Approval Framework 24
- application developers, setting up Approval Framework 6
- Application Engine Request page 77
- approval
  - authorizing 60, 62
  - comments 14
  - header level 9
  - line level 9
  - managing workflow 5
  - process description 7
  - workflow steps 14
- Approval Authorization page 63, 64
- ApprovalEventHandler class, use 41
- Approval Monitor Configuration page 96
- Approval Path Definition page 18, 30, 63, 65
- Approval Process component (EOAW\_PRCs) 18
- Approval Process Definition page 18
- Approval Step Definition page 18, 33
- approval transaction registry
  - configure transactions 43, 44
  - overview 37
  - prerequisites 37
  - registering 38
- approver
  - alternate 13
  - role name 33
  - user list 33
- auto approval 13

## C

- comments, approvals 14
- common fields xii
- Configure Transactions page 44, 77, 79
- Criteria Definition page 18, 63
- Criteria Definition Page 23
- cross-reference table, approvals 40

## D

- dynamic paths 59
- dynamic user role 54

## E

- Email Addresses page 77

- Email Address page 79
- email collaboration 67
- email collaboration, setting up 67
- end-user roles, Approval Framework 7
- end actions for approvals 9
- EOAW\_PRCs component 18
- escalation
  - event setup 92
- Event Status page 91
- Event Type page 92
- Event Types page 91

## F

- field mapping 76
- Field Mapping page 74, 76
- Form Element Designer page 74
- Form Layout Designer page 74
- functional business analysts, setting up Approval Framework 6

## G

- General page - Service Operation Definition 71
- General page - User Profile 63
- Generic Template Definition page 51
- getting started 1

## H

- Handler Details page 72
- Handler page - Service Operation Definition 72
- header approval 9, 43

## L

- line-level approval
  - overview 9
  - setting 43

## M

- Message Definition page 71
- monetary alert criteria, defining 29
- Monitor Approvals page 101
- Monitor Approvals page - User Monitor 106
- Monitor Approvals page
  - (EOAW\_ADM\_MON\_ACT) 99
- Monitor Approvals page
  - (EOAW\_ADM\_MON\_SRC) 98, 99

## N

- Node Definitions page 70
- Node Properties page 70
- notification and escalation
  - overview 91
- notification and escalation manager
  - setting up 93
- Notification and Escalations page 91, 92
- notifications
  - approval events 13
  - approval process 5
  - configure transactions 43, 44
  - default options 40
  - defining template 51
  - email 5
  - URL 51
  - worklist 5
- Notifications page 18

## P

- paths
  - criteria 23
  - dynamic 59
  - in approvals 7
- prerequisites for registering an approval
  - transaction 37
- process scheduler, escalation setup 93
- PSFT\_EMG\_GETMAIL Node 70
- push back, approvals 14

## R

- reassign work 56
- Recurrence Definition page 77
- registering approval transactions, prerequisites 37
- Register Transaction page 77
- Register Transactions page 38, 77
- requisition approval
  - notification and escalation manager
    - NEM 91
- reviewer user list 34
- Roles page - User Profiles 53
- route control, workflow 54

## S

- Schedule JobSet Definitions page 91, 93
- self-approval
  - overview 11
  - setting up 34
- Services page 71
- Setup Process Definition page 17
- Setup Process Definitions page 17, 63
- stage, approvals 7
- static user role 54
- steps, Approval Framework 14
- step source 31
- supervising user ID, setting 55

## T

- thread, in approvals 7, 9
- Timeout Options page 18
- transaction key input to approvals 57

## U

- URL Maintenance page 51
- user input to approvals 57
- User List Definition page 53, 56, 63

## V

- view definition for roles 54

## W

- workflow 5
  - alternate approvers 13
  - defining path criteria 23
  - defining steps for 32
  - defining workflow paths 30
  - defining workflow processes 17, 18
  - roles 6
- Workflow page - User Profiles 53, 54
- worklist, using in approvals 41