AppsFlow

User's Guide

Software Version 7.1

LogicalApps®

Restricted Rights Legend

Logical Apps provides on-site support as well as remote phone and web support to ensure quick and effective product implementation. To request support, to suggest product enhancements, or to comment on Logical Apps software or documentation, send email to support@logicalapps.com, or contact us at the address or phone number given below.

Document AR007-710B

8/20/07

Logical Apps
15420 Laguna Canyon, Suite 150
Irvine, CA 92618
949.453.9101

Contents

About AppsFlow 1

Architecture 2

Prerequisites 2

AppsFlow in Other AppsRules Applications 2

Starting AppsFlow 2

Creating a Process 3

Using the Process Details Panel 3

Using the Primary Keys Panel 4

Using the Display Table/View Columns Panel 5

Using the Effectivity Dates Panel 6

Configuring Launch Criteria 7

Trigger Criteria 8

Periodic Criteria 9

Business Event Criteria 10

Configuring Elements Called by Rules 11

Using the Advanced Rules Wizard 11

Starting an Advanced Rule 12

Using the Definition Panel 13
Using the Return Columns Panel 15
Using the SQL Panel 16
Using the Notification Panel 17
Creating a Notification Function 18
Creating an Approval Group 19
Creating a Workflow Role 21
Creating Process Flows 23
Starting a Process Flow 23
Configuring Constraint Rules 25
Configuring Concurrent Program Rules 26
Configuring Approval and Notification Rules 28
Configuring Exception Rules 30
Configuring SQL Rules 31
Configuring Processes Flows 32
Configuring Workflow/Event Rules 33
Adding Conditions to Process Flows 34
Testing Process Rules 37
Integration with Oracle Workflow 39
AppsFlow Migration 41
Preparing for Migration 41
Stipulations 43
Migrating, Exporting, or Copying Rules 43
Importing a Process or a Flow 45
Running AppsFlow Utilities 47
Monitoring Constraint Processes 47
Gathering Debug Data 48
Collecting Processes in Libraries 48
Creating a Library 49
Adding to Value Sets 50
Using the Mass Associate Utility 52
Monitoring Concurrent Programs 52

Chapter 1

About AppsFlow

AppsFlow defines and implements business processes — sets of actions to be completed in specified sequences. An AppsFlow process may serve as a component of an Oracle Applications Workflow or may run on its own.

In AppsFlow, a single "process rule" defines an entire process. The rule consists of subordinate rules, called "process flows," each of which constitutes a step in the process. Each flow is assigned a "rule type" that determines what it does. A process flow can:

- Notify, or request approval of, a designated person when some action has been completed.
- Alert designated persons to errors or other exceptional conditions.
- Implement a "constraint," which alerts designated persons if necessary conditions have not been met, and pauses the process pending a response.
- Run a concurrent program, or monitor one as it runs.
- Run structured query language (SQL) scripts.
- Link the current process to other processes.
- Run separately defined workflows or events within a process.

AppsFlow users can choose whether a process rule is to be triggered by an event or evaluated on a regular schedule, and can configure "launch criteria" — either define the triggering event or set the schedule.

Moreover, AppsFlow rules may call elements required for a process to be completed — for example, an approval group if a process flow requires an action to be approved or rejected. AppsFlow provides an "Advanced Rules Wizard" and other tools for use in configuring notification text, SQL scripts, approval groups, workflow roles, and other elements called by process rules. It also provides a tool for "migrating" process rules from one Oracle Applications instance to another.

Architecture

AppsFlow is completely integrated with Oracle Applications. It stores process rules in the Oracle database, and it uses the Oracle Workflow product to execute process rules.

AppsFlow does not replace, but rather enhances Oracle Workflow by giving the user the power to create complex flows without having to write the code that goes in Workflow process nodes. AppsFlow processes can be used either in existing Workflows or on their own.

Prerequisites

You are assumed to have a basic understanding of the Oracle Applications modules for which your organization is deploying AppsFlow. Although the creation of process rules does not require a knowledge of programming languages, AppsFlow tools permit the direct manipulation of structured query language (SQL) code, and a knowledge of SQL is helpful. Moreover, you are expected to have some knowledge of the relationships among tables and views (and their primary keys) in your Oracle database.

AppsFlow in Other AppsRules Applications

AppsFlow includes several rules that support processing in AppsAccess and AppsControl: APPSACCESS User Flow, AppsControl Approval, AppsControl Notification, and AppsControl Reject. Do not alter or delete these rules.

Starting AppsFlow

To open AppsFlow:

1. Select the Logical Apps AppsRules responsibility in the Oracle Applications list. (Ensure first that the AppsRules responsibility is available to you.)
2. A Logical Apps — AppsRules form appears. In it, click on the AppsFlow tab.

If you close the AppsRules form, you can reopen AppsFlow:

1. In the Logical Apps Navigator, click on the AppsRules option and then on the Open button. (Or, double-click on the AppsRules option.)
2. Once, again, click on the AppsFlow tab in the AppsRules form.

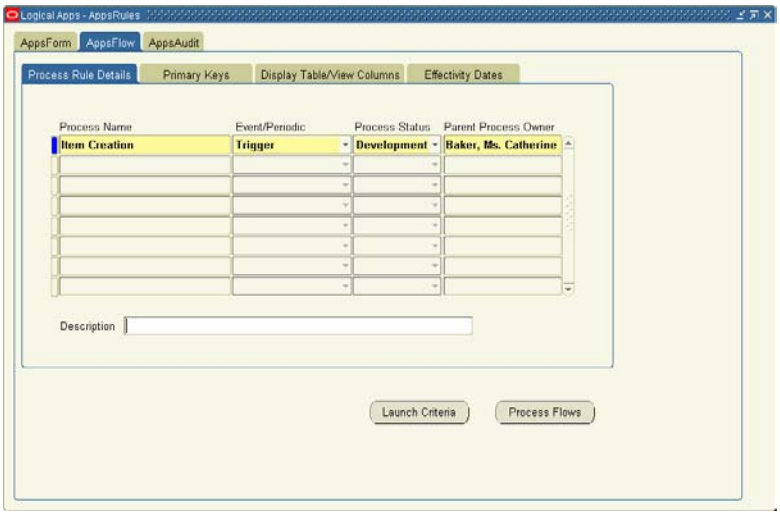
Chapter 2

Creating a Process

The first step in creating a process is to set up the process rule itself. Subsequently, you will define process flows and launch criteria for the rule (and later chapters discuss the creation of these elements). When you start AppsFlow, it opens to the form in which you define (or view) a process rule. The form consists of four panels, each accessible from a tab.

Using the Process Details Panel

To create a new process, ensure that the Process Rule Details tab is selected:



In this panel, complete the following steps:

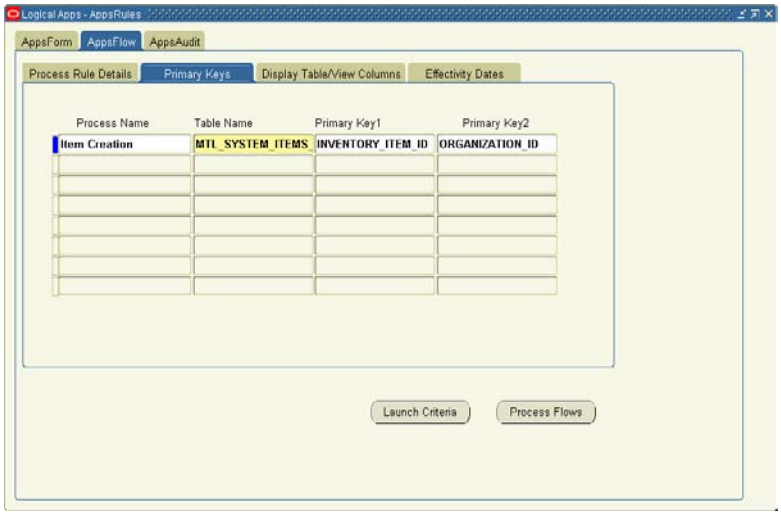
1. Activate a row in the grid. Use any of these methods:
   - If the grid contains any empty rows, click in the first one.
   - Click on the New button, which is first on the left in the tool bar.
   - Click on File in the menu bar, then on New in the File menu.

   Once a row is active, some fields on the form take on a yellow coloring, and others white. Yellow fields require input; white fields are optional.
2. In the Process Name field, type a unique name for the process rule.
3. In the Event/Periodic list of values, select a "subscription type," which determines what causes a process rule to be run. Choose one of the following:
   - Trigger: The process rule runs when some action occurs, and as a result a record is created or updated in a specified database table. (For example, a new customer may be created in a table that stores information about customers.)
   - Business Event: The process rule fires when a business event occurs. (Business events are entities seeded by Oracle in its Workflow system.)
   - Periodic: The process rule is evaluated on a regular schedule.
4. In the Process Status list of values, specify whether the rule is under development or deployed in a production environment: Select either of the following:
   - Development: When status is set at this level, workflow communications —such as notifications, approval requests, and emails —are sent only to a user identified as a "parent process owner."
   - Production: When status is set at this level, workflow communications are sent to the actual users or groups defined in process flows.
5. In the Parent Process Owner list of values, select the person who is to receive workflow communications if the process status is set to Development.
6. In the Description field, briefly explain the purpose of the process rule. (The use of this field is optional.)

Using the Primary Keys Panel

If you selected the Trigger subscription type in the Process Rules Details panel, you need to specify the table or view upon which rule firing depends. To do this, click on the Primary Keys tab (see the illustration at the top of the next page).

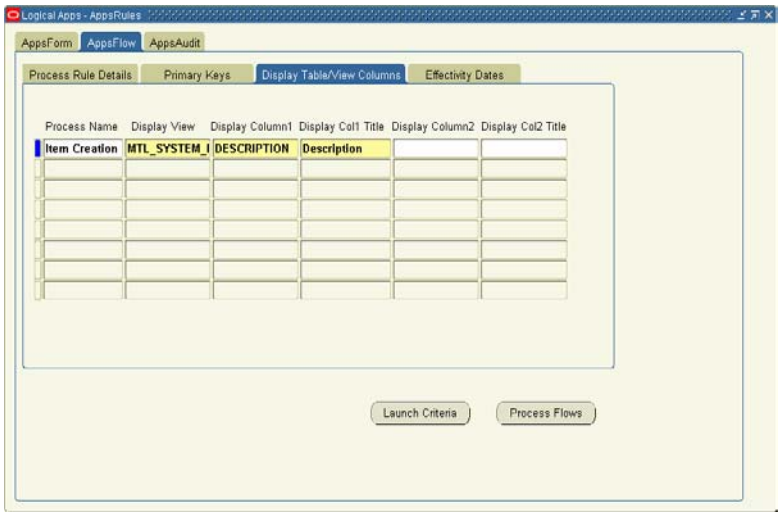
1. The Process Name field is completed automatically; it contains a copy of the value you entered in the Process Rules Details panel. Don't change it.
2. In the Table Name list of values, select the table or view in which the creation or updating of a record causes the rule to fire.
3. As you navigate to the Primary Key 1 and Primary Key 2 fields, AppsFlow inserts the primary keys defined for the table you selected in step 2. Don't change them.



The Primary Keys panel does not apply, and accepts no input, if you selected Business Event or Periodic in the Event/Periodic field of the Process Rules Details panel.

Using the Display Table/View Columns Panel

If you selected the Trigger subscription type in the Process Rules Details panel, you must also specify one or two "display columns," which contain information to be included in the subject lines of communications associated with Constraint process flows (which notify users if specified conditions have not been met). To do so, click on the Display Table/View Columns tab:



In this panel:

1. The Process Name field automatically contains a copy of the value you entered in the Process Rules Details panel. Don't change it.
2. The Display View field contains, by default, the table or view you selected in the Primary Keys panel. You can change the value to any related view.

3 In the Display Column 1 list of values, select a column containing information that identifies records within the table — for example, a customer-name column in a table that contains information about customers.

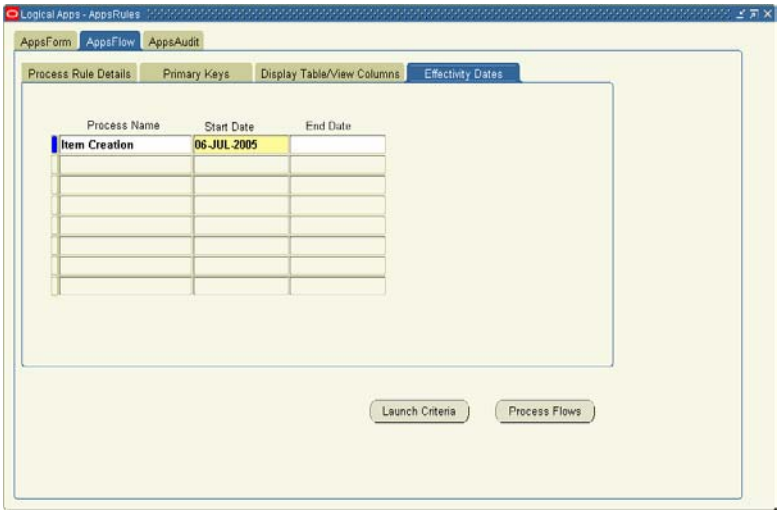
4 In the Display Col 1 Title field, type a label that describes the information from the column you selected in step 3 — for example, "Customer Name."

5 In the Display Column 2 and Display Col2 Title fields, you may select a second column to supply information to the subject line of Constraint-flow communications. (The second column is optional; the first is required.)

The Display Table/View Columns panel does not apply, and accepts no input, if you selected Business Event or Periodic in the Event/Periodic field in the Process Rules Details panel.

Using the Effectivity Dates Panel

Finally, click on the Effectivity Dates tab to specify when the rule is to be active:



In this panel:

1 The Process Name field is once again completed automatically; it contains a copy of the value you entered in the Process Rules Details panel. Don't change it.

2 Specify start and end dates.

The Start Date field defaults to the date on which you create the rule. Accept the default or modify it to specify a future date on which you want the rule to take effect.

The End Date field is blank by default. Insert the date on which you want the rule to expire, or leave the box blank to allow the rule to remain in effect indefinitely. An end date earlier than a start date prompts an error message.

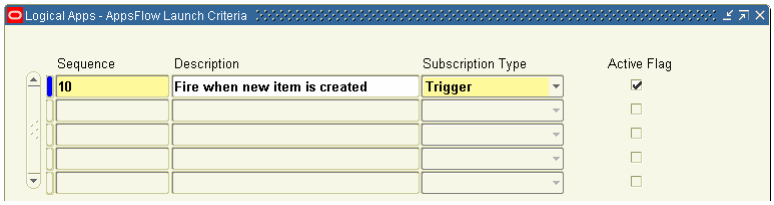
In either case, you can select a date in the pop-up calendar that appears when you click on the list-of-values icon. Or type a date in the format configured for your instance of Oracle Applications.

Chapter 3

Configuring Launch Criteria

To define circumstances under which the process defined by a rule is set in motion, click the Launch Criteria button in the main AppsFlow form. A Launch Criteria form opens.

Its content varies according to the subscription type you chose in the Event/Periodic field of the Process Rule Details panel on the main AppsFlow form. No matter what subscription type is chosen, however, the upper portion of the Launch Criteria form always contains this grid:



Enter values in at least one row of this grid:

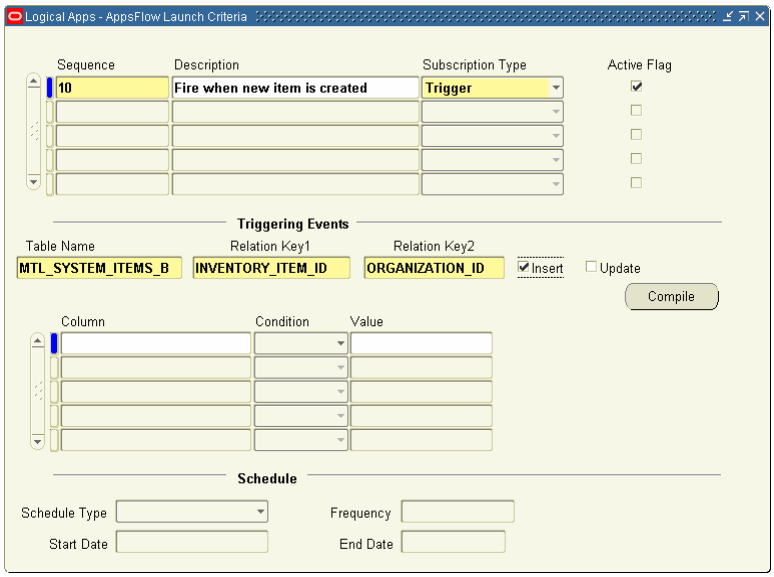
1. In the Sequence field, enter a number that reflects the order in which you want this row to be processed with respect to any other rows that contain content. (Enter a number even if no other rows contain content.)
2. In the Description field, type a brief explanation of the launch criteria you are configuring. (The use of this field is optional.)
3. In the Subscription Type field, AppsFlow copies the value that was selected in the Event/Periodic field of the Process Rule Details panel on the main AppsFlow form — Trigger, Periodic, or Business Event. You cannot change this value.

4 The Active Flag check box is selected by default. Leave it selected for the launch criteria you are configuring to remain active, or clear the check box to define launch criteria but hold them in reserve.

For each row you complete in this upper grid, you can select a new set of values that define a trigger, periodic schedule, or business event (as appropriate). If you complete more than one row in this upper grid, the rows have an AND relationship — the launch criteria defined for all active rows must evaluate to true for the process rule to fire.

Trigger Criteria

If you chose the Trigger subscription type (in the Event/Periodic field of the Process Rule Details panel on the main AppsFlow form), the Launch Criteria form looks like this:



For each row you complete in the upper grid, you would enter values in the Triggering Events section of the form. (The Schedule section of the form is inapplicable and its fields do not accept input.) To fill in the Triggering Events section of the form:

1 In the Table Name field, specify the name of a table in which the creation or updating of a record causes the rule to fire.

You have already specified such a table (or view) in the Primary Keys panel of the main AppsFlow form. The selection you make here is related to the selection you made there. In the main AppsFlow form, you had the option of selecting a table or view. Here, however, you can select only a table. Therefore:

- If the Table Name field on the Primary Keys panel of the main AppsFlow form displays a table, select the same table here.
- If the Table Name field on the Primary Keys panel of the main AppsFlow form displays a view, select a table associated with that view here.

2 In the Relation Key 1 list of values, select a table column that identifies individual records in the table — typically, the first primary key.

3 In the Relation Key 2 list of values, select a second table column that identifies individual records in the table — typically, the second primary key — if one exists.

4 Select either or both of the Insert and Update check boxes to trigger the process rule to fire if a table record is created or updated, respectively. (Click on a check box to select or clear it; a box is selected when a check mark appears.)

You may define additional conditions that must be met for the rule to fire. To do so, use the grid in the lower portion of the Triggering Events section of the form. A condition is, in effect, a logical statement specifying a value (or values) that table cells may hold; the rule would be triggered if the logical statement evaluates to true.

1 In the Column list of values, select a column from the table you selected in the Table Name field.

2 In the Condition field, select a logical operator — Is Updated, Is Null, Not Null, Equals, or Not Equals.

If you selected Is Updated, Is Null, or Not Null, the condition is complete (and a third field in the grid, Value, does not accept input). For example, if you select Is Null, the condition would evaluate to true (and so the process rule would fire) for records whose cells within the specified column contain no content.

3 If you select Equals or Not Equals as the operator, enter a value in the Value field. (Enclose an alphabetic value in single quotation marks.) The condition evaluates to true for values in cells from the column you selected in step 1 that either do or do not match the constant you enter here (depending on the operator you chose).

You may elect not to define conditions. Or, you may define as many conditions as you like, one in each row of the grid. If you define more than one, the conditions have an AND relationship — all must be true for the process rule to fire.

When you finish defining triggering criteria, click on the Compile button. A message displays status — either reports errors or announces that the trigger has been compiled correctly. Click on the OK button to clear the message.

Periodic Criteria

If you chose the Periodic subscription type (in the Event/Periodic field of the Process Rule Details panel on the main AppsFlow form), the Launch Criteria form looks as it did for the Trigger subscription type (see page 8), except the Schedule section of the form is now active, while the Triggering Events section is inapplicable and its fields do not accept input. To create a schedule on which the process rule runs:

1 In the Schedule Type box, select Hourly, Minutes, Days, or Monthly to designate the unit of time you use to define an interval at which the process rule is to run.

2 In the Frequency box, type a number that expresses the interval at which the process rule is to run. For example, if you type 5 here and select Hourly in the Schedule Type box, the process rule is evaluated every five hours.

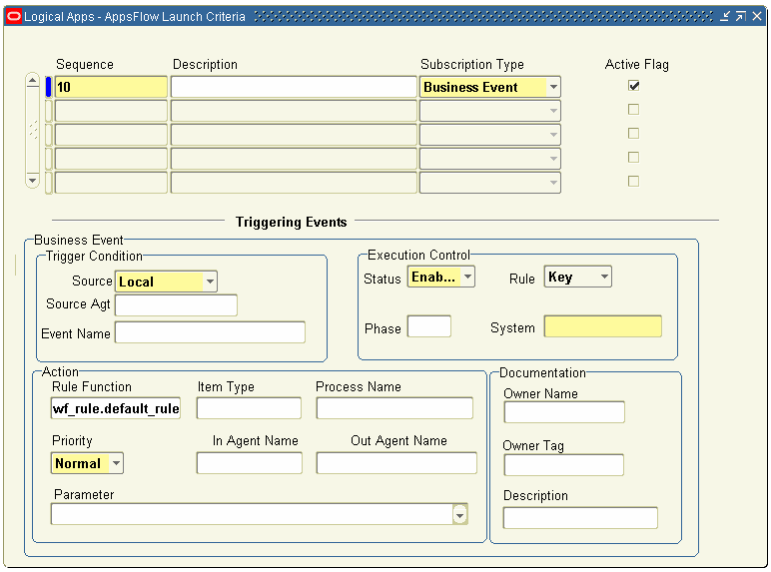
3 In the Start Date and End Date boxes, type or select dates and times that mark the beginning and end of the period in which the process rule should be run. Use

the format configured for your instance of Oracle Applications. (If you want the process rule to remain in effect indefinitely, leave the End Date box blank.)

4 Save your changes: Click on File in the menu bar, then on Save in the File menu. (If you do not save the changes, you are prompted to do so when you close the Launch Criteria form.)

Business Event Criteria

If you chose the Business Event subscription type (in the Event/Periodic field of the Process Rule Details panel on the main AppsFlow form), the Launch Criteria form looks like this:



For each row you complete in the upper grid, you would enter values in the Triggering Events section of the form:

1 Select the business event that will call AppsFlow:

- Business events are raised by forms of workflow processes.
- These business events are sent to an advanced queue.
- The workflow manager processes the events that are waiting in the advanced queue and if any workflows are tied to the event the workflows are initiated.

2 Select Actions/Subscription for this event:

- The AppsFlow API generates when a business event subscription is created on the AppsFlow from a subscription for that event.
- When a form raises that business event or a workflow, the AppsFlow process gets launched.
- The fields on this tab are similar to the standard Oracle "Create Subscriptions" from under the workflow responsibilities.

Chapter 4

Configuring Elements Called by Rules

A process rule consists of process flows, and each flow calls "elements" needed for the process to be implemented. An element may be a message that notifies (or requests approval of) designated reviewers when an action has been completed; a list of people (an approval group, a workflow role, or both) who would receive such messages; a constraint to be met before a process can continue; a SQL script to be executed; or other items.

Such elements must exist to be called from process flows, so you should create the elements you need before you configure process flows. Use the Advanced Rules Wizard for most of these elements; the exceptions are approval groups and workflow roles, for which you use tools available from the AppsFlow menu.

Using the Advanced Rules Wizard

In an Advanced Rules Wizard form, you can create additional rules that define elements called by process flows. Each of these rules is implemented (at least in part) as a SQL SELECT statement, and in most cases the Advanced Rules Wizard generates SQL code automatically from values entered into a user interface.

Depending on the type of element you are creating (and therefore on a "rule type" you choose as you create that element), the Advance Rules Wizard may consist of up to four panels, each accessible from a tab. These panels include:

- Definition, in which you select a "driving table" and may select other tables related or joined to it. An advanced rule is based on the driving table, which

provides information needed for an element to be configured. (It and tables joined to it might, for example, provide data to be included in a notification message.)

- Return Columns, in which you select columns from the driving table and any joined tables. These columns supply information to the element being configured.
- SQL, in which you can generate SQL code automatically from the values you selected in the Definition and Return Columns panels. In the SQL panel, you can also edit code manually and verify it.
- Notification, in which you write the text for notification or approval messages. This panel enables you to insert into these messages the names of columns selected by your SQL code; the resulting messages contain case-specific values returned by the SQL.

Starting an Advanced Rule

To use the Advanced Rules Wizard to create an element for use in a process rule:

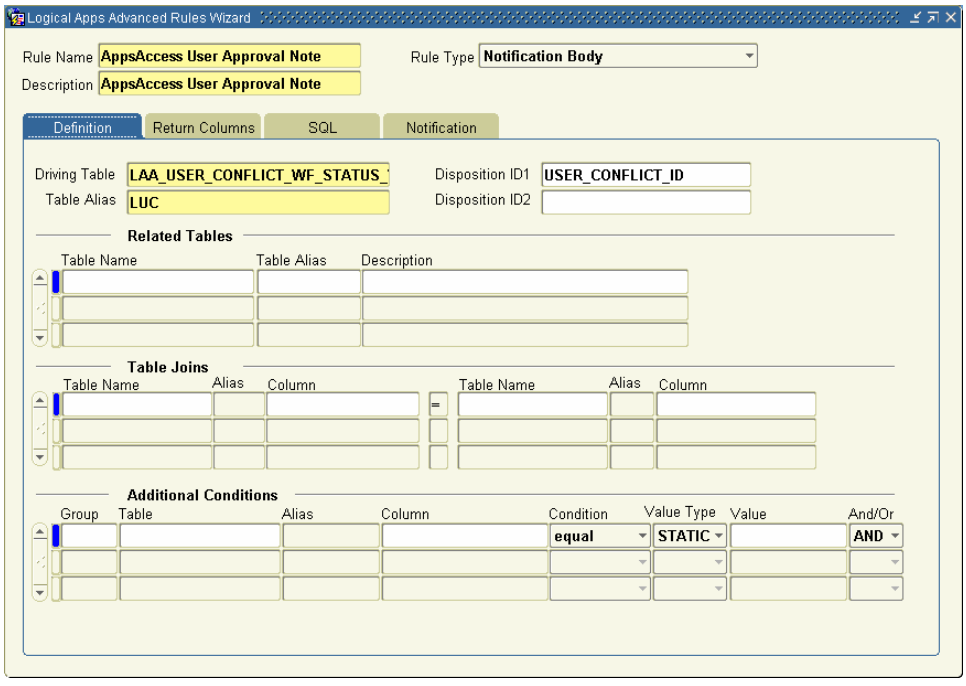
1. With AppsFlow running, click on LogicalApps Utilities in the menu bar, and then on Advanced Rules Wizard in the LogicalApps Utilities menu. The Advanced Rules Wizard opens with the Definition panel active. (See the illustration on page 13.)
2. In the Rule Name field, type a unique name for the element you are creating.
3. In the Description field, type a brief explanation of the element you are creating.
4. In the Rule Type list box, select the type of element you are creating:
   - Constraint/Condition: This element specifies values that may be held in one or more columns of a driving table (or related tables), and evaluates to true for records that either do or do not contain the specified values.

     It may be the basis of a Check Constraints process flow — one that determines whether appropriate data has been provided before allowing a process to continue. Or it may be added to any type of process flow as a way of selecting records that are subject to the flow.
   - Exception: This element defines an error or exceptional condition that must be addressed.
   - Sql Statement: This rule type creates a SQL statement that is to be run as part of a process. (One use for a Sql Statement element is to create a "resubmission program" that selects data to resolve an exception; see page 30.)
   - Notification Body: This element specifies the text (and column values) to be included in messages that notify or request approval of designated reviewers when some action has been completed.
   - Runtime Functional Owner Sql: This element constructs a SQL statement to select, at run time, the "functional owner" for a process flow. This is the person (or role) who receives communications generated by a process flow and, when appropriate, responds to them. The Runtime Functional Owner SQL must return a value that corresponds to a RESPONSIBILITY_ID or USER_ID key.

- Runtime Timeout Hours: This rule type constructs a SQL statement to define, at run time, a period in which a response must be given for a process flow that requests approval of reviewers when some action has been completed.
- TimeOut Sql: This rule type constructs a SQL statement to be executed when an approval times out.
- Reject Sql: This element constructs a SQL statement to be executed if a designated reviewer rejects an approval request in an Approvals process flow.
- Runtime Approval Group Sql: This element constructs a SQL statement to select, at run time, a group of reviewers for Approvals process flows. It must return a value corresponding to an APPROVAL_GROUP_ID defined under the Define Approval Groups option of the AppsFlow menu (see page 19).
- Single Return Function: This element returns a value that may be used in place of a Reject Sql or Runtime Approval Group Sql element, or as a runtime parameter for a concurrent program (see page 28).
- Runtime Responsibility: This rule type selects, at run time, a responsibility from which a concurrent program may be run in a Concurrent Programs process flow.
- Runtime User : This rule type selects, at run time, a user who can run a concurrent program in a Concurrent Programs process flow.
- AppsRules SQL: This element is not used in AppsFlow.

Using the Definition Panel

The Definition panel is available for most rule types. Use it to select tables from which an advanced rule is to draw information:



Always select a driving table and values related to it:

1. In the Driving Table field, select a table upon which the advanced rule is based.

   If you are creating an element to be called by a process flow that belongs to a process rule with a Trigger subscription type, the driving table (or view) must be the same as the table or view you selected for the process rule in the Primary Keys panel of the main AppsFlow form (see page 4). If the subscription type is Periodic or Business Event, you can select any table or view.

2. In the Table Alias field, create an alias for the table.
3. Optionally, in the Disposition ID 1 and 2 fields, enter the names of columns that distinguish a given record in the table from others (typically the primary keys).

If the rule is to use information stored in a table related to the driving table, select appropriate values in the Related Tables and Table Joins grids:

1. Devote one row of the Related Tables grid to each table you want to specify: Select its name from the Table Name list of values, create an alias in the Table Alias field, and optionally provide a description in the Description field.
2. For each row in the Related Tables grid, enter values in a corresponding row of the Table Joins grid. These values specify how each of the related tables is joined to the driving table or one of the other tables. Select the names and aliases of the two joined tables as well as the column in each that contains join values.

In the Additional Conditions grid, you may construct logical statements specifying values that table cells may hold. Such statements establish criteria by which some records in the driving table (or related tables) are distinguished from others — they either contain the specified values, or they don't.

Conditional statements are appropriate for rule types whose purpose is to select records that meet certain criteria. For example, a conditional statement defines the constraint that is the reason for creating a Constraint/Condition rule, or the exception that is the purpose of an Exception rule. You are expected to create at least one condition for any such rule type (even though conditions are technically "optional").

Conversely, conditional statements are inappropriate for rule types that do not filter records. For example, the Notification Body rule type designates columns that supply information to notification messages; it may be applied to any record in the driving (or related) tables. You are expected not to create conditions for any such rule type.

To create a condition:

1. In the Table and Alias fields, select values for the driving table or a related table.
2. In the Column list of values, select a column from the table.
3. In the Condition field, select a logical operator — Equal, Not Equal, Greater, Lesser, Is Null, or Is Not Null.

   Either of the Is Null or Is Not Null entries completes a logical statement (so if you select either, a third field, Value, does not accept input). For example, if you select Is Null, the condition evaluates to true for records whose cells within the specified column contain no content.

4 If you selected the Equal, Not Equal, Greater, or Lesser operator, specify a value to be compared with values contained in the column you selected in step 2.

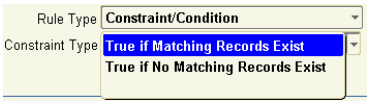
- To specify a constant value, select Static in the Value Type list box and type the constant value in the Value field.
- To specify the value of a field in an Oracle Applications form, select Form Field in the Value Type list box and the name of the field in the Value field.

For example, if you select the Greater operator, the Static value type, and the number 15 in the Value field, the condition evaluates to true for records whose cells within the specified column contain a number greater than 15.

5 Enter a number in the Group field. All rows with the same number are grouped together (enclosed in parentheses in the SQL statement).

6 If you create more than one condition, select AND or OR to determine how each condition (row) relates to other rows (or groups, or rows within groups). If two entities are joined by AND, both must by true; if by OR, either may be true.

If you are creating a Constraint/Condition rule, a Constraint Type list box appears beneath the Rule Type list box:

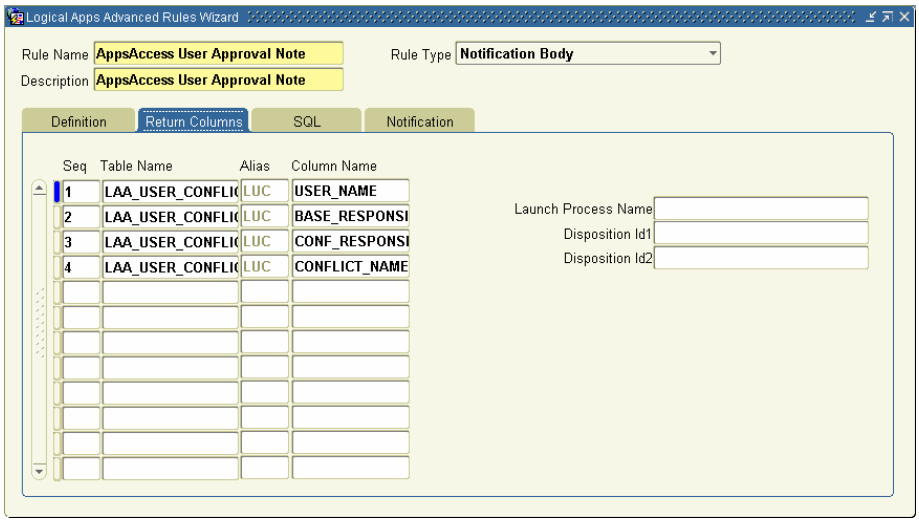


In this box, select either True if Matching Records Exist or True if No Matching Records Exist to determine whether the rule evaluates to true for records that either do or do not contain values specified in conditions you create for the rule. This list box is active only if you select the Constraint/Condition rule type.

Before continuing to the next panel, save the rule: Click on File in the menu bar and Save in the File menu.

Using the Return Columns Panel

The Return Columns Panel is available for rule types that return data values. Use it to select columns that return data from the tables chosen in the Definition panel:



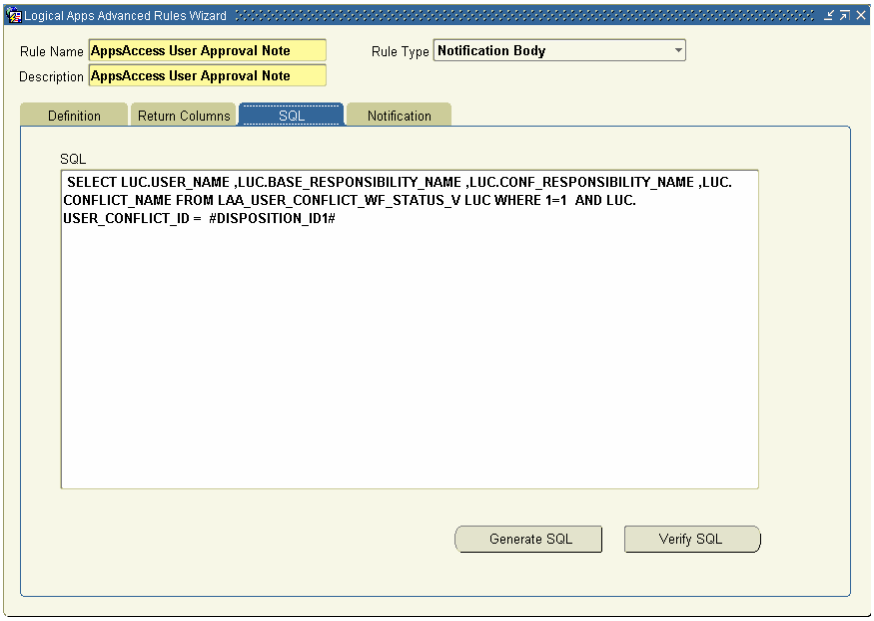
Specify one column in each row of the grid:

1. In the Seq field, type a number that indicates the sequence in which the column is to be called in a SQL statement to be generated from the selections you make.
2. In the Table Name list of values, select one of the tables you specified in the Definition panel.
3. In the Alias field, the Wizard supplies automatically the alias you created for the table in the Definition panel.
4. In the Column Name list of values, select a table column that contains information you want to use in the advanced rule.

Before continuing to the next panel, save the rule: Click on File in the menu bar and Save in the File menu.

Using the SQL Panel

The SQL panel is available no matter what rule type you select. Use it to generate a SQL SELECT statement automatically from the values you chose in the Definition and Return Columns panels, or to write or edit SQL statements directly:



1. Create a SQL SELECT statement:
   - If you have made selections in the Definition and Return Columns panels (or the Definition panel alone), click on the Generate SQL button. A SELECT statement appears.
   - If you have not made Definition or Return Columns selections, type a SQL SELECT statement directly in the SQL panel.
   - In either case, edit the SELECT statement as you see fit.
2. Click on the Verify SQL button. A pop-up message either informs you that the SQL statement is valid or explains any errors it contains.

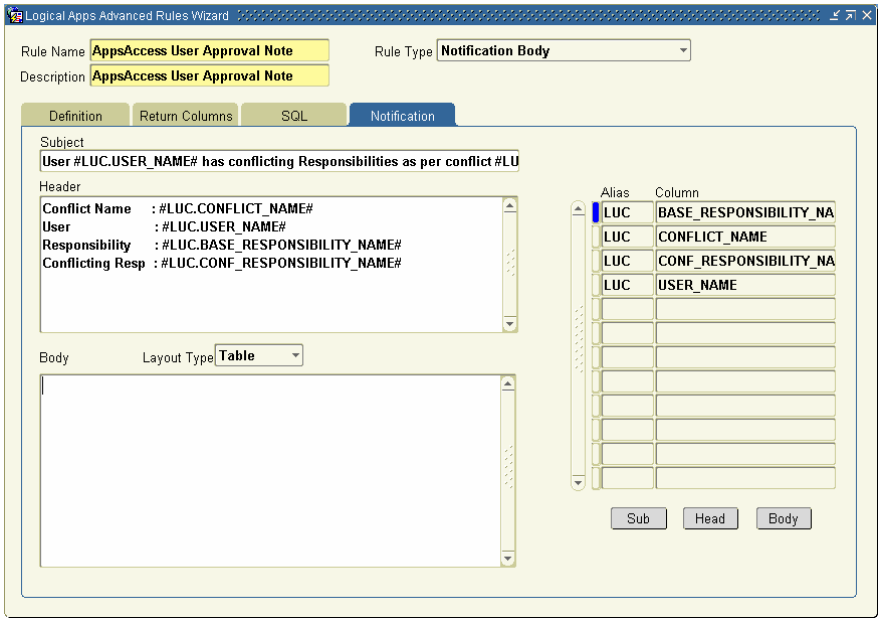
Note

The SQL statement incorporates the aliases you created in the Definition panel. If you generate and save a SQL statement, then change an alias in the Definition panel, the alias is not automatically updated in the SQL panel. You must update it manually.

Before continuing to the next panel, save the rule: Click on File in the menu bar and Save in the File menu.

Using the Notification Panel

The Notification Panel is available only if you select the Notification Body rule type. Use this panel to compose the text that is to appear in the subject line, header, and body of messages associated with notification and approval process flows. A message can incorporate not only text, but also the columns selected by the SQL statement generated in the SQL panel. At run time, these column names are replaced by values from those columns that are associated with the specific record about which a notification or approval is being sent.



To enter text, click in the Subject, Header, or Body text box and type whatever text you want.

To insert a column name into the text:

1. In the grid on the right, click on the name of the column you want to insert.
2. Click on the Sub, Head, or Body button to insert the correctly formatted column name in the Subject, Header, or Body text box, respectively.

When you insert a column name, it goes at the end of whatever text you have already entered in the text box. You can then type additional text either before or after the column name.

Click on the Layout Type list box to select Table (text is presented in a tabular format with visible borders), Form (text is presented in tabular format without visible borders), or no value (text is presented in free form).

Save the rule: Click on File in the menu bar and Save in the File menu.

Creating a Notification Function

As an alternative to creating a Notification Body element in the Advanced Rules Wizard, you can create a database function that provides text to a notification or approval process flow.

To do so, use the CREATE FUNCTION or CREATE OR REPLACE FUNCTION command available in SQL. Consult a SQL reference for detailed information about the use of these commands. However, the following conditions apply to a notification function intended for use with Flow Rules:

- The function can pass in only one argument, and that argument must be of the varchar2 data type.
- The notification function will be used with a flow rule that is based on a database table; the function's argument passes in primary-key values from this table. (Both table and primary keys are selected in the Primary Keys panel of the Flow Rules application; see page 4). The primary-key values are those appropriate for a database record which, by being inserted or updated in the table, has triggered the flow rule to run (see "Trigger Criteria" on page 8).

  Note that a notification function can therefore be used only with flow rules of the Trigger subscription type. Because a Periodic rule (one that runs on a schedule) is not associated with a specific table, primary-key values are not returned for it.
- If the table has two primary keys, the format of the argument is *key1*::*key2*. If the table has only one, the format is *key1* (the two delimiting colons are not retained).
- You may configure the function so that, while composing text, it uses primary-key values to select information specific to the record that has triggered the flow rule to run. If so, and if the argument contains two key values, you must parse them. (In the example on the next page, the two blocks after the BEGIN line use substr and instr to return portions of the argument, arg1, that precede the first delimiting colon and follow the second, and place them into variables.) If the argument contains only one key value, of if you configure the function to ignore the argument and provide static text, parsing is not necessary.
- The function may contain SELECT statements, but otherwise must not contain data manipulation language (DML) statements.

For example, the following code creates a function called ItemNumNotify. It is intended for use with a flow rule based on the mtl_system_items_b table, for which the primary keys are inventory_item_id and organization_id. It returns the statement "This notification applies to item number *value*," in which *value* is replaced by a number contained in a SEGMENT1 column of the mtl_system_items_b table.

```
CREATE FUNCTION ItemNumNotify (arg1 in varchar2)
RETURN varchar2 AS

  v_item varchar2(100);
  v_header varchar2(250);
  p_item_id number;
  p_org_id number;

BEGIN
  select substr(arg1,1,instr(arg1,':')-1)
  into p_item_id
  from dual;

  select substr(arg1,instr(arg1,':')+2)
  into p_org_id
  from dual;

  SELECT SEGMENT1
    INTO v_item
    FROM mtl_system_items_b
    WHERE inventory_item_id=TO_NUMBER(p_item_id)
    AND organization_id=TO_NUMBER(p_org_id);

  v_header := 'This notification applies to item number' || v_item;

  RETURN(v_header);

END;
```

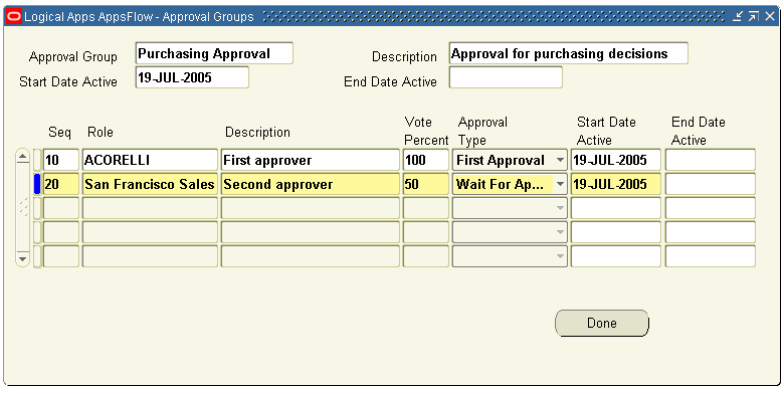
Creating an Approval Group

As one step in creating a Notification or Approvals process flow, an AppsFlow user would select an approval group — a set of people who may receive notification that an action has occurred, or who may be authorized to approve or reject such an action.

An approval group consists of workflow roles, which in turn comprise Oracle Applications users. So you may want to create workflow roles (see page 21) before approval groups. But AppsFlow recognizes each Oracle Applications user as a workflow role, so you can create approval groups without first creating workflow roles if you wish.

To create an approval group:

1 With AppsFlow running, click on AppsFlow in the menu bar, and then on AppsFlow Define Approval Groups in the AppsFlow menu. An Approval Groups form opens:



2. In the Approval Group field, type a unique name for the group.
3. In the Description field, type a brief explanation for the purpose of the approval group.
4. Set start and end dates for the group as a whole:

   The Start Date Active field defaults to the date on which you create the group. Accept the default or modify it to specify a future date on which you want the group to begin receiving notifications.

   The End Date Active field is blank by default. You may insert a date on which you want the group to expire, or you may leave the field blank to allow the group to remain active indefinitely. An end date earlier than a start date prompts an error message.

   In either case, you can select a date in the pop-up calendar that appears when you click on the list-of-values icon. Or type a date in the format configured for your instance of Oracle Applications.

Next, add as many workflow roles as you like to the approval group, one in each row of the grid:

1. In the Seq field, type a sequence number. The sequence effects an "approval hierarchy": each workflow role receives approval requests after they have been approved by roles with lower sequence numbers and before they are sent to roles with higher sequence numbers. Moreover, a workflow role does not receive a request if it is rejected by a role with a lower number.
2. In the Role list of values, select a workflow role (which may consist of more than one user).
3. In the Description field, type a brief explanation of what part the workflow role plays in the approval group.
4. In the Vote Percent field, type a number that expresses the percentage of workflow role members who must endorse a request before it moves to its next stage — consideration by the next workflow role in the approval group or actual approval. For example, if a workflow role comprises ten people and the Vote Percent field is set to 50, then five members must approve the request; otherwise it is rejected. Note the following:
   - Set the Vote Percent field to 100 if a workflow role consists of one person.
   - Change the Vote Percent field from its default value of 100 only if the Approval Type field is set to Wait for Approval (see step 5).
5. In the Approval Type field, select one of the following values:
   - First Approval: Only one member of a workflow role need approve a request for it to move to its next stage. (If you select this value, select 100 in the Vote Percent field.)
   - Wait For Approval: More than one person in a workflow role must approve a request for it to move to its next stage. (The number of required approvers is determined by the value in the Vote Percent field.)

- No Approval Required: Members of the workflow role receive notification that an action has occurred, but do not have the opportunity to approve or reject it. (This is the appropriate selection for all workflow roles in a group created for "Notification" process flows.)

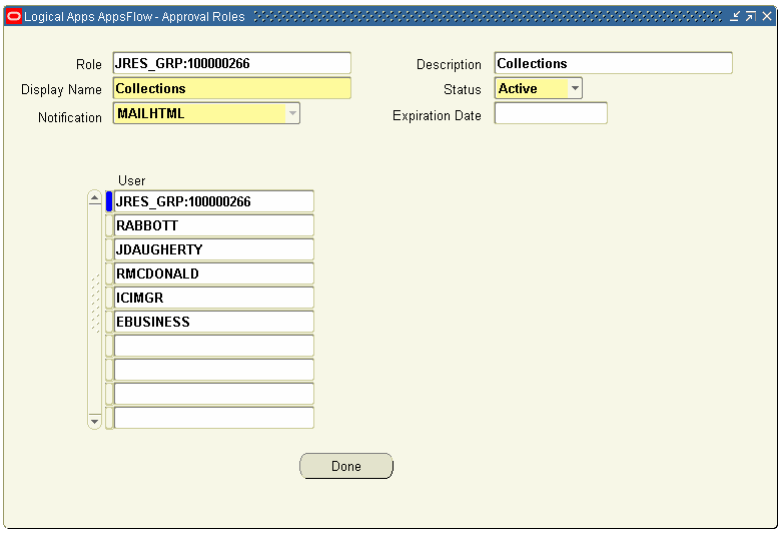
6 In the Start Date Active and End Date Active fields, set start and end dates for the participation of the workflow role in the approval group. (Use the same conventions as you did when specifying start and end dates for the group as a whole.)

When you finish configuring an approval group, save it: click on File in the menu bar and Save in the File menu. When you finish working in the Approval Group form, click the Done button to close it.

Creating a Workflow Role

A workflow role is a set of one or more Oracle Applications users, and serves as a building block for approval groups. You can also select a workflow role as the functional owner of a process flow (see page 24). To create a workflow role:

1 With AppsFlow running, click on AppsFlow in the menu bar, and then on AppsFlow Define Roles in the AppsFlow menu. An Approval Roles form opens:



2 In the Role field, type a unique name for the role.

3 In the Description field, type a brief explanation for the purpose of the role.

4 In the Display Name field, create an easily recognizable name for the role. AppsFlow uses this name to identify the role to users as they add it to approval groups or select it as the functional owner of process flows.

5 In the Status list box, select Active to place the role into use or Inactive to remove it from use. Note that once you inactivate a role, you cannot reactivate it.

6 In the Notification list of values, select the format in which members of the role will receive notifications.

7 In the Expiration Date field, enter a date on which the role expires, or leave the field blank to allow the role to exist indefinitely. Select a date in the pop-up calendar that appears when you click on the list-of-values icon. Or type a date in the format configured for your instance of Oracle Applications.

8 In the User list, select users, responsibilities, or other roles to be included in this role

When you finish configuring a workflow role, save it: click on File in the menu bar and Save in the File menu. When you finish working in the Approval Roles form, click the Done button to close it.

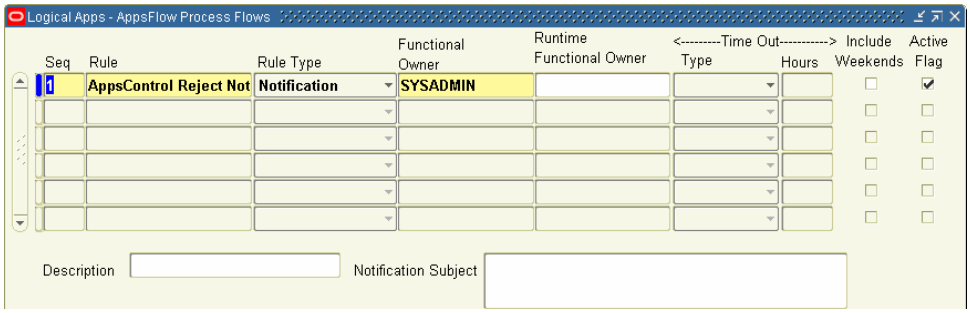
Chapter 5

Creating Process Flows

To define the individual process-flow rules that combine to make up a process rule, click the Process Flows button in the main AppsFlow form. A Process Flows form opens.

Starting a Process Flow

Each process flow is assigned a "rule type" that determines what it does. The content of the Process Flows form varies according to the rule type. No matter what rule type you select, however, the upper portion of the Process Flows form contains this grid:



Enter values in one row of this grid for each process flow you want to create:

1 In the Seq field, enter a number that reflects the order in which you want this flow rule to be processed with respect to other flow rules listed in the grid. Two or more rows may share a sequence number; if so, the rules they define are processed in parallel. (Enter a number even if only one row contains content.)

2. In the Rule field, type a unique name for the flow rule.
3. In the Rule Type field, select a rule type from among the following options:
   - Check Constraint: This rule determines whether appropriate data has been provided before allowing a process to continue. It would incorporate a Constraint/Condition element created in the Advanced Rules Wizard.
   - Concurrent Programs: This rule runs one or more concurrent programs. It may incorporate Runtime Responsibility or Runtime User elements created in the Advanced Rules Wizard.
   - Approvals: This rule sends approval requests to users authorized to approve or reject them. It may incorporate any of several elements created in the Advanced Rules Wizard (chief among them a Notification Body) as well as an approval group created in the Approval Groups form.
   - Exceptions: This rule alerts designated persons to errors or other exceptional conditions. It would incorporate an Exception element created in the Advanced Rules Wizard.
   - SQL: This rule runs SQL scripts, and may incorporate a Sql Script element created in the Advanced Rules Wizard.
   - Notification: This rule notifies designated persons that actions have been completed. It may incorporate any of several elements created in the Advanced Rules Wizard (chief among them a Notification Body) as well as an approval group created in the Approval Groups form.
   - Processes: This rule specifies another AppsFlow process to be run from within the AppsFlow process you are configuring.
   - Workflow/Event: This rule calls Oracle workflow components.
4. In the Functional Owner list of values, select a workflow role that "owns" this process flow. In other fields, you may designate users responsible for receiving communications generated by the process flow; in an Approval rule, for example, you may specify an approval group. If no such users are designated, however, the functional owner assumes the responsibility.
5. In the Runtime Functional Owner list of values, you may choose a SQL statement that selects a functional owner at runtime. The LOV displays Runtime Functional Owner Sql elements configured in the Advanced Rules Wizard. A selection here supersedes the selection in the Functional Owner field; if you leave the Runtime Functional Owner field blank, the Functional Owner selection takes effect.
6. The Time Out fields become active only if you have selected the Approvals rule type. In them, you can designate an amount of time in which designated approvers are to respond to approval requests:
   - To set a fixed amount of time, select Fixed in the Type list box and type a number of hours in the Hours field. Select the Include Weekends check box if you wish the time period to include Saturdays and Sundays.
   - To set an amount of time that may vary according to circumstances, select Runtime in the Type list box. You would then need to specify a Runtime

Timeout Sql element configured in the Advanced Rules Wizard. (You would do so elsewhere than in this grid; see page 29.) Leave the Hours field blank. Once again, select the Include Weekends check box if you wish the time period to include Saturdays and Sundays.

- To allow an unlimited time for approval responses, leave both the Type and Hours fields blank, and clear the Include Weekends check box.

7 Select the Active Flag check box to activate the process-flow rule or clear the check box to deactivate the rule.

8 In the Description field, type a brief explanation of the process flow you are configuring

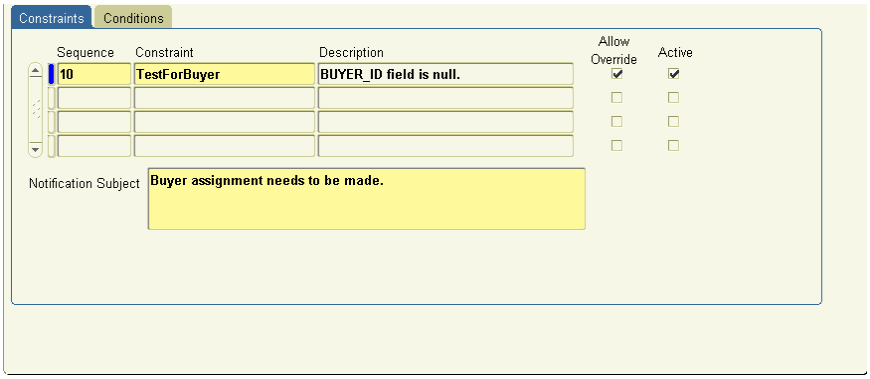
9 In the Notification Subject box, type text that may appear in the subject line of a notification sent to designated reviewers when this process-flow rule fires. This text is used if no other subject-line text is specified. For example, a Notification rule may call a Notification Body element in which subject text has been configured; in that case the subject text from the Notification Body element would be used, and the subject text from this Notification Subject field would be ignored.

Configuring Constraint Rules

A Check Constraint rule determines whether appropriate data has been provided before allowing a process to continue; if not, it notifies a user who can obtain and enter that data.

For example, a company's database may contain a table that stores records of items that it purchases. The company may require that a buyer be assigned for each new item added to the table, and each record may contain a field that holds the ID of the assigned buyer. In the Advanced Rules Wizard, a Constraint/Condition element may have been created to test whether that buyer ID field is null. Now, a Check Constraint process flow would call that element; whenever it evaluates to true, the process flow would send a notification to the functional owner of the flow. In response, the owner would be expected assign a buyer and enter the buyer's ID in the appropriate field. The Check Constraint flow then verifies that the ID is correctly entered, resending the notification if not.

If you select the Check Constraint rule type, the Process Flows form displays a panel labeled Constraints:



To complete these fields:

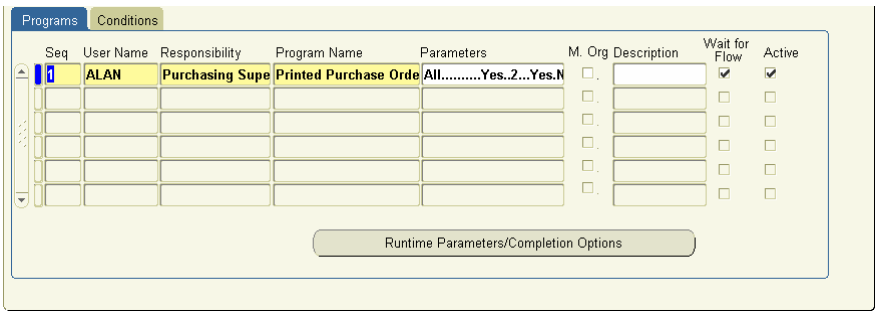
1. In the Sequence field, enter a number reflecting the order in which you want this constraint to be processed with respect to others listed in the grid. Two or more rows may share the same sequence number; if so, the constraints they define are processed in parallel. (Enter a number even if only one row contains content.)
2. In the Constraint list of values, select one of the constraints previously configured in the Advanced Rules Wizard. In the Description field, AppsFlow automatically provides the description configured for the constraint in the Advanced Rules Wizard.
3. Select the Allow Override check box if you want to enable the functional owner to override the constraint — allow the process to continue without supplying the requested data. Clear the Allow Override check box if you want to require the functional owner to supply the requested data.
4. Select the Active check box to place the constraint in use, or clear the check box to remove the constraint from use.
5. In the Notification Subject box, type text that appears in the notification sent to the functional owner. Note that the header of the notification contains identifying values and labels selected in the Display Table/View Columns panel; see page 5. Text entered in the Notification Subject box in the top half of the Process Flows form (see step 9 on page 25) is ignored.

You may create any number of constraints, one per row in the grid. Each time you select a new row, the Notification Subject box clears so that you can enter a new subject wording for each constraint. When you finish creating the rule, save it: Click on File in the menu bar and Save in the file menu.

Configuring Concurrent Program Rules

A Concurrent Program rule runs one or more concurrent programs. For each program, you can configure the rule either to accept static parameters or to execute SQL statements that determine parameters at runtime, and you can have it notify a user (or workflow role) when each program has finished running. For programs that produce output files (such as reports), you can also specify a printer to which output files should be sent and a number of copies to be printed.

When you select the Concurrent Program rule type, the Process Flows form displays a panel labeled Programs:



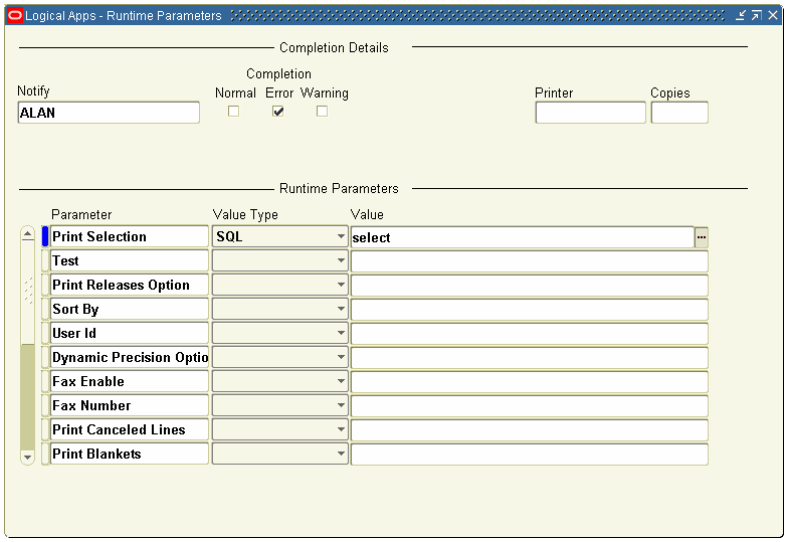
To complete these fields:

1. In the Seq field, enter a number that reflects the order in which you want this concurrent program to be run with respect to other programs listed in the grid. Two or more rows may share a sequence number; if so, the programs they call are processed in parallel. (Enter a number even if only one row contains content.)
2. In the next three fields, select the following values in any order:
   - In the User Name list of values, select a user associated with the running of the concurrent program. The list of values displays both workflow roles and Runtime User elements created in the Advanced Rules Wizard.
   - In the Responsibility list of values, select a responsibility from which the concurrent program can be run. The list of values displays both responsibility names and Runtime Responsibility elements created in the Advanced Rules Wizard.
   - In the Program Name list of values, select the concurrent program you want to run.

   You must select a responsibility from which the program can be run and a user who has been assigned that responsibility. As you make a selection in any one of the fields, selections in the remaining fields are narrowed accordingly.
3. When you select a program, one or more pop-up windows prompt you to select parameters for the program. When you finish making selections in these windows, the parameters you've selected appear in the Parameters field.
4. If you select the M.Org check box, a Multiorg Program Setup tab appears. Click on the tab to reveal a panel in which you can select additional users to run the program, as well as a responsibility and individual set of program parameters for each. (Typically, you would leave the M.Org check box cleared.)
5. In the Description box, type a brief explanation for why the program is being run.
6. Select the Wait for Flow check box to pause the process defined by this process rule until the concurrent program has finished running. Clear the Wait for Flow check box to enable the process to continue running while the concurrent program is also running.
7. Select the Active check box to activate the concurrent program configuration, or clear the check box to suspend it.

You may enter values in any number of rows in order to run any number of concurrent programs. For each, you can click on a Runtime Parameters/Completion Options button to substitute a set of parameters to be determined at runtime for the static parameters you've already chosen, or to select notification or printing options that take effect once the program finishes running.

When you click the Runtime Parameters/Completion Options button, the following form appears:



In the Completion Details section of the form, you may do the following:

- In the Notify list of values, choose a user to be notified when the program finishes running. Select any combination of Completion check boxes to set the circumstances under which the user is notified: when the program finishes normally, with an error, or with a warning.
- In the Printer list of values, select a printer to which the concurrent program would print its output file (if it produces one). In the Copies field, type the number of copies to be printed.

The Runtime Parameters section of the form should present, in the Parameter column, a list of the parameters that apply to the program you are running. For each, you may do either of the following:

- Select Constant in the Value Type field. Then, in the Value field, type a constant value.
- Select SQL in the Value Type field. Then, in the Value field, provide a SQL statement that sets a parameter value. You may type a SQL statement directly in the field, or you may select a Sql Statement element or Single Return Function element that has been created in the Advanced Rules Wizard.

In either case, any parameter values you select here supersede those you selected in the Process Flows form (see step 3 on page 27).

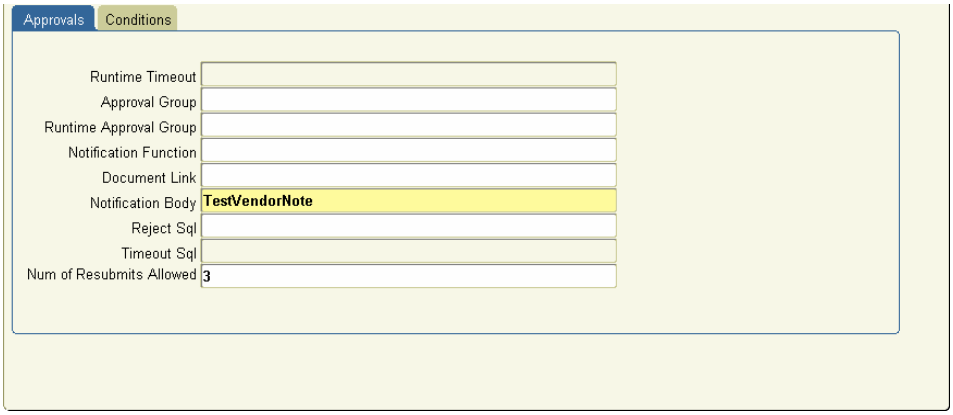
When you finish creating the rule, save it: Click on File in the menu bar and Save in the file menu.

Configuring Approval and Notification Rules

A Notification rule sends an information-only message to designated users when a specified action has been completed, and an Approval rule sends a message requiring

that designated users respond with an approval or rejection of the newly completed action. Configuration of the two rule types is very similar.

When you select either rule type, the Process Flows form displays a panel labeled Approvals:



In this panel, you may do the following. (Note that, as always, yellow fields require input, white fields are optional, and gray fields do not accept input.)

- Designate users who are to receive notifications generated by the rule:
  - In the Approval Group list of values, you may select among approval groups configured in the Approval Groups form (see page 19).
  - In the Runtime Approval Group list of values, you may select among Runtime Approval Group Sql elements configured in the Advanced Rules Wizard.

  Typically you would choose one or the other. If you choose one of each, the Runtime Approval Group selection supersedes the Approval Group selection. If you choose neither, the functional owner of the rule receives its notifications.
- Select the content of the notification message. Do one of the following:
  - In the Notification Body list of values, select among Notification Body elements configured in the Advanced Rules Wizard. If you do, the subject line configured for that message is used, and any text entered in the Notification Subject box in the top half of the Process Flows form is ignored.
  - In the Notification Function field, designate a database function that generates notification content. For information on configuring such a function, see "Creating a Notification Function" on page 18.
- In the Document Link field, create a link to a form in which a message recipient would enter data in response to the message. Enter the name of the appropriate form function in this field.
- Make timeout selections (which apply only if you are configuring an Approvals rule, not a Notification rule).
  - If you selected Runtime in the Time Out Type list box, you have chosen to set an amount of time in which users must respond to a request generated by an Approvals rule, and you have chosen to designate a SQL statement that

defines the period at runtime. In the Runtime Timeout list of values, select that statement by choosing among Runtime Timeout Sql elements configured in the Advanced Rules Wizard. (If you chose to create a fixed timeout or no timeout, this field does not accept input.)

– In the Timeout Sql list of values, select a SQL statement that determines what happens if a timeout period (fixed or runtime) expires. The LOV displays TimeOut Sql elements created in the Advanced Rules Wizard. (If you chose not to establish a timeout period, this field does not accept input.)

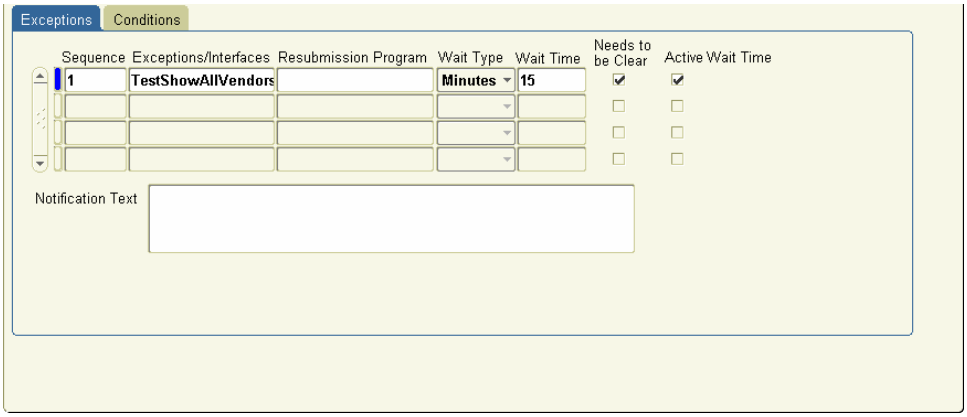
- In the Reject Sql list of values, select a Reject Sql element configured in the Advanced Rules Wizard. This element executes a SQL statement if a designated reviewer rejects an approval request in an Approvals process flow.
- In the Number of Resubmits Allowed field, type a number that expresses how many times an approval request may be reconsidered after being rejected. If rejections exceed this number, the process with which the request is associated comes to an end.

When you finish creating the rule, save it: Click on File in the menu bar and Save in the file menu.

Configuring Exception Rules

An Exception rule tests whether an error or some other exceptional condition exists. As you configure the rule, you can specify not only the exception itself, but also a "re-submission program" that may be run to resolve the exception. You can also write text of a message to notify users that the exception exists, determine whether it must be resolved, and specify an interval at which AppsFlow tests to determine whether the exception has been resolved.

When you select the Exception rule type, the Process Flows form displays a panel labeled Exceptions:



To complete these fields:

1 In the Sequence field, enter a number reflecting the order in which you want this exception to be tested with respect to other exceptions listed in the grid. Two or

more rows may share a sequence number; if so, the exceptions they call are processed in parallel. (Enter a number even if only one row contains content.)

2. In the Exceptions/Interfaces list of values, select the exception for which you want to test. The LOV displays Exception elements created in the Advanced Rules Wizard.
3. In the Resubmission Program list of values, select a SQL statement that provides data to resolve the exception. The LOV displays Sql Statement elements defined in the Advanced Rules Wizard.

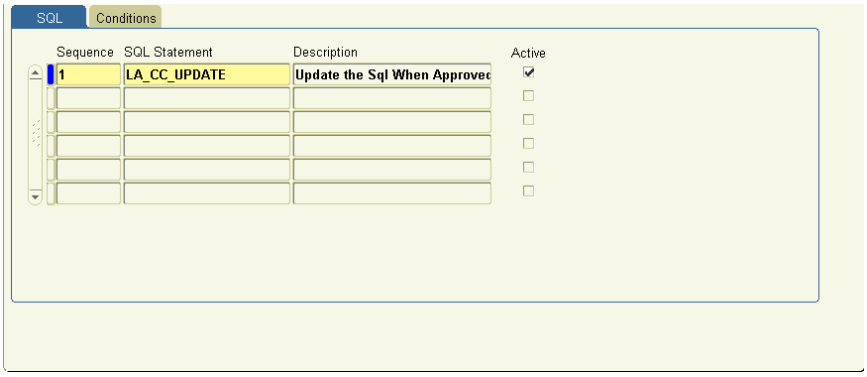
   The notification message generated by an Exception rule automatically contains a Reprocess button, and the resubmission program is run when the recipient of the notification message clicks on its Reprocess button.
4. Define an interval at which the system reruns the Exception element (selected in step 2):
   - In the Wait Type list box, designate the unit of time you use to define the interval: Minutes, Hours, or Days. (Or select the blank entry to prevent retesting for the exception.)
   - In the Wait Time field, type a number that expresses the interval. For example, if you type 15 here and select Minutes in the Wait Time field, the exception is evaluated every 15 minutes.
   - Select the Active Wait Time check box to implement the retesting interval (or clear the check box to prevent retesting).
5. Select the Need to be Clear check box to require that the exception be resolved in order for the process of which it is a part to continue. Or, clear the check box to allow the process to continue even if the exception is not resolved.
6. In the Notification Text box, type text to be displayed in a message notifying users that the exception exists. The recipient of this message is the functional owner of the Exception rule. The message also includes a subject line created in the Notification Subject box in the top half of the Process Flows form (see step 9 on page 25).

You may test for any number of exceptions, one per row in the grid. Each time you select a new row, the Notification Text box clears so that you can enter a new text wording for each exception. When you finish creating the rule, save it: Click on File in the menu bar and Save in the file menu.

Configuring SQL Rules

A SQL rule executes one or more Sql Statement elements created in the Advanced Rules Wizard. Each of these elements contains free-form SQL, so you can use SQL rules to execute any database procedure or SQL statement that must be run as part of a process.

When you select the SQL rule type, the Process Flows form displays a panel labeled SQL:



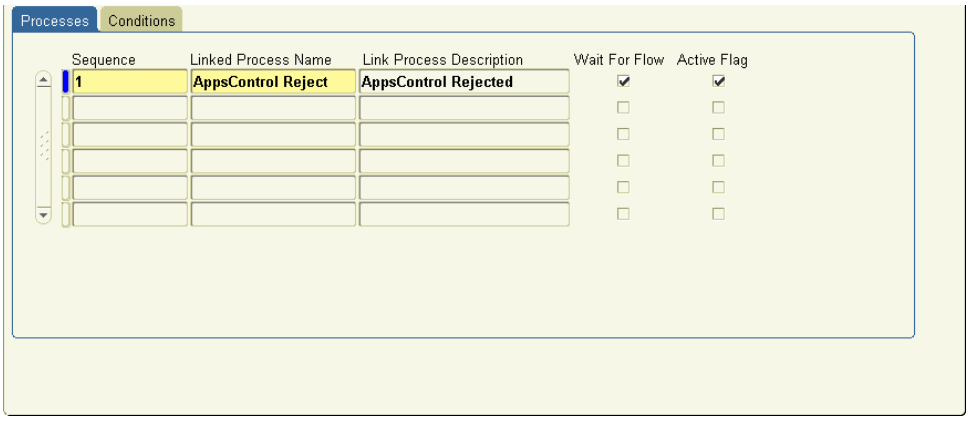
To complete these fields:

1. In the Sequence field, enter a number that reflects the order in which you want this SQL statement to be run with respect to other statements listed in the grid. Two or more rows may share a sequence number; if so, their SQL statements are processed in parallel. (Enter a number even if only one row contains content.)
2. In the SQL Statement list of values, select the SQL statement you want to run. The LOV displays Sql Statement elements created in the Advanced Rules Wizard.
3. The Description field displays a description associated with the Sql Statement element when it was configured. You cannot enter a value directly in this field.
4. Click the Active check box to cause the SQL statement to be run; clear the check box include the statement in the rule without actually running it.

You may include any number of SQL statements in the rule, one per row in the grid. When you finish creating the rule, save it: Click on File in the menu bar and Save in the file menu.

Configuring Processes Flows

A Processes flow runs one or more process rules (that is to say, one or more full AppsFlow processes) from within the current process rule. When you select the Processes rule type, the Process Flows form displays a panel labeled Processes:



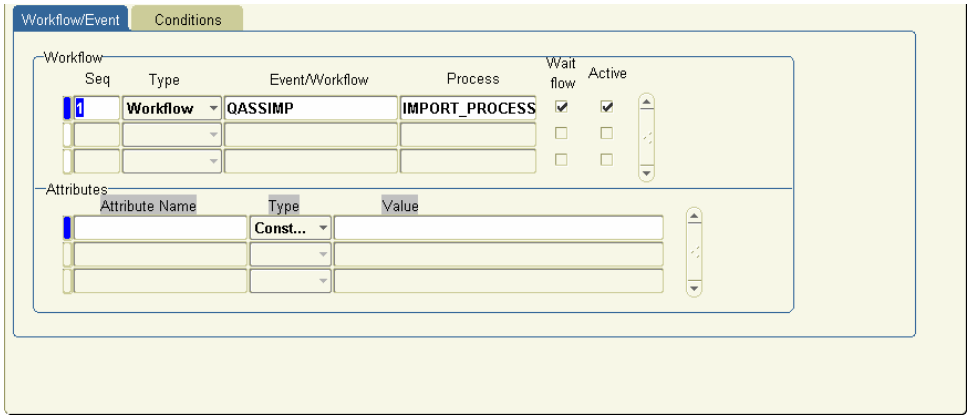
To complete these fields:

1. In the Sequence field, enter a number that reflects the order in which you want this process to be run with respect to other processes listed in the grid. Two or more rows may share a sequence number; if so, the processes they call are run in parallel. (Enter a number even if only one row contains content.)
2. In the Linked Process Name list of values, select a process rule. The LOV displays process rules created in the main AppsFlow form.
3. The Description field displays a description associated with the process rule (if any) when it was configured. You cannot enter a value directly in this field.
4. Select the Wait For Flow check box to cause the current process to be paused until the called process finishes running. Clear the Wait For Flow check box to allow the two processes to run simultaneously.
5. Click on the Active Flag check box to activate the row, or clear the flag to deactivate the row.

You may include any number of process rules in this flow, one per row in the grid. When you finish creating the flow, save it: Click on File in the menu bar and Save in the file menu.

Configuring Workflow/Event Rules

A Workflow/Event rule calls workflow or business event components, as configured for the Oracle workflow system, to be run from within the AppsFlow process. (See Oracle documentation for information on creating such components.) When you select the Workflow/Event rule type, the Process Flows form displays a panel labeled Workflow/Event:



To complete these fields, begin in the Workflow block:

1. In the Seq field, enter a number that reflects the order in which you want this component to be run with respect to other components listed in the grid.
2. In the Type list box, select Workflow or Business Event.
3. In the Event/Workflow list of values, select a workflow or a business event, depending on the selection you made in step 2.

4 In the Process list of values, select a process (in the Oracle workflow sense of that word) associated with the workflow you selected in step 3. If you selected a business event in step 3, a process does not apply and the Process field does not accept input.

5 Select the Wait Flow check box to cause the AppsFlow process rule to pause until the workflow or business event finishes running. Or clear the check box to allow the AppsFlow process rule and the workflow or business event to run simultaneously.

6 Select the Active check box to activate the row, or clear the box to deactivate the row.

You may call any number of workflow or business event components, one per row in the grid. For each row, you may also set the input-data values provided to the workflow or business event as attributes. To do so, work in the Attributes block of the form:

1 Select attributes, one in each row, in the Attribute Name column.

2 In the Type column, select the format in which you will provide a value for the attribute: Constant, Sql, Current Date, or Profile.

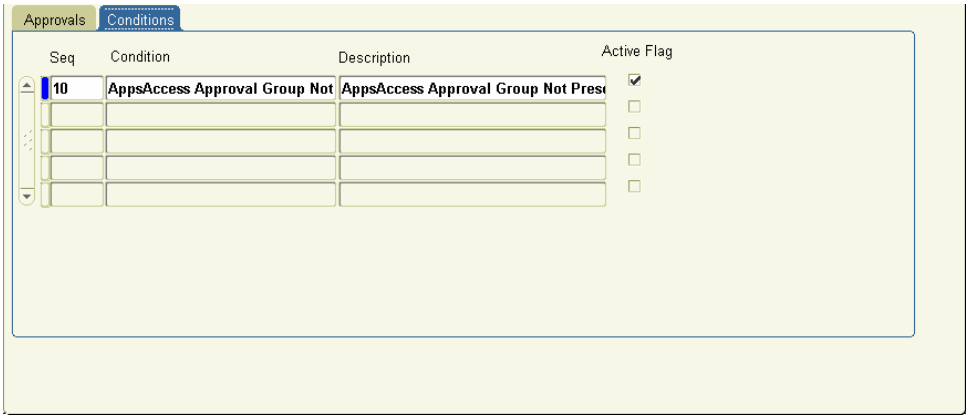
3 In the Value column, enter a value appropriate for the type you have selected.

Adding Conditions to Process Flows

As noted earlier, a Constraint/Condition element (which is created in the Advanced Rules Wizard) specifies values that may be held in one or more columns of a table, and it evaluates to true for records that either do or do not contain the specified values. As a result, it can be added to any process flow as a way of selecting records — the process flow would apply only to those records for which the Constraint/Condition element evaluates to true.

No matter what rule type you select as you create process flows, AppsFlow provides a Conditions panel in which you can add conditions to the process flow:

1 In the bottom half of the Process Flows form, click on the Conditions tab. A conditions panel appears:



2. In the Seq field, enter a number that reflects the order in which you want this condition to be evaluated with respect to other conditions listed in the grid.
3. In the Condition list of values, select a condition you want to use to select records that are subject to the process flow you are creating. The LOV displays Constraint/Condition elements created in the Advanced Rules Wizard.
4. The Description field displays a description associated with the Constraint/Condition element when it was configured. You cannot enter a value directly in this field.
5. Select the Active Flag check box to activate the condition, or clear the check box to hold the condition in reserve.

You may list any number of conditions, one per row in the grid.

Chapter 6

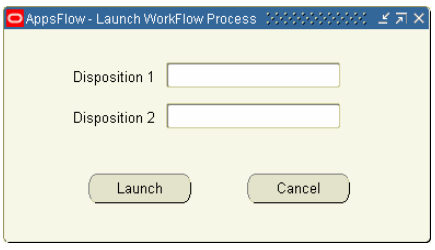
Testing Process Rules

Once you have created and saved a process rule, you can launch it in order to test whether it works as you intend.

When launched, the process performs the actions it is configured to perform, such as sending notifications or running concurrent programs. If the Process Status field of the Process Details panel is set to Development (see page 4), notifications go only to the user identified as the parent process owner. But if the Process Status field is set to Production, notifications go to the users actually configured to receive them.

To launch a process for testing:

1. Select (click on) the process rule you want to test in the Process Details panel of the main AppsFlow form.
2. Click on AppsFlow in the menu bar, then on AppsFlow Launch Flow in the AppsFlow menu.
3. If you selected a rule with the Periodic or Business Event subscription type, the process is launched; skip to step 4. If you selected a rule with a Trigger subscription type, the following dialog box appears:

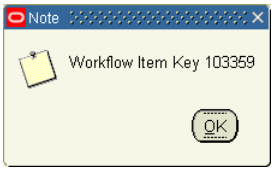


In the Disposition 1 and Disposition 2 fields, enter actual values for the disposition fields you identified as you configured the process rule. These fields may be the primary keys for the table on which the process is based (selected in the Primary Keys panel; see page 4), the relation keys specified in the Launch Criteria form (page 8), or the disposition IDs selected for an advanced rule element that is to be called by the process (page 13).

You can use any record from the table containing these fields, but be sure to choose an appropriate one. For example, if the process includes a Check Constraint flow rule that tests for missing data, you might want to select a record from which that data item is in fact missing.

When you have entered disposition values, click the Launch button.

4 A window informs you of a workflow item key associated with the launching of the process. Make a note of the number it displays, and click the OK button to close the window.



Once the process has been launched, you can look in email, Oracle work list, or other media for notifications you would expect the process to have sent, use the View Requests feature to determine whether concurrent programs have been run, and so on.

If expected events do not take place, you can use the Status Monitor in the Oracle Workflow system to review how the process ran and check for errors. Search for the workflow item key generated when you launched the process. (As you do so, select LAAF Header as the workflow type.)

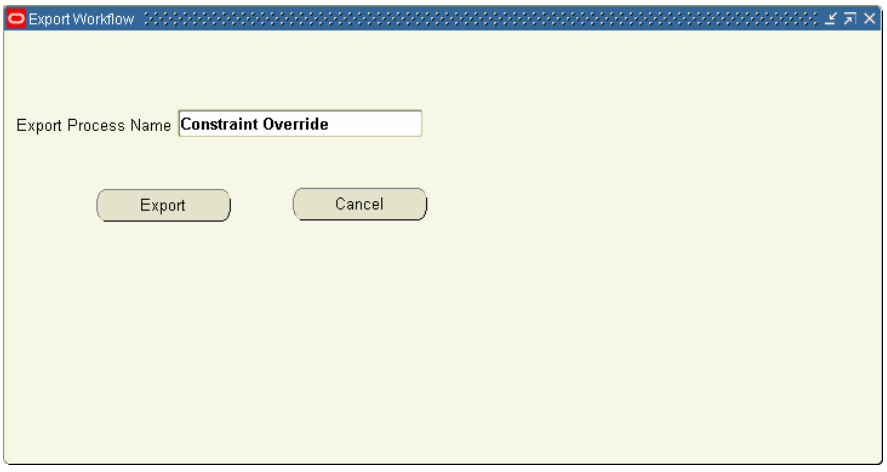
Chapter 7

Integration with Oracle Workflow

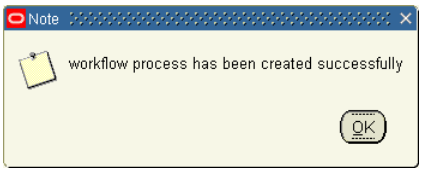
A process created in AppsFlow may serve as a component of an Oracle Applications Workflow. For that to happen, you must export the process to the Oracle Workflow system. A process need be exported only once; any changes made subsequently to the process in AppsFlow are reflected in its implementation within Oracle Workflow.

To export an AppsFlow process to Oracle Workflow:

1. In the Process Details panel of the main AppsFlow form, select (click on) the process rule you want to export.
2. Click on AppsFlow in the menu bar, then on AppsFlow Export to Workflow in the AppsFlow menu. The following form appears:



3 Confirm that the Export Process Name field displays the process you selected in step 1. Then click on the Export button. The following confirmation window appears:



4 Click on the OK button to clear the confirmation message.

Once exported, the process exists in the Oracle workflow system within an item called Logical Apps Standard Functions.

Having exported the process, you would need to open the Oracle Workflow Builder, drag the process from Logical Apps Standard Functions into the Oracle workflow to which you want to link it, and set keys as required by the Oracle workflow system. (See Oracle documentation for information on using the Workflow Builder and working with the Oracle Workflow system.)

Chapter 8

AppsFlow Migration

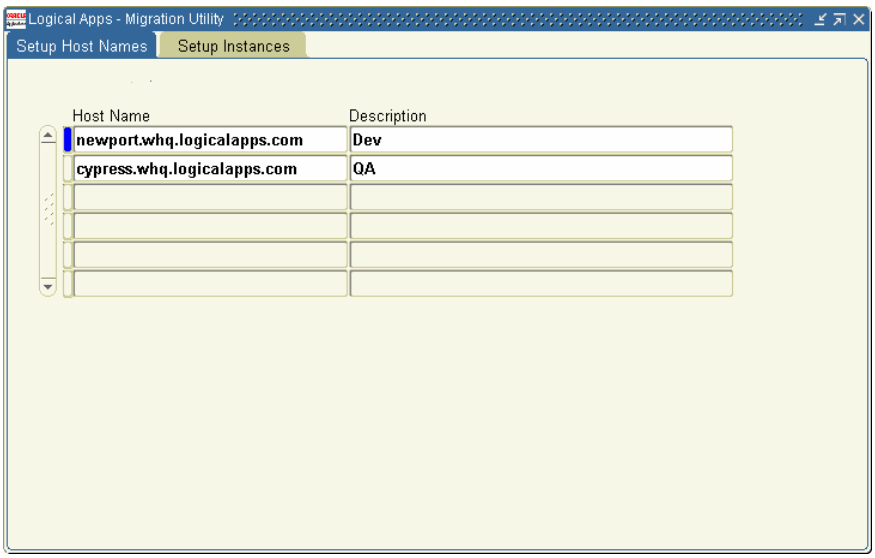
Once you have created process rules for an instance of Oracle Applications, you can "migrate" them — copy a process rule, or a process flow within a process rule, directly to another Oracle Applications instance. You can also export individual process rules or flows to, or import them from, XML files, or copy them under new or modified names on the source instance.

Preparing for Migration

Before you can migrate process rules or flows, you need to specify connection information in all the environments to and from which you plan to transmit data. You need to know the host name, instance SID, and database instance port for each environment. This information is found in the TNSNAMES.ora file, which is located in *ORACLE_HOME*/network/admin.

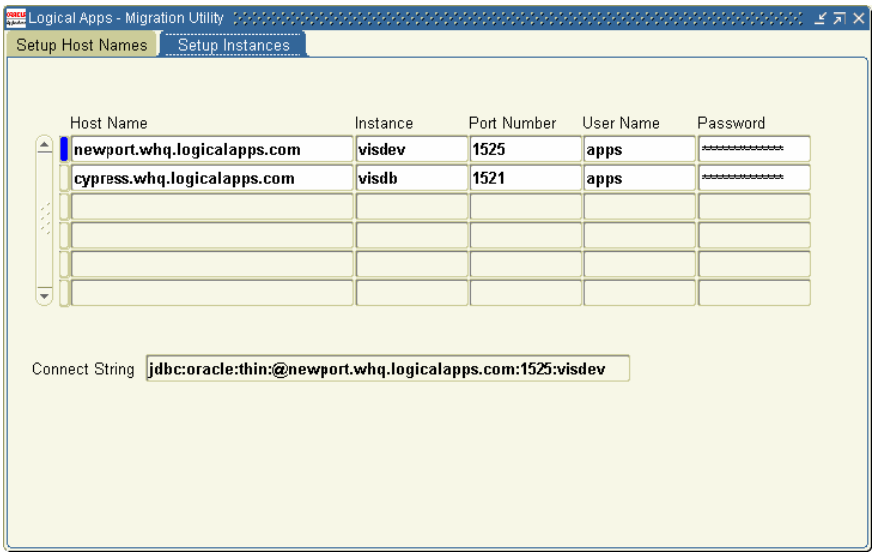
Once you've gathered this information, use the Logical Apps Migration Utility to perform the connectivity configuration:

1. With AppsFlow open (see page 2), click on LogicalApps Utilities in the menu bar, then on Migration Setup in the Utilities menu. A Migration Utility form appears (as shown at the top of the next page).
2. Ensure that the Setup Host Names tab is selected.
3. In the Host Name column, enter the host name (machine name) for each of the machines hosting the database and involved in the migration.



4 In the Description column, you may enter a description for each host name. (This step is optional.)

5 Click on the Setup Instances tab. The following form appears:



6 In the Host Name column, select the host name for each of the machines from the list of values. (The entries are those defined in the Setup Host Names tab.)

7 In the Instance and Port Name columns, type the instance name and port number that corresponds to each host name.

8 Under User Name, type the value *apps* for each entry. Under Password, enter the password for the apps user.

9 Click on File in the menu bar and Save in the File menu. When the configuration is saved, the system automatically generates and displays a connection string.

10 Close the Migration Utility. Click on the × symbol in the upper right corner of the form.

Stipulations

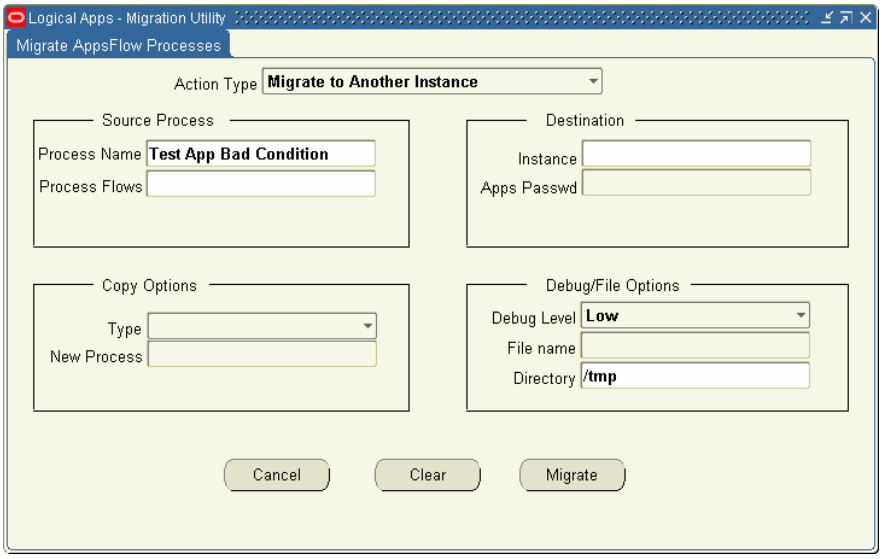
The following conditions apply to migration, export, and import operations:

- If you migrate an entire process rule, its process flows and the elements called by those flows — advanced rules, approval groups, and workflow roles — move with the rule to the target instance or XML file. If you migrate a process flow, the elements called by it do not move to the target instance or XML file.
- For a process flow to be migrated, the process rule that contains it must already exist on the destination instance.
- A log file gathers information about a migration, export, or import operation. If an operation fails and you are unable to determine why, rerun the operation with the debug level changed from low to high and evaluate the log data.

Migrating, Exporting, or Copying Rules

To migrate a process rule or process flow rule to another instance, or to copy one either to an XML file or on the source instance, complete the following steps:

1. Open AppsFlow (see page 2) and select the process rule you want to use as the source for a migration, export, or copy operation.
2. Click on LogicalApps Utilities in the menu bar, then on Migrate Rules in the Utilities menu. The Migrate AppsFlow Processes form appears:



3. In the Action Type list box, select the operation you want to perform:
   - Migrate to Another Instance (the default) if you want to migrate a process or flow.
   - Export to File if you want to export a process or flow.
   - Copy within the Same Instance if you want to copy a process or flow under a new or modified name on the source instance.

4 In the Source Process block, identify an item to be migrated, exported, or copied:

- The Process Name list of values displays the process rule you selected in step 1. Retain the entry to work with that process (or flows within that process); or, select another.
- In the Process Flows list of values, do one of the following:

  Leave the field blank to migrate, export, or copy the full process rule you selected in the Process Name field. This includes the process, all its process flows, and all the elements called by those flows.

  Select All Process Flows to migrate, export, or copy all the flows that belong to the process rule, but not the process rule itself or the elements called by the process flows.

  Select a single flow belonging to the process to migrate, export, or copy that flow, but not the process rule itself or elements called by the process flow.

5 If you are performing a migration, make entries in the Destination Instance block:

- In the Instance list of values, select a destination instance for the migration.
- In the Apps Passwd text box, type the apps password for the destination instance if you are prompted to do so. (This prompt appears if the XXLAAPPS: Enable for Migration Security profile option is set to Yes on the source instance. If the option is set to No, the prompt does not appear and a password need not be entered.)

If you are performing a file export or copying to the source instance, fields in the Destination block do not apply and do not accept input.

6 If you are copying to the source instance, make entries in the Copy Options block:

- If you are copying a process flow or all process flows (if you entered any value in the Process Flows field of the Source Process block), the Type field defaults to Add to Existing Process no matter what value you select there. In the second field, select the existing process into which you want to copy the flows.
- If you are copying an entire process (if you left the Process Flows field blank), select one of three values in the Type field: Copy as a New Process if you want to assign a completely new name to the copy, or Add a Prefix or Add a Suffix if you want to assign the copy a name that consists of the original process name with text added at the beginning or end. In the second field, type the text you want to use as a new process name or as a prefix or suffix to the original name.

If you are performing a migration or a file export, fields in the Copy Options block do not apply and do not accept input.

7 Make selections in the Debug/File Options block:

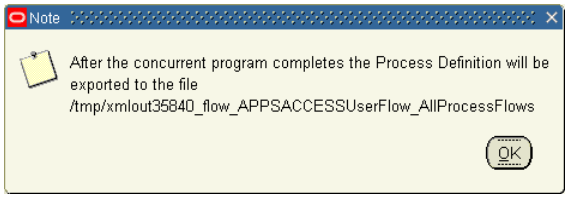
- In the Debug Level list box, select a level of detail for error reporting to a log. Ordinarily, select Low; select High instead if you need to uncover the cause of a failed migration, export, or copy.

- In the Directory text box, type the path that designates a temporary staging file location for XML files to be generated and, in the case of migration, copied to the destination instance.
- The File Name field does not accept input.

8 Click on a button that launches the process. Its label varies depending on your selection in step 3: Migrate if you chose Migrate to Another Instance, Export if you chose Export to File, or Copy if you chose Copy within the Same Instance.

9 Review several messages:

- The system launches a concurrent program to implement the migration, export, or copy. The first message provides an ID number. Click on the OK button to clear the message.
- If you have performed a file export, a message similar to the following one displays the name of the export file you have generated.



In the file name, the term *xmlout* designates XML output, a number (35840 in this example) uniquely identifies the export operation, the term *flow* identifies the Logical Apps component involved in the export, and final phrases (in this example, *APPSACCESSUserFlow* and *AllProcessFlows*) identify the export items. Make a note of the file name and location, and then click the OK button.

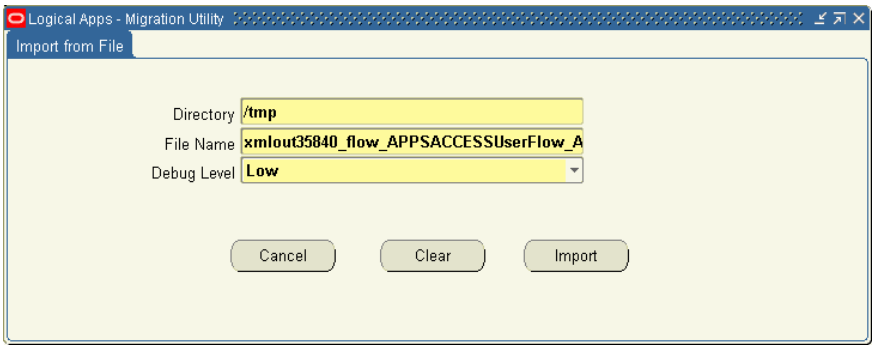
- Finally, a dialog prompts you to perform another migration. Click Yes to do so or No to close the Migration form.

Importing a Process or a Flow

To import an XML file containing a process or process flows, complete these steps:

1 Transmit exported file via FTP to the destination OS for import.

2 With AppsFlow open (page 2), click on LogicalApps Utilities in the menu bar, then on Import from File in the Utilities menu. An Import From File form appears:



3. In the Directory box, type the path to the folder that contains the import file.
4. In the File Name box, type the name of the file you want to import. This would be a name displayed by a message at the culmination of a file export (see page 45).
5. Select a value for Debug Level. Ordinarily, select Low; select High instead if you need to uncover the cause of a failed import.
6. Click on the Import button. A concurrent request message displays the ID number of the concurrent program that implements the import. Click on the OK button to clear the message.
7. If you need to import more than one file, click the Clear button to remove current settings and repeat this process as necessary.

Chapter 9

Running AppsFlow Utilities

In addition to features described elsewhere in this manual, the AppsFlow and LogicalApps Utilities menus provide access to several utility features, which can do the following:

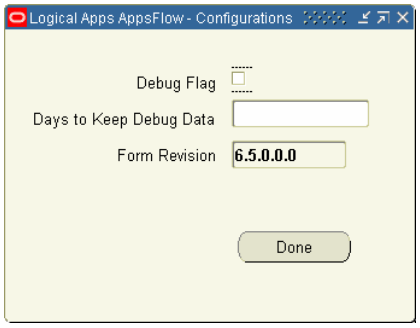
- Launch a background program that monitors the status of constraint processes.
- Generate and display debug data for use in resolving problems with processes created in AppsFlow.
- Gather process rules into libraries.
- Associate Logical Apps form functions with menus or responsibilities.
- Monitor concurrent programs and managers.

Monitoring Constraint Processes

An optional background program monitors processes that contain Check Constraint flows, checking every 30 minutes for any Constraint/Condition element that has been resolved, even though no user has "completed" the notification sent by the Check Constraint rule that called the Constraint/Condition element. The background program completes the Check Constraint flow and moves its process to the next flow rule. The program runs continuously once it has been started until the instance is bounced. To start the program, click on AppsFlow in the menu bar, then on AppsFlow Launch Background Program in the AppsFlow menu.

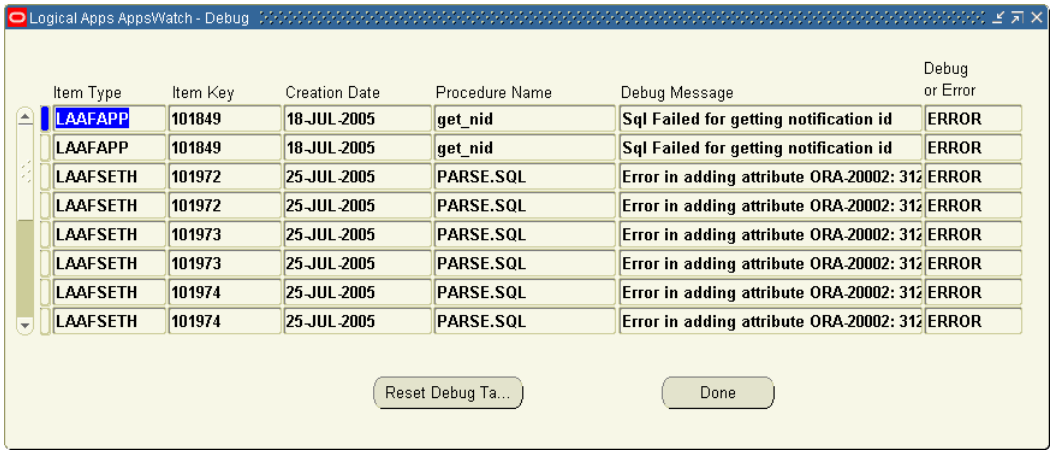
Gathering Debug Data

To generate data about errors that may occur in AppsFlow processing, click on AppsFlow in the menu bar, then on AppsFlow Configurations in the AppsFlow menu. The following Configurations form appears:



In it, select the Debug Flag check box and type a number of days in the Days to Keep Debug Data field. (Note that the form also provides the software revision number.) Then save the changes (click on File in the menu bar, then Save in the File menu) and click on the Done button to close the form.

To review the debug data you generate, click on AppsFlow in the menu bar, then on AppsFlow Debug Data in the AppsFlow menu. The following form appears:



Each row displays information about errors in the running of AppsFlow processes. You can use values from the Item Type and Item Key columns as search parameters to gather more information in the Status Monitor of the Oracle Workflow system.

Click on the Reset Debug Table button to clear old data from the form. Click on the Done button to close the form.

Collecting Processes in Libraries

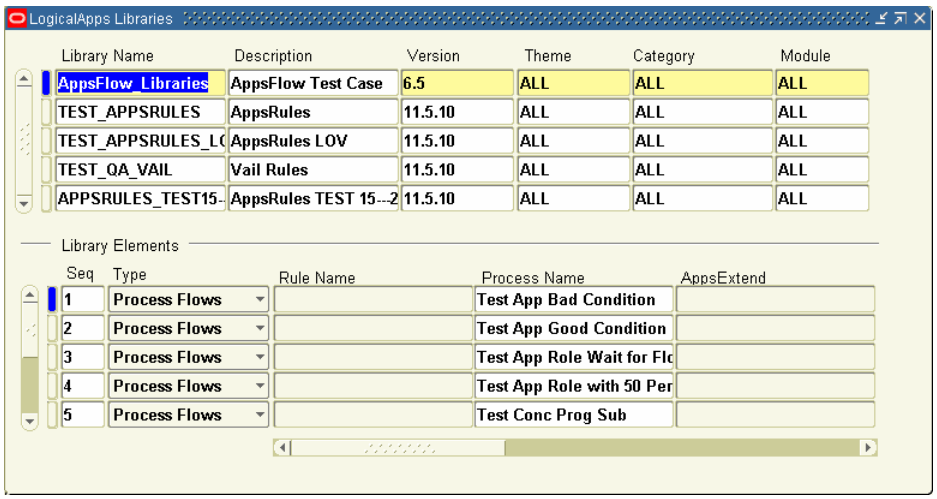
You can gather AppsFlow processes (as well as rules and AppsExtend forms created in AppsForm) into libraries.

As you do, you can characterize the contents of each library by assigning it a theme, category, or module. You have the opportunity to define each class of values in any

way you choose. For example, a theme might be a broadly defined set of items, a category might be a more narrowly defined set of those items, and a module might be a class of applications to which the library items apply. You can also assign each of the items you place in a library to a user. To create themes, categories, or modules, or to identify users to whom library items can be assigned, you would add values to "value sets" that have been created for use with libraries.

Creating a Library

To create a library, click on Logical Apps Utilities in the menu bar, and then on AppsRules Libraries in the Utilities menu. A LogicalApps Libraries form appears:



1. Select a row in the upper grid. Use any of these methods:
   - If the grid contains any empty rows, click in the first one.
   - Click on the New button, which is first on the left in the tool bar.
   - Click on File in the menu bar, then on New in the File menu.
2. In the Library Name field, type a unique name.
3. In the Description field, type a description of the library.
4. In the Version field, type a version number. Note that it can, but does not need, to reflect the version number of the software whose components you are collecting in a library.
5. In the Theme, Category, and Module lists of values, select among values you have defined for characterizing the contents of the library (see "Adding to Value Sets," page 50). Or, if you have not defined values, *ALL* is the only value available to you; select it.

To add AppsFlow processes to the library, use the Library Elements grid:

1. In the Seq field, type a sequence number.
2. In the Type list box, select Process Flows. (Although this list box uses the phrase *process flows*, you cannot add individual flows to the library. Rather, you can add full

process rules, each of which includes the process flows and elements configured for it.)

3 In the Process Name list of values, select one of the process rules you have configured in AppsFlow. (When the Process Flows value is selected in the Type list box, the Rule Name and AppsExtend fields do not accept input.) Then slide the scroll bar to the right to reveal additional fields.

4 In the Complexity list box, select Low, Medium, or High to assess the relative intricacy of the item you are adding. (Your site should develop its own definitions of low, medium, and high complexity.)

5 In the Assigned To list of values, select a user from a set of users to whom a library might be assigned (see "Adding to Value Sets," below). Or, if you have not identified such users, the field does not accept any entry; leave it blank.

6 Optionally, record the state of the library item by selecting appropriate check boxes: Developed, Documented, Tested, QA, or Completed.

7 Optionally, type a brief explanation of this library item in the Description field.

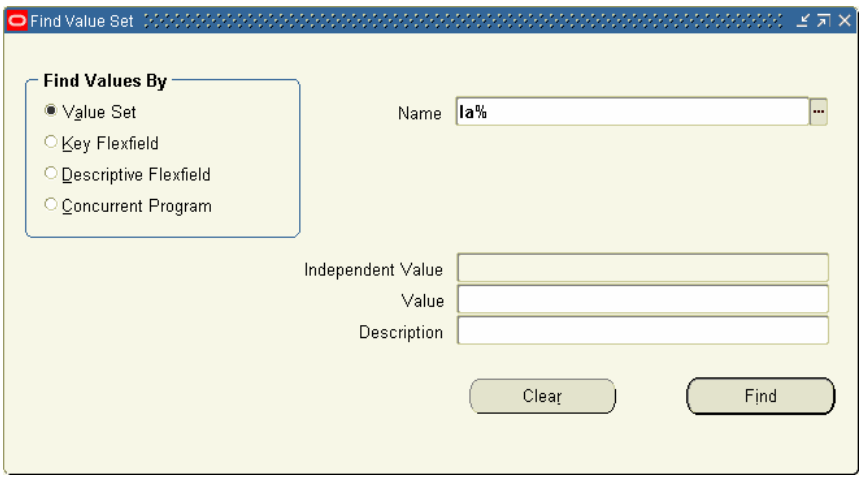
8 Repeat these steps to add as many additional processes to the library as you like, one per row.

9 Save the library: Click on File in the menu bar and then Save in the File menu.

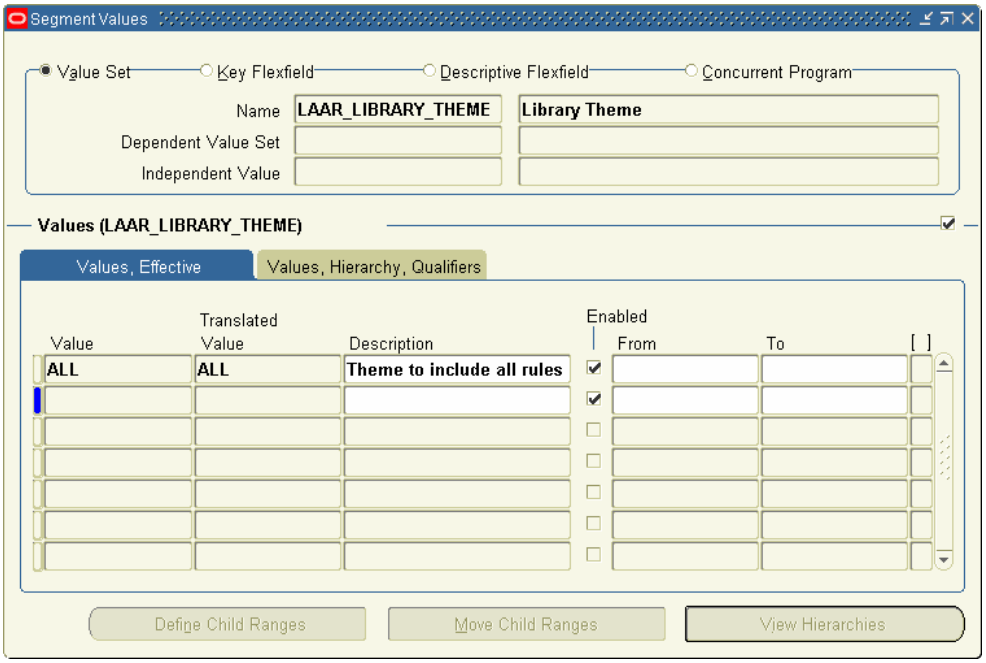
Adding to Value Sets

To define themes, categories, or modules that classify the contents of libraries, or to identify users to whom library items can be assigned, add entries to value sets that have been created for use with AppsRules libraries:

1 With the LogicalApps Libraries form open, click on Tools in the menu bar, and then on Value Sets in the Tools menu. A pair of forms open, with a Find Value Set form initially active:



2. In the Find Values By area, ensure that the Value Set radio button is selected. Then type the string *la%* in the Name box and click on the Find button.
3. A Flexfield Value Sets window opens. In it, click on an entry for the type of values you want to create: LAAR_LIBRARY_THEME corresponds to the Theme field in the Libraries form, LAAR_LIBRARY_CATEGORY to the Category field, LAAR_LIBRARY_MODULES to the Modules field, and LAAR_LIBRARY_ASSIGNED_TO to the Assigned To field. Then click on the OK button.
4. Segment Values form becomes active. In the Effective Values panel, complete one row for each value you want to create.
   - In the Value field, type the name of the value you are creating — for example, the name of a theme if you selected LAAR_LIBRARY_THEME as you opened the form. The Translated Value field defaults to the same entry; you can't change it.
   - In the Description field, type a brief explanation of the value.
   - Ensure that the Enabled check box is selected for the value to take effect.
   - To define a limited period for the value to remain in effect, enter starting and ending dates in the From and To fields. For each, select a date in the pop-up calendar that appears when you click on the list-of-values icon, or type a date in the format configured for your instance of Oracle Applications. Or, to allow the value to remain in effect immediately and indefinitely, leave the From and To fields blank.

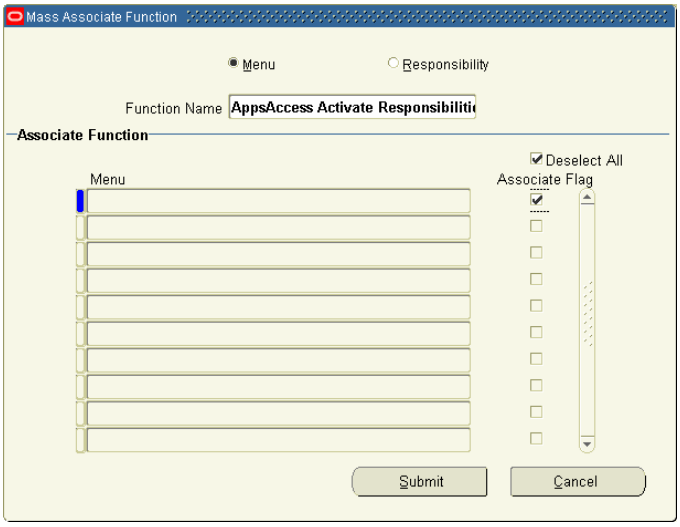


5. When you finish entering values, save your work: Click on File in the menu bar, and then Save in the File menu.

Using the Mass Associate Utility

A Mass Associate utility enables you to add Logical Apps form functions to Oracle Navigator menus or to responsibilities which they are not already associated:

1. Click on LogicalApps Utilities in the menu bar, and then on Mass Update function in the Utilities menu. The following form appears:



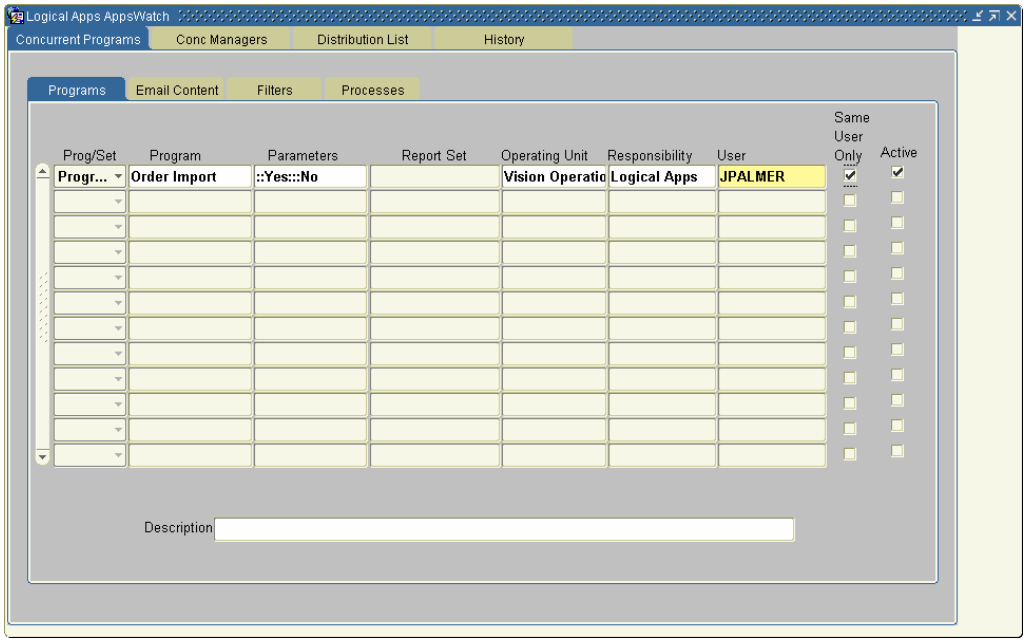
2. In the Function Name field, select the Logical Apps form function you want to associate with menus or responsibilities.
3. Click on the Menu or Responsibility radio button. The Associate Function list then displays all menus or responsibilities not already associated with the function selected in the Function Name field.
4. Click the Associate Flag check box corresponding to each menu or responsibility you want to associate with that function.
5. Click on the Submit button. Users with access to the newly associated menus or assigned to the newly associated responsibilities then have access to the function.

Monitoring Concurrent Programs

An AppsWatch feature helps to track concurrent request processing and provides a mechanism for distributing reports via workflow notifications:

1. Select AppsFlow in the menu bar, then AppsFlow Monitor Conc Programs from the AppsFlow menu. An AppsWatch form appears (shown at the top of page 53).
2. In the Prog/Set list box, select Program or Report Set. Depending on this setting, select a program in the Program field or a report set in the Report Set field.
3. Go to the Parameters field; pop-up windows prompt for program parameters. When you finish making selections, they appear in the Parameters field.

   Or, select Ignore Parameters in the Tools menu; in the Ignore Parameters form, click the Ignore check box for each parameter you wish to ignore.

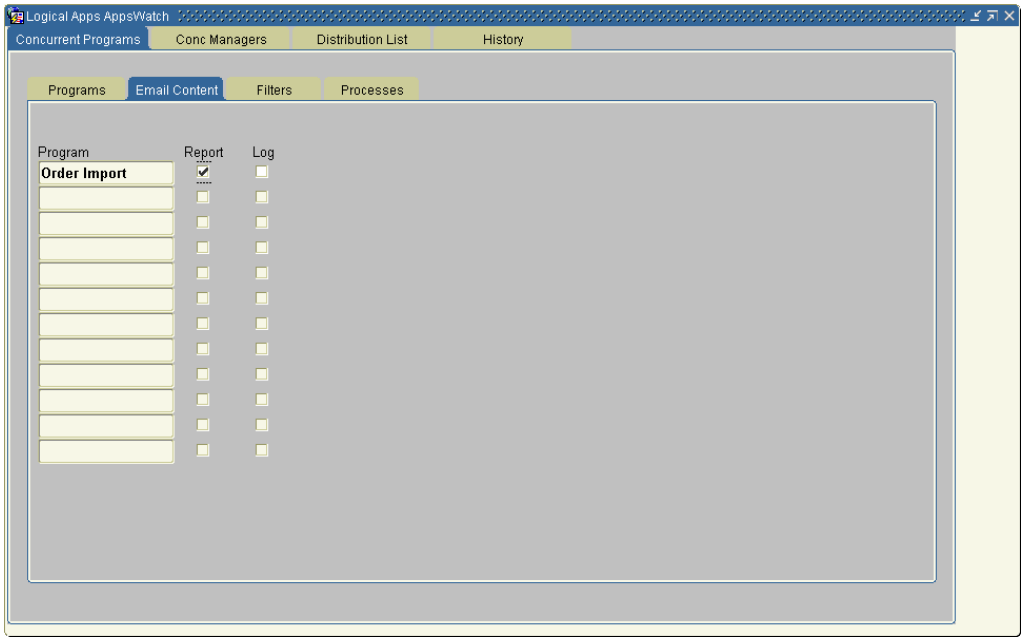


4 Optionally, select an operating unit in the Operating Unit field, a responsibility in the Responsibility field, or both. Only when the request is run from the unit and responsibility specified would the notification get sent.

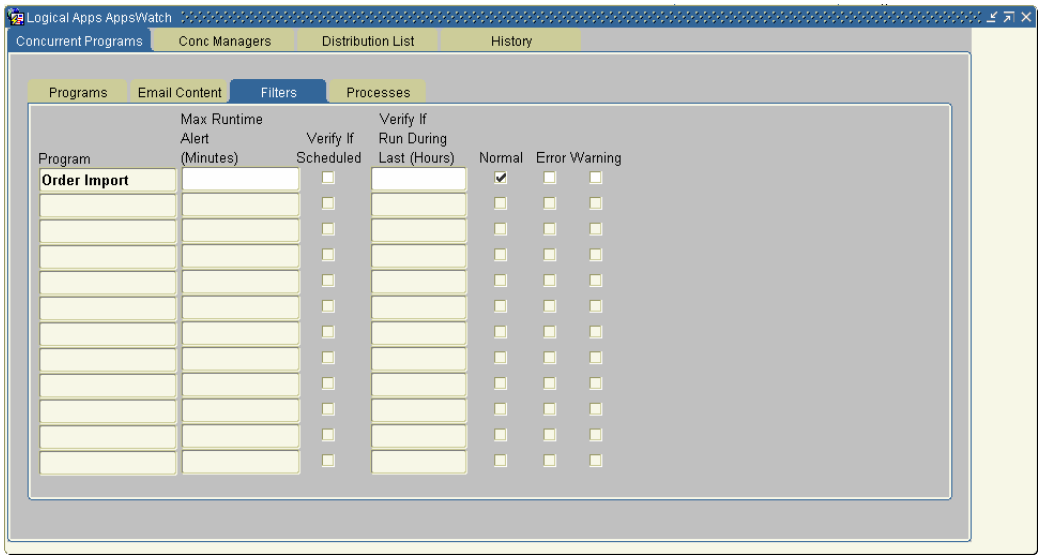
5 In the User list of values, select the ID of the user who is to be notified. Select the Same User Only check box to generate notifications only when the specified user runs the program.

6 Click the Active check box to activate the row.

7 Click on the Email Content tab and select or clear the Report and Log check boxes. When either is selected, the email notification includes the component that is checked: a report or a log generated by the program.



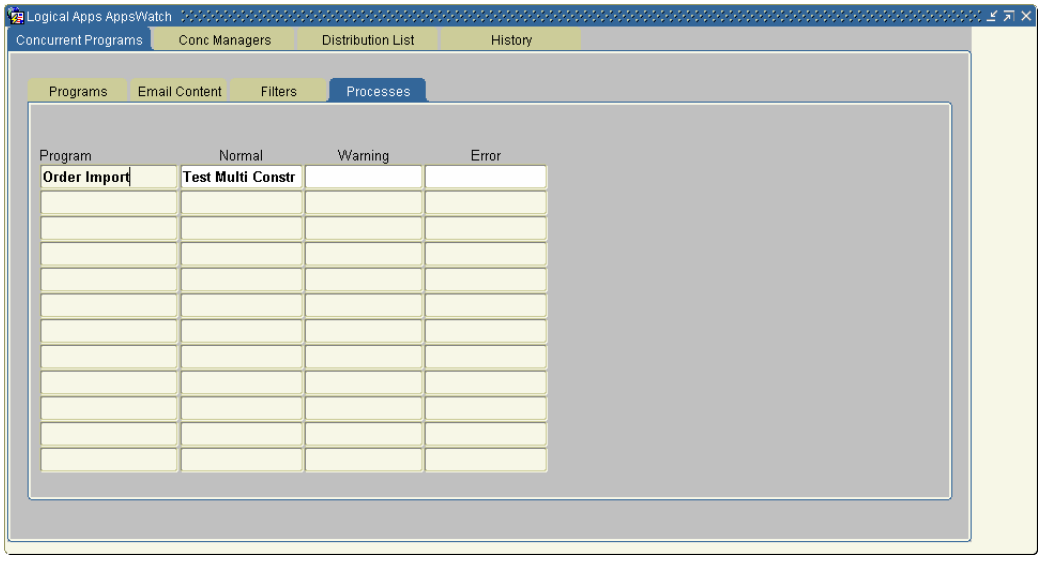
8 Click on the Filters tab to create additional filters:



In this form:

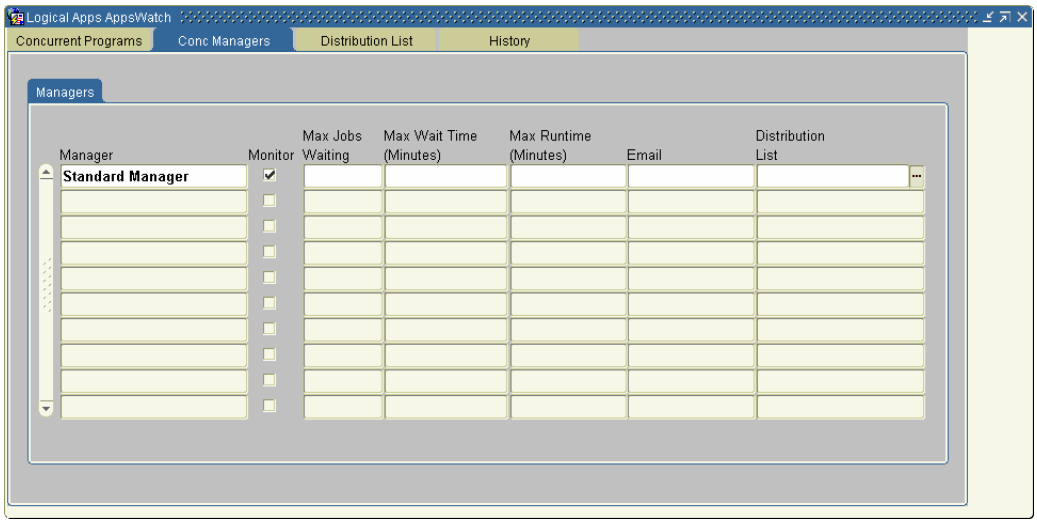
- In the Max Runtime Alert field, set a number of minutes. If the program runs longer than this time, a notification is sent.
- Select the Verify if Scheduled check box if the program runs on a schedule. If it misses its schedule, a notification is sent.
- In the Verify if Run During Last field, type a number of hours. If the program is not run within that interval, a notification is sent.
- Select any combination of the Normal, Error, or Warning check boxes to receive a notification when the program finishes running in any of these states.

9 Click on the Processes tab to specify AppsFlow processes to be executed when the concurrent program finishes running. You may choose up to three processes, one each to be run when the program ends in normal, warning, and error states.

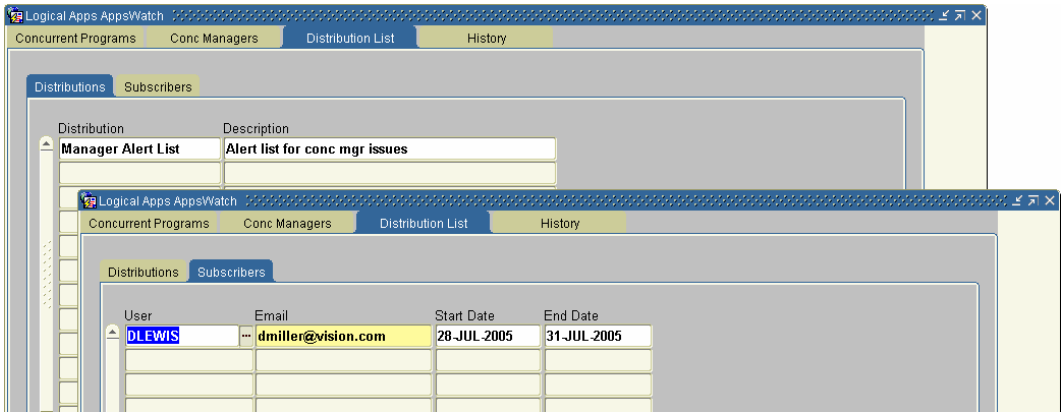


10 Click on the Conc Managers tab to specify concurrent managers that need to be monitored, and who must be notified if managers go down or are backed up.

- In the Manager list of values, select a manager to be monitored. Select the Monitor check box.
- Enter values for the maximum number of jobs that can be in the queue, the maximum amount of wait time permitted, and the maximum runtime permitted. Alerts are sent when any of these values is exceeded.
- In the Email field, type the email address to which alerts are to be sent, or in the Distribution List field, select a distribution list. (These are configured in a panel that appears when you click on the Distribution List tab.)



11 Click on the Distribution list tab to create the email distribution lists you can select in the Conc Mangers panel. Within the Distribution panel, click on the Distributions tab to name and describe the distribution list, then click on the Subscribers tab to select its members — for each (one per row) a user ID, email address, and start and end dates:



12 Click on the History tab to display records of all monitored concurrent managers and programs. Entries in this history log are purged after the number of days entered in the Retain Log (Days) field.