Workflow Implementation Manual

Document Version 0.8

CONTENTS

Preface

This manual explains in detail how to use Workflows and configure and maintain workflows throughout your system.

This preface contains these topics:

Audience

This guide is intended for users who are comfortable running some system administration operations, such as creating users and groups, adding users to groups, and installing operating system patches..
Users who administer Permissions need Administration rights.

Conventions

The following text conventions are used in this document:
Convention Meaning boldface Boldface type indicates graphical user interface elements associated with an action, or terms defined in text or the glossary.
italic Italic type indicates book titles, emphasis, or placeholder variables for which you supply particular values.
monospace Monospace type indicates commands within a paragraph, URLs, code in examples, text that appears on the screen, or text that you enter.

Purpose

This manual is intended for]PO[administrators and explains how to:

- Create Workflows
- Configure existing Workflows
- Import/Export Workflows
- Create Parallel Workflows
- Create hierarchical Workflows
- Attach SmartForms & Letters to Workflows

Glossary

User: A person who has a login to the]PO[system.

Role: A group of users, set up within workflows to facilitate assignment of workflow tasks and permissions.

Action: A function that one role may have permission to do and another may not. This can be of the form of a link to a page or a particular access mode for a page.

Workflow: A workflow is the process by which a case is processed from start to end.

Permission: The ability that a person has to perform an action (or function). This is "decided" by looking up the function and the role the person is in. Some permission's are workflow dependant.

Workflow dependant permission: A function that is on a screen that can be said to be within a specific workflow (at a particular point in time) can be workflow dependant. This means that the permission is checked against that specific workflow e.g. if you are in the APS you are always within a specific workflow and this workflow is checked when checking the permission in the role permissions matrix for example "Extreme actions" (in the Application Processing Screen).

I Bar: This is the information panel at the top of the Application Processing Screen, containing info about the application, eg. App ID, Applicant name etc.

General (or NON-workflow dependant) permission: Any permission that is not specifically related to a workflow is a "general" permission e.g. View person. These permissions are set up on and checked against the default workflow)

1 INTRODUCTION

A workflow may be described as a process by which a case is processed from start to end. Each case could have a number of stages it might go through in order to reach the end of its process, and depending on certain conditions it might have a number of paths to choose from before it can finish its journey.

Examples of such cases could be an application for a grant, a complaint, goods to be manufactured etc. For the purpose of this document we will use the example of an application for a grant.

In business processes, Workflow management is used to provide a solution for the following:

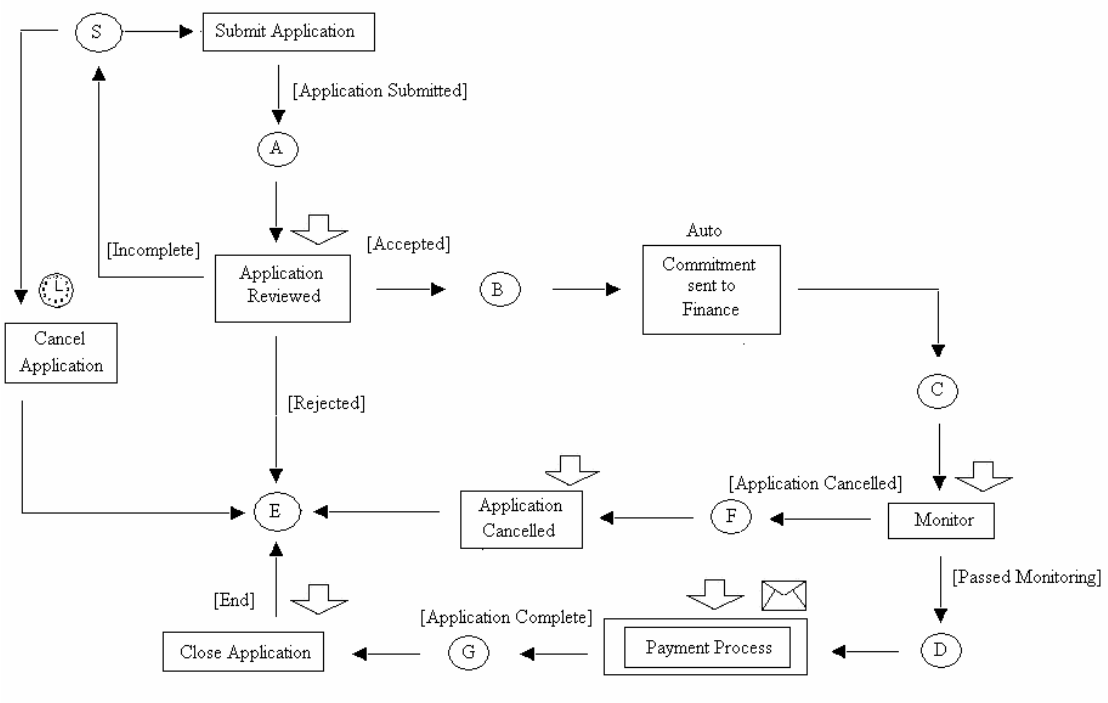
- Controlling
- Monitoring
- Optimising
- Supporting

The definition of a workflow (what needs to be done to a case and in what order), are formalized in terms of a computational model called **Petri nets.** Petri nets are an established tool for modelling and analysing processes.

2 THE WORKFLOW CONCEPTUAL MODEL

The following diagram is simple workflow set up for the process of applying for a grant from start (S) to end (E).

Fig : 1.0 – Workflow for Application Process



- The **circles** are called **places** and represent the state of the task before it has started.
- The **rectangles** are called **transitions** and represent the **tasks** to be performed.
- The **arrows** are called **arcs** and these, point to the next stage(s) possible.
- A **Token** is the form representing a case. Tokens stay in places.
- The transition within a rectangle i.e. double rectangle represents a child workflow (a sub process).

Places are inactive. The function of a place is to hold **tokens** representing the state of the process. If, for example, there's a token in place 'A' above, then that means we're ready to Review the Application

Transitions are active. They move tokens from their **input places** (the places that have an arc pointing into the transition) to their **output places** (the places you get to by following the arcs going out of the transition). When this happens, the transition is said to **fire**.

Transitions can only fire when there's **at least one token in each input place**. When that is the case, the transition is **enabled**. When the transition is **enabled** means it is **able to fire**.

The time the transition is enabled and the time it fires are different. The thing that causes an enabled transition to fire is called **trigger**.

There are four different types of triggers:

1. Most transitions will normally be performed by a person. This is called a **user trigger** and is symbolized with a fat arrow pointing to the task.
2. Some tasks, such as the user updating financial information, are beyond the control of the workflow software. The workflow software receives a message that the task has been performed, and thus these are called **message trigger**, symbolized with an envelope.
3. Transitions with an **Automatic trigger** are performed by the system as soon as the transition is enabled. The 'Commitment sent to Finance' task above is such a transition. When fired, it will execute some code to send off the financial details to the finance system. All other transitions can also execute application-specific code when they fire.
4. Some automatic transitions need to occur at a certain point in time. The 'Cancel Application' transition above has a **time trigger**, symbolized with a stop watch, which will automatically cancel the application if the user hasn't started their task.

2.1 Routing

When the workflow is started, a token is placed in the **start place** (S in the example). This enables the user transition 'Submit Application'.

The transition fires with the submission of the application form. When submitted, it produces a token in place A and the application is ready for review.

When reviewed, the application can be deemed as accepted, rejected or incomplete. If it was accepted, it produces a token in place B. If it was rejected then it produces a token in place E (i.e. the application is closed) and if the application was Incomplete, the token goes back to the start, place S. Thus, the outcome of the application review governs the further routing of the process.

The rule is that firing a token **consumes one token from each of its input places, and places a token on each of its output places, for which the guard is true**.

The **guard** is a condition, in this case the [accepted], [rejected] and [incomplete] on the arcs going out of 'Application Reviewed'. Guards are what allow us to do **conditional routing**. The 'Application Reviewed' transition acts as an **or-split**, because it chooses either one route or the other.

The above form of or-split is called an **explicit** or-split. There's another form of conditional routing, which is the **implicit or-split** that chooses between the transitions 'Submit Application' and 'Cancel Application'. Since there's only one token in place S, only one of the two transitions can have it. But, contrary to the explicit or-split, where the decision is explicitly made, the choice between 'Submit Application' and 'Cancel Application' is made as late as possible.

Both transitions will be enabled when there's a token in place S (i.e. when the application has been created). If the user submits their application before the timed 'Cancel Application'

transition times out, 'Cancel Application' is never fired. And vice versa: If the application is cancelled, then the user won't be able to submit their application and will have to create a new one. Thus, the choice is made implicitly, based on the timing.

The guard will generally depend on **case attributes**. The ‘Application Reviewed’ transition above will set a case attribute to 'accepted', ’rejected’ or 'incomplete', and the guard will check this value to determine its result. Case attributes can hold more complex values than simple yes/no values, but the guard must always be either true or false.

The workflow package also handles **parallel routing**, where two or more things happen concurrently or in no particular order. This is done by having a transition produce more tokens than it consumes, which is called an **and-split**. To re-synchronize after an and-split a transition is used that waits for both concurrent threads to finish before it continues. This is called an **and-join** and is simply a transition that consumes more tokens than it produces.

3 CHANGING AN EXISTING WORKFLOW

Use of workflow or, more technically Petri nets, in business applications is a very complex area. Changing workflows while retaining the data within them (the applications) is also quite complex and difficult to document or train people for. Changing workflows with applications in them can be very easy in some cases or more difficult in others. The complexity depends on the change being made and the nature of the workflow. Situations can arise whereby clients may need to contact]po[to help design the best way to make changes. However the following principles apply:

- Once the workflow is being used with live applications, it is still possible to change the workflow in most cases.
- Adding or removing an Arc is always simple, only the possible path through the workflow has changed. It is important to consider the change, particularly when removing an ARC so as not to introduce any "dead ends" in the workflow. Of course if a dead end is introduced then the arc can be recreated.
- Removing a stage requires the administrator to think about the applications at that stage, the functionality that happens at it and the other stages that are linked to it.
- If no applications have gone through a stage, the stage can simply be removed and the relevant arcs set up appropriately.
- Once applications have passed through a stage then there are records in the application log relating to that stage and the stage must be retained for data integrity reasons. . A stage falling into this category should be 'removed' from the workflow by changing the guard leading to the stage. See 7.8.1.
- If the changes to the workflow are so great then the administrator should consider creating a new workflow for the simple reason that this might actually be a new scheme.

3.1 Verify Integrity

When a new attribute is added to a workflow and there are current cases is the workflow, these cases must be altered slightly to accommodate the new attribute.
This is because when a case is started all of the attributes in the workflow are given a value on the acs_attribute_values table. If an attribute is added after a case has been started this case will not have a default value for the new attribute, and so when the case reaches the point in the workflow where the new attribute is used it will not be able to proceed, or may "disappear".

To avoid this after a new attribute is added we must click on the 'Verify Integrity' button on the workflow admin screen. This checks that all active cases in the workflow have a value for each attribute, and adds a default of 'f' for any that don't.

4 PROTOTYPING

Before creating a workflow, it is important to carry out some preliminary tasks before using the software. Such tasks include Prototyping.

Prototypes should be developed for each workflow and be used as part of the functional specification process. Comprehensive prototyping i.e. a three phase approach will identify risk areas and promote a collaborative effort on the part of the business analysis team, the technical team and key end users throughout the course of the project.

4.1 Prototyping Phases

In preparing for phase 1 of the prototyping process, the business analysis team should deliver detailed process flow diagrams describing the end-to-end process of the respective workflows.

The recommended approach to prototyping workflows within]PO[, is to deal with the simplest workflow first. The knowledge gained in completing this workflow will be of benefit to]po[and the Client in prototyping and delivering the more complex workflows that follow.

Prototyping will primarily involve the use of]PO[workflow, smart forms and template letters. Contacts, regions and business rules and their involvement in each workflow will be identified during the prototyping phases but will not be addressed in the prototypes themselves.

4.1.1 Phase 1

Through whiteboard discussion,]po[and the business analysis team will review process flow diagrams for a given workflow. Through the use of]PO[workflow; smartforms, template documents, workflow stages and workflow roles associated with each stage will be documented. These documents may be revised several times before initial build. A development system will be used by]po[to produce a draft workflow reflecting the template documents. The Client will review the workflow and detail refinements for prototype phase 2.

As a pre-requisite to phase 2 prototyping, the business analysis team will be required to produce a complete list of documents and forms associated with the workflow. Each item on the list will need to be cross-referenced against the corresponding stage in the workflow.

4.1.2 Phase 2

This phase involves]po[refining the workflow roles and stages in accordance with the phase 1 review. Pre-prepared template letter and smart forms will be added to the appropriate workflow stages through the use of]PO[workflow options.]po[with the aid of the business analysis team and key Client users will review the workflow.

As a pre-requisite to phase 3 prototyping, the business analysis team will be required to identify business rules and exception cases relating to the workflow. Any additional workflow refinements, including additional letters and forms should also be identified.

A log of the identified business rules (whether they will be used in the workflow or not) should be updated so that it can be referred to at a later stage if required.

4.1.3 Phase 3

This phase involves refining the workflow, letters and forms in accordance with the phase 2 review.]po[may incorporate some simple business rules and deadlines into the workflow. The business analysis team in association with key users will produce comprehensive smart forms and template letters. A complete and final review of the workflow will take place involving all parties with a view to sign off of final prototype by the Client.

A full build along with system and integration test will follow phase 3 with a view to delivering the completed scheme to the Client user acceptance system.

]po[may play a less involved role in prototyping subsequent workflows as the Client technical and business analysis teams grow their knowledge of]PO[and the workflow development process.

5 APPLICATION PROCESSING SCREEN

Before creating a workflow, it is important to know what the user screen will look like when completing a task. References to this screen are made in the subsequent sections so please refer back to this screen when adding panels, attributes etc.

In the example below we can see:

- Panels – Application Summary, Application Log & Options
- Role Assignment drop-down – 'Assign Grants Team'
- Logical Attributes – Buttons 'Enter Details' & 'Not Eligible'
- Journal – Application audit trail

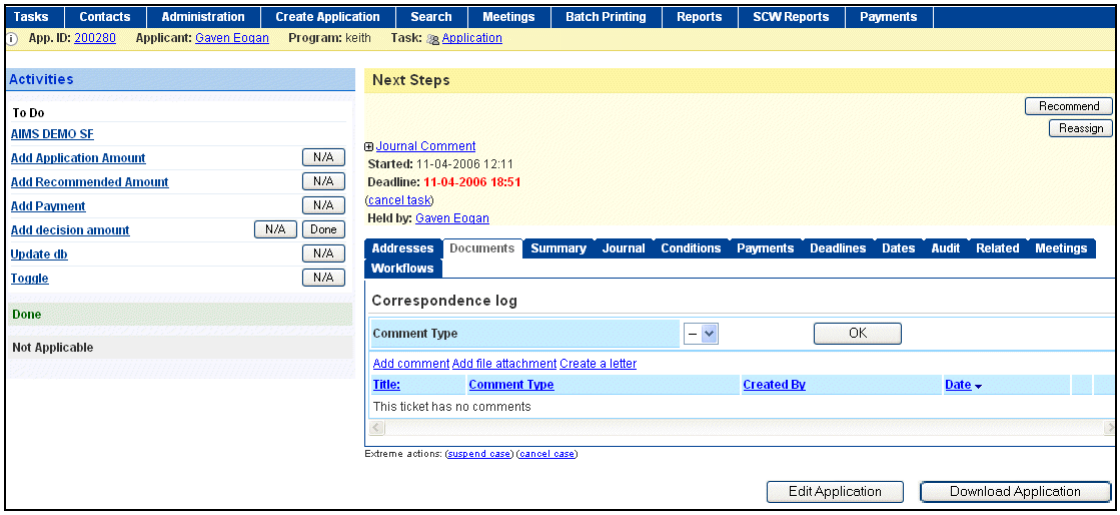
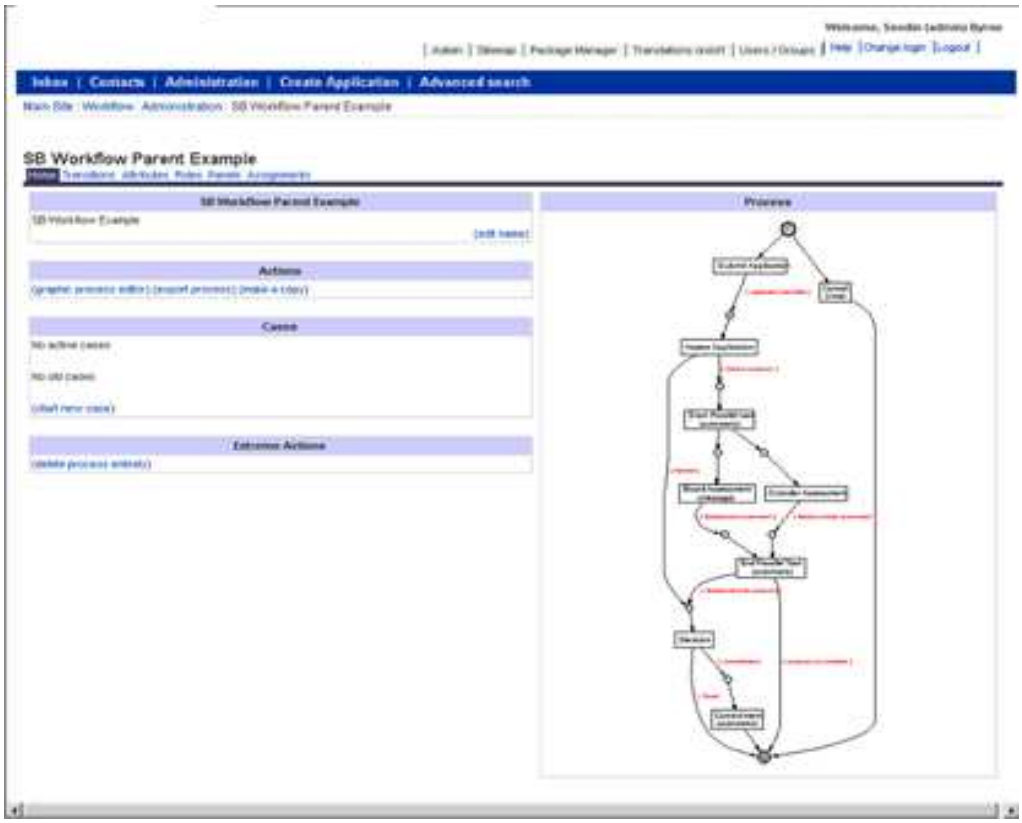


Figure 1 Application Processing Screen

6 WORKFLOW FUNCTIONS

A prepared workflow is shown below. This page is available from the Business Process Administration page (Workflow admin page) when the workflow you wish to edit is selected. A description of each link and function will be described in detail in the following section.

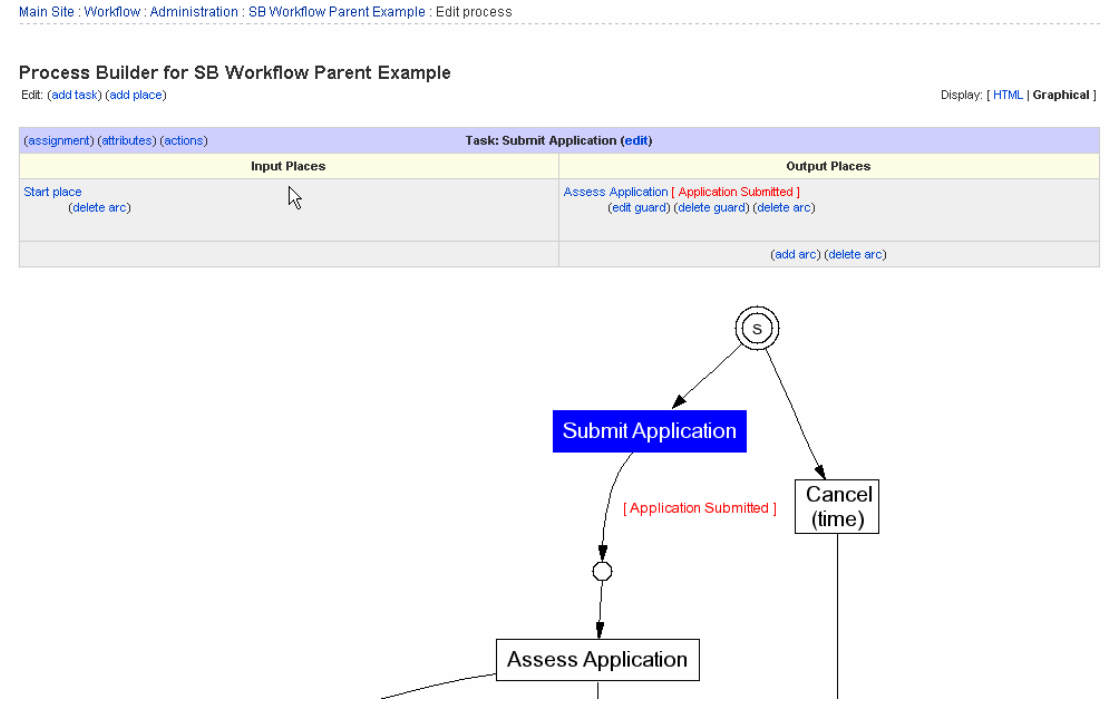


6.1 Home

This is the workflow home page for an individual workflow. Options available are described below.

6.1.1 Graphic Process Editor

This is where the workflow is produced, i.e. transitions are joined to places with arcs and vice versa. Guards are added and attributes can be assigned to transitions.



6.1.1.1 Tasks

Tasks are another name for Transitions. The boxes in a workflow diagram represent these.

Tasks may be triggered in four different ways.

1. User Tasks: A user must manually select a button to move the task to the next step
2. Automatic Tasks: This task is moved on automatically and immediately by the system.
3. Time Tasks: Also an automatic task but is only triggered at a certain time calculated by a value passed into a the callback.
4. Message Tasks: These may also be automatic but are only triggered when a message is received back from a procedure that is called.

6.1.1.1.1 Edit Task

Any of the task properties may be edited at any time, e.g. Task Name, Trigger Type, Role, and Instructions etc.

6.1.1.1.2 Assignment

Roles may be assigned to the task using the 'Assignment' link on the graphical process editor screen. When a role is added a drop-down containing the members of that role will be displayed on the application-processing screen. The user may then select whom they wish to assign the task to.

6.1.1.1.3 Attributes

Please see the sub section 6.3 below for a full description on Attributes.

6.1.1.1.4 Actions

An action is another name for a trigger that runs a callback procedure. Actions, don't by themselves alter the state of the process, they only have side effects. They can, however, set workflow attributes, which can then be queried by guards.

You can use an action to implement a process that doesn't require human intervention. If you have a process, such as updating certain application details and sending finance details to another system, you can model the order and the dependencies in a workflow process, and have the actions performed automatically. By doing it this way, you separate the logic governing the overall process from the technicalities of actually performing the actions.

6.1.1.1.4.1 Enable

The Enable action occurs when a transition becomes enabled. This means that the transition has at least one token in each of its input places.

Please see How to run a background procedure before a task is started for more information.

6.1.1.1.4.2 Fire

The Fire action occurs when a transition is being fired. When a transition fires, it **consumes** one token from each of its input places and **produces** one token in each of its output places, **for which the guard evaluates to true.** A transition must be enabled in order for the transition to fire.

Please see How to run a background procedure when a task is complete for more information.

6.1.1.1.4.3 Time

Timed transitions are automatic transitions that trigger at some pre-specified time. You must supply a callback to compute the time that the transition should fire. This callback will be executed each time a timed transition becomes enabled, and should return an Oracle date. At the date and time specified by this returned date, the transition will automatically fire.

Please note that a timed transition will only work from a place that has no attribute assigned to get to that place. It will also not work in a Parallel section of a workflow.

For further information on how to set timed transitions please see the section below on How to move a task along in the process if a certain time has elapsed.

6.1.1.1.4.4 Deadline

Tasks may have deadlines. In order to compute the deadline, a callback can be executed, which must return the deadline as an date. A deadline date may be set either automatically or manually by user input.

For further information on how to set deadlines please see the section How to set deadlines against a task below.

6.1.1.1.4.5 Hold Timeout

When a user starts a task, he obtains a lock on that enabled transition, and the tokens it will consume. But you might not want a user to hang on to a task forever without finishing it. Thus, you can supply a hold timeout date, which is similar to a deadline.

Please see the section on How to remove a task from a user's Inbox if a certain time has elapsed for further information.

6.1.1.1.4.6 Notification

The notification callback will get called whenever a user is assigned to this transition. If no notification callback is provided, the party assigned to the task will be notified automatically by the workflow engine.

Please see the section on How to run a background procedure when a user is assigned to a task for more information.

6.1.1.1.4.7 Unassigned task

Whenever a transition is enabled, but there are no assignees, this callback will get called. You will typically use this to notify some principal that there's an unassigned task that they will want to take a look at. This is not used in the]PO[system, however a custom callback could be written for this purpose.

Please see the section on How to run a background procedure when a task has been un-assigned for further detail.

6.1.1.2 Places

Places act as token holders for transitions. Each transition must have at least one input place but may have multiple. In order for a transition to become enabled it must have a token in each of its input places. When the transition then fires it puts a token in each of its output places where the guard returns true.

6.1.1.2.1 Add place

Each transition must have at least one place.

6.1.1.2.2 Delete place

A place may only be deleted when there are no active cases in the workflow.

6.1.1.3 Arcs

Arcs allow the Workflow to move from a place to a transition or vice versa.

6.1.1.3.1 Add arc

Arcs are added between places and transitions. A place may have multiple arcs coming into it and leaving it. A transition may have multiple arcs entering and leaving it.

6.1.1.3.2 Delete arc

Selecting the transition where it begins, then ‘delete arc’ and then selecting the place where it ends can delete an arc.

6.1.1.4 Guards

A guard is a PL/SQL procedure that resides in the WF_CALLBACK package. Guards can be executed when an action is performed in the application processing screen, for example when the user clicks on a button in the Action panel in the Application Processing Screen.

6.1.2 Export Process

This function allows a Workflow to be exported from the]PO[system. It exports the process definition as a SQL script, which can subsequently run on a different system, in order to recreate the process there.

There are 4 different methods of exporting :

1. Download the SQL script
2. View the SQL script in your browser
3. Show the SQL script in a <textarea> field, so you can copy and past it
4. Save it to a file on the server

The procedure called for exporting workflows is wf_export_workflow and can be found in 'Packages/acs-workflow/tcl/workflow_procs.tcl'.

This process extracts the following data :

Table	**Fields**
	Workflow name and description
Places	place_key, place_name, sort_order
Roles	role_key, role_name, sort_order
Transitions	transition_key, transition_name, role_key,sort_order, trigger_type
Arcs	transition_key, place_key, direction, guard_callback, guard_custom_arg, guard_description
Attributes	attribute_name, datatype, pretty_name, default_value
Transition_attribute_map	transition_key, attribute_name, sort_order
Transition_role_assign_map	transition_key, assign_role_key
Context_transition_info	context_key, workflow_key, transition_key, estimated_minutes, instructions, enable_callback, enable_custom_arg, fire_callback, fire_custom_arg, time_callback, time_custom_arg, deadline_callback, deadline_custom_arg, deadline_attribute_name, hold_timeout_callback, hold_timeout_custom_arg, notification_callback, notification_custom_arg, unassigned_callback, unassigned_custom_arg
context_role_info	role_key, assignment_callback,

	assignment_custom_arg
context_task_panels	transition_key, sort_order, header, template_url, overrides_action_p, only_display_when_started_p

Please note that any data with a unique id that are taken from a sequence will be created on import of the workflow as they cannot be exported. Also cases created for that workflow are not exported.

6.1.3 Make a copy

If a Workflow with active cases needs changing, i.e. if a process changes, a copy of the Workflow should be taken and changes made to the new Workflow. When selected, a new name must be entered for the workflow. After selecting the Copy button, a script will be executed, the results of the copying process are displayed to screen informing what tables have been copied.

6.1.4 Start new case

An administrator developing a workflow that wants to test out the workflow uses this function. By selecting this link, a new case is created and the administrator can use the debugging mode to track the case through the Workflow path.

For more information, please see How to Debug a Workflow below.

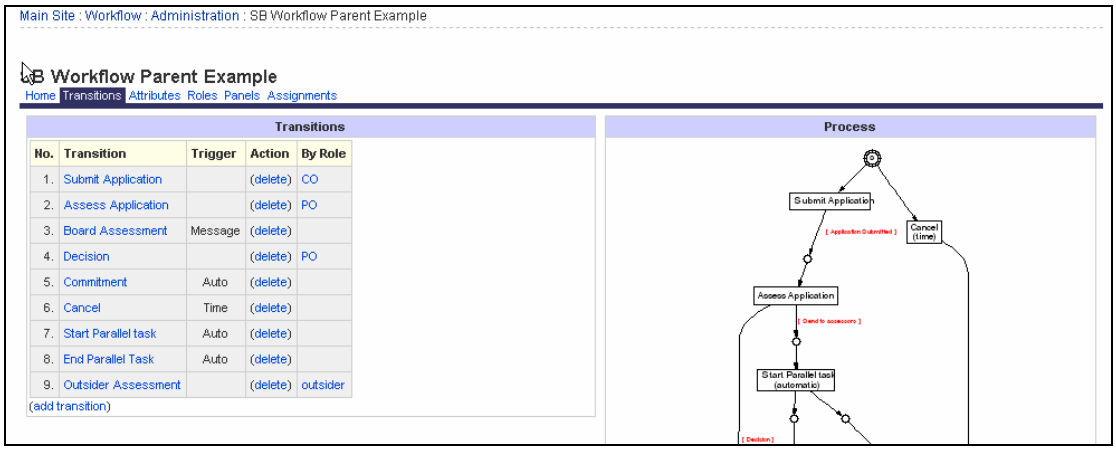
6.1.5 Delete process entirely

This should be treated with great care. When selected, all cases within the selected Workflow are deleted. All cases must be deleted when adding/deleting transitions to a Workflow.

Please see How to Delete a Workflow for more information.

6.2 Transitions

Transitions are another name for Tasks. They are represented by the boxes in a workflow diagram.



6.2.1 Add a transition

When adding a transition you must specify:

- The task name
- The trigger type
- The role

6.2.2 Edit a transition

A transition may be edited in order to change the name, trigger type, role etc.

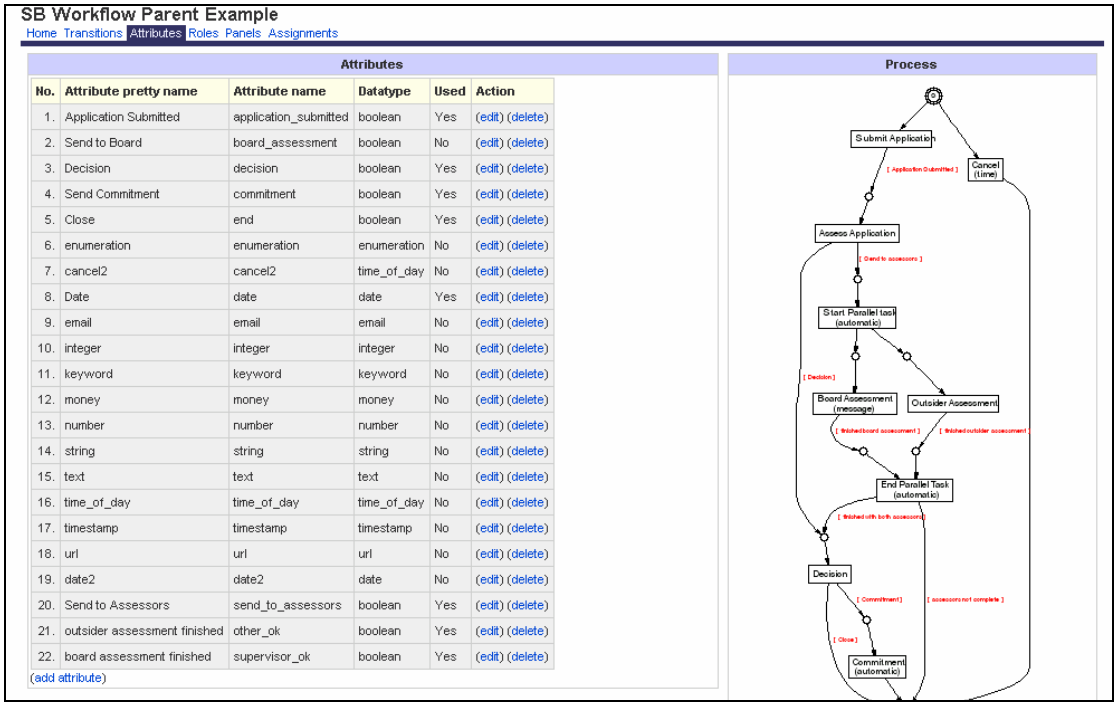
6.2.3 Delete a transition

A transition may only be deleted when there are no active cases in the workflow

6.3 Attributes

Attributes are added to a task in order to allow the user to send the task on/back in the workflow. When an attribute is added it appears as a button or field (depending on the data type) on the application-processing screen. A Boolean attribute is shown as a button while all others are shown as an input field. These fields are not labelled.
An attribute can be of data type:

- Boolean
- Date
- Email
- Enumeration
- Integer
- Keyword
- Money
- Number
- String
- Text
- Time_of_day
- Timestamp
- url



No.	Attribute pretty name	Attribute name	Datatype	Used	Action
1.	Application Submitted	application_submitted	boolean	Yes	(edit) (delete)
2.	Send to Board	board_assessment	boolean	No	(edit) (delete)
3.	Decision	decision	boolean	Yes	(edit) (delete)
4.	Send Commitment	commitment	boolean	Yes	(edit) (delete)
5.	Close	end	boolean	Yes	(edit) (delete)
6.	enumeration	enumeration	enumeration	No	(edit) (delete)
7.	cancel2	cancel2	time_of_day	No	(edit) (delete)
8.	Date	date	date	Yes	(edit) (delete)
9.	email	email	email	No	(edit) (delete)
10.	integer	integer	integer	No	(edit) (delete)
11.	keyword	keyword	keyword	No	(edit) (delete)
12.	money	money	money	No	(edit) (delete)
13.	number	number	number	No	(edit) (delete)
14.	string	string	string	No	(edit) (delete)
15.	text	text	text	No	(edit) (delete)
16.	time_of_day	time_of_day	time_of_day	No	(edit) (delete)
17.	timestamp	timestamp	timestamp	No	(edit) (delete)
18.	url	url	url	No	(edit) (delete)
19.	date2	date2	date	No	(edit) (delete)
20.	Send to Assessors	send_to_assessors	boolean	Yes	(edit) (delete)
21.	outsider assessment finished	other_ok	boolean	Yes	(edit) (delete)
22.	board assessment finished	supervisor_ok	boolean	Yes	(edit) (delete)

6.3.1 Add an attribute

When adding an attribute we give it a name (attribute_name), a pretty name which is what will appear on the button, a data type and maybe a default value.

6.3.2 Edit an attribute

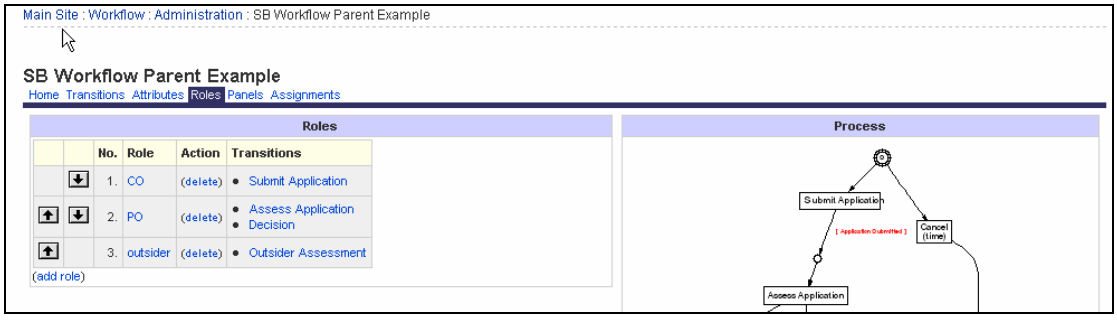
The only part of an attribute that can be edited is the pretty name, the text that appears on the button.

6.3.3 Delete an attribute

Attributes may only be deleted if no active cases exist for that workflow.

6.4 Roles

A role is a group name for one or more users that can be grouped together by their function in a workflow. Roles may be added to the workflow and associated with a transition.



		No.	Role	Action	Transitions
	↓	1.	CO	(delete)	• Submit Application
↑	↓	2.	PO	(delete)	• Assess Application • Decision
↑		3.	outsider	(delete)	• Outsider Assessment

6.4.1 Add a role

When adding a role you must specify a role name. The payment approval limit allows that user to approve applications for an amount less than or equal to the limit entered.

6.4.2 Edit a role

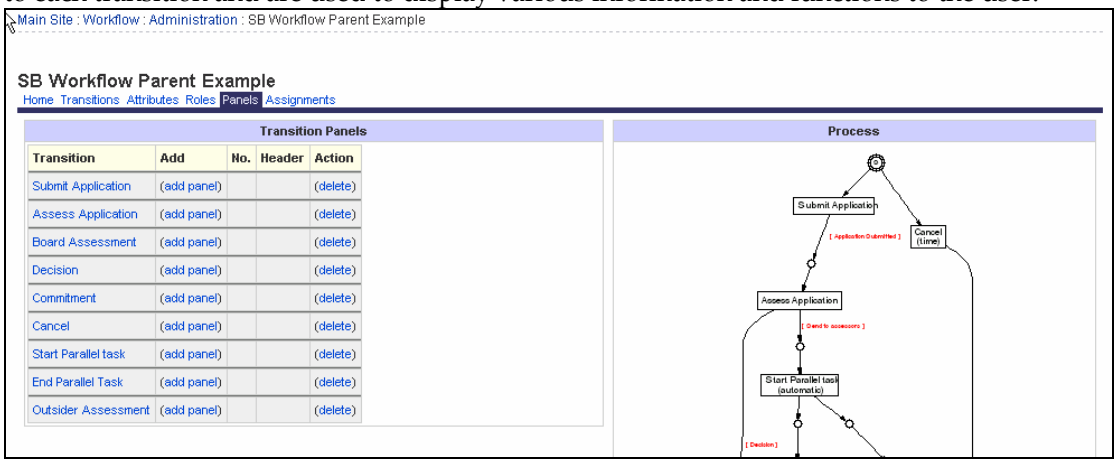
A role's name and payment approval limit may be edited at any time.

6.4.3 Delete a role

Roles may only be deleted when there are no active cases in the workflow.

6.5 Panels

The application-processing screen is comprised of various panels. These panels may be added to each transition and are used to display various information and functions to the user.



6.5.1 Add a panel

When adding a panel you must enter a Header e.g. Application Summary and the URL for where the panel template is stored, e.g. /wf-templates/ticket-summary/ticket-summary

6.5.2 Edit a panel

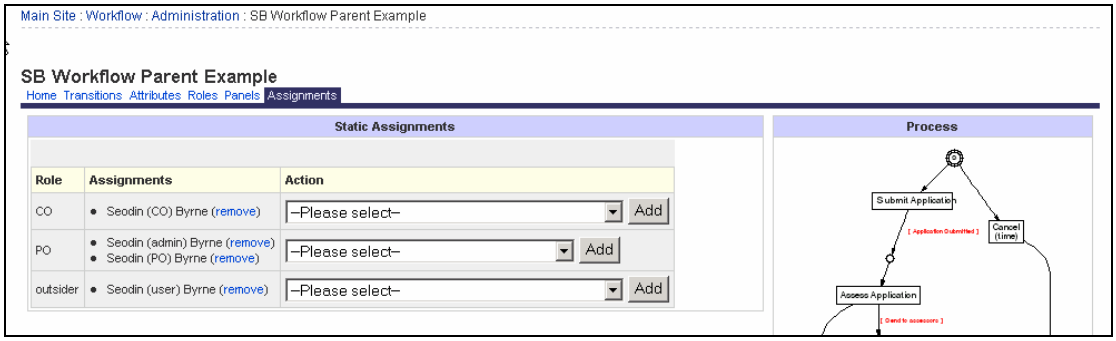
A panel may be edited at any time, both the header and the URL may be changed.

6.5.3 Delete a panel

Panels may be deleted at any time.

6.6 Assignments

Users are assigned to roles by adding them in the 'Assignments' section. Here we can see a list of all the roles setup for this workflow, all of the users assigned to each role and a drop-down allowing more users to be added to each.



6.6.1 Assign a user to a role

Select the user from the drop-down beside the role and click 'Add'

6.6.2 Remove a user from a role

In the assignments column, select 'remove' beside the user that you wish to remove from the role.

7 HOW TO CREATE A WORKFLOW

From Phase 1 in the prototyping phase the following details should have been established:

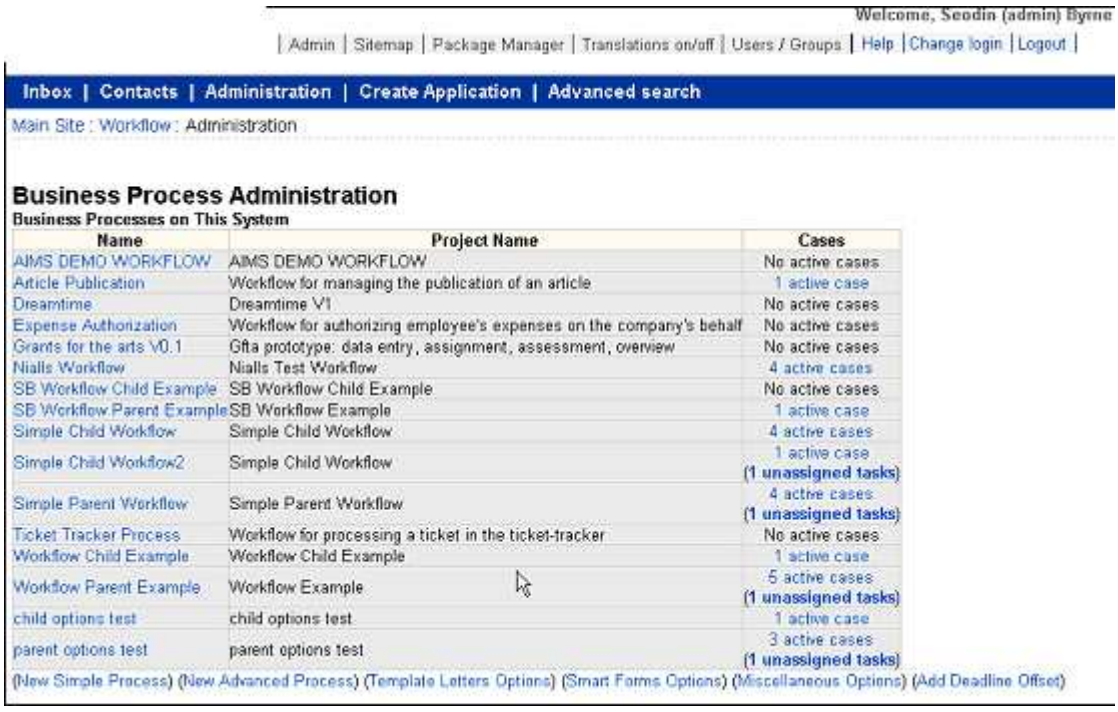
- All tasks(transitions) involved
- All links(arcs) between these transitions
- All the roles involved
- The roles that are associated with each transition
- The users in each role

Please see Prototype Example below.

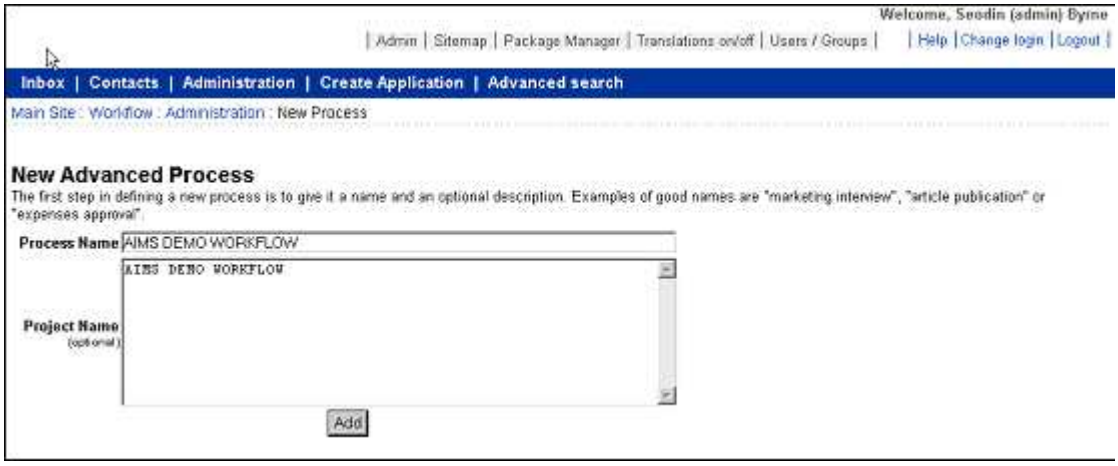
It is now possible to start building the workflow on the]PO[system.

7.1 Create the new workflow

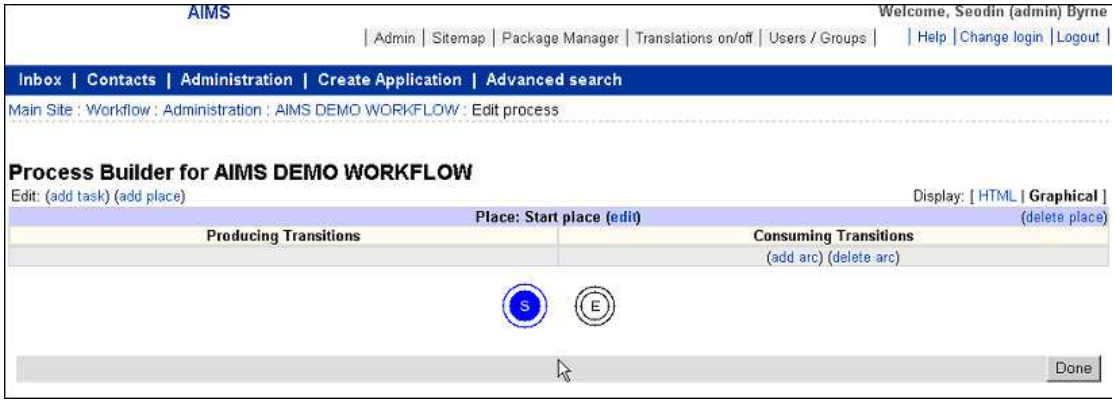
1. Log in to the]PO[system as an administrator and go to the Administration menu option. Select the Workflow link from the list.



2. Select 'New Advanced Process' and type in the name of the new Workflow and it's description.

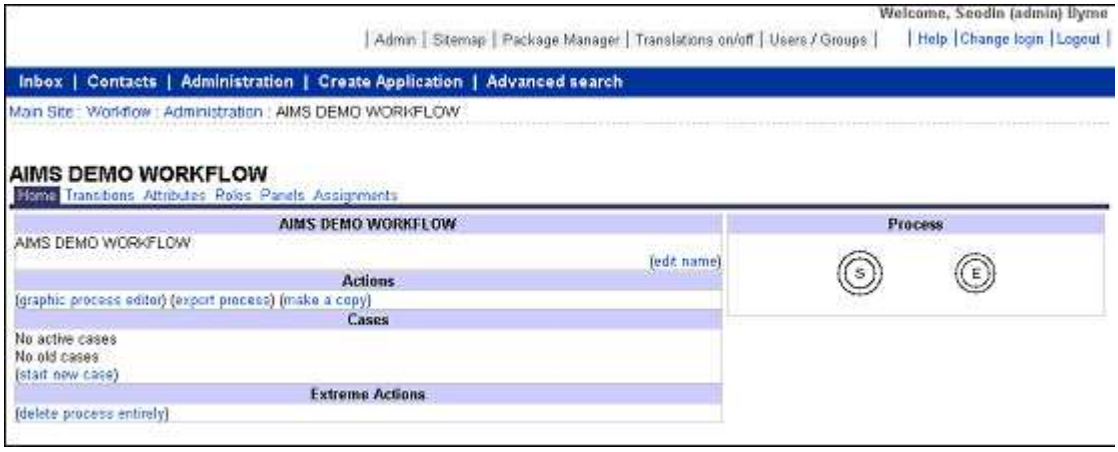


3. Press the Add button and the Process Builder for the new workflow is shown.



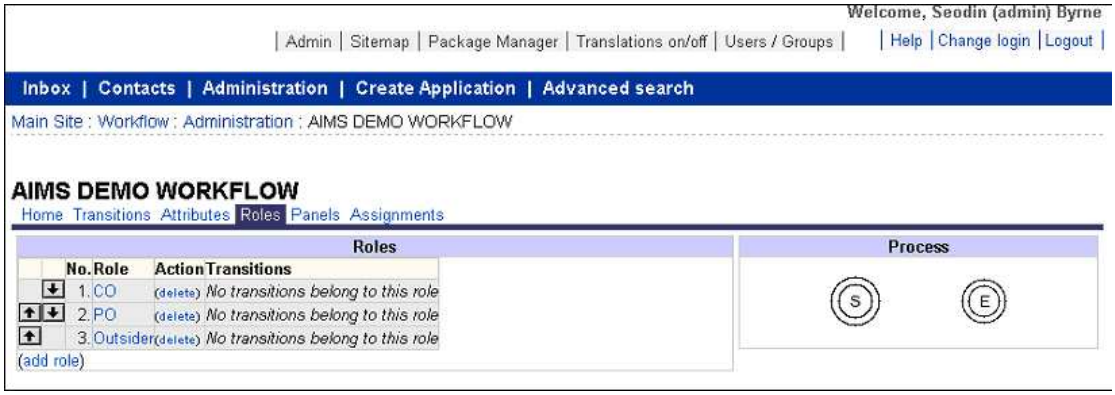
4. Go back to the Workflow Admin screen and select the new workflow from the list.

5. This will open the Home page for the workflow you wish to edit.



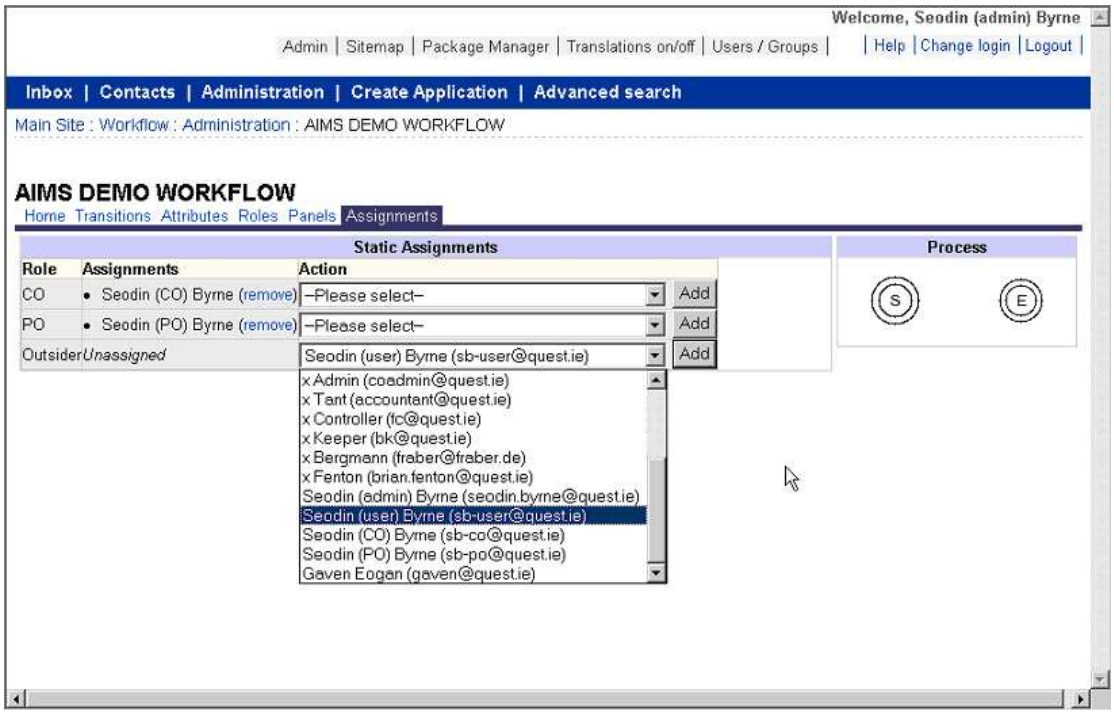
7.2 Create the Roles

Create an entry for every role on the prototype.



7.3 Assign users to Roles

- Select the 'Assignments' tab from the workflow admin screen
- Select the users to be added to the role from the drop-down in the 'Action' column and click 'Add'
- Repeat for each Role



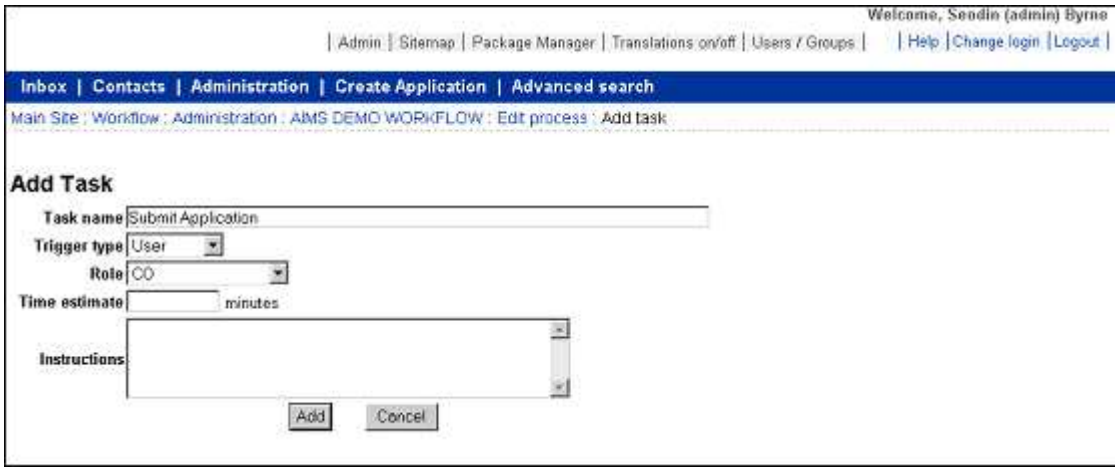
7.4 Create the Transitions

- Select the 'Transitions' tab from the workflow admin screen and click 'Add'

Enter in details for each transition.

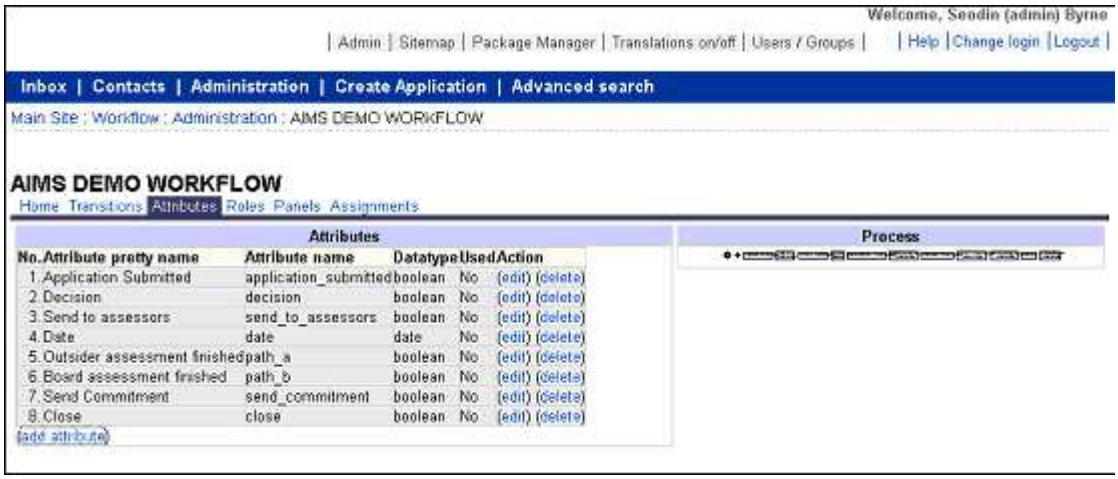
- Task Name: Name of Transition
- Trigger Type: Select 'User'
- Role: Select the role that is responsible for this transition
- Time Estimate: this indicates the estimated time required for the task.

- Instructions: Allows instructional text to be displayed on the user task screen.



7.5 Create the Attributes

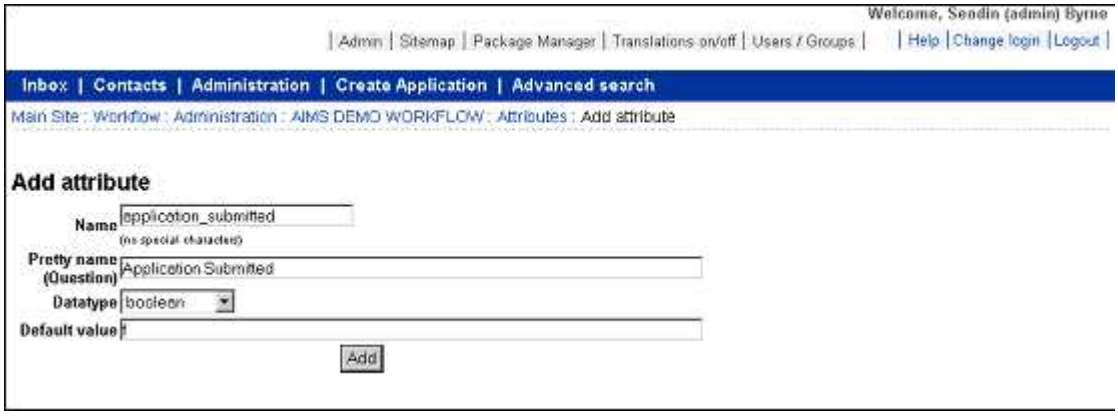
- Select the 'Attributes' tab from the workflow admin screen and click 'Add'



Enter the details and repeat for each attribute.

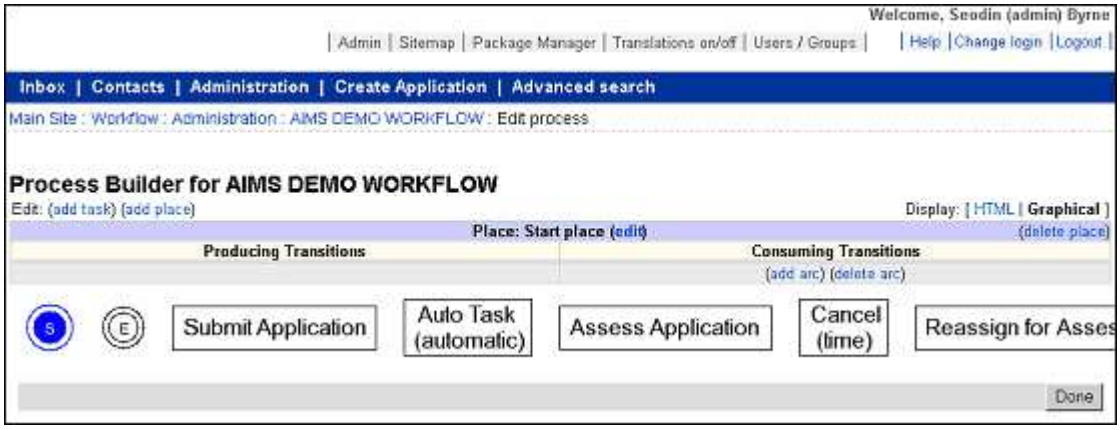
- Name: lower_case using underscores between words
- Pretty Name: The name that will appear on the button
- Data type: select 'Boolean'
- Default Value: enter 'f'. This will default the attribute to false.

o

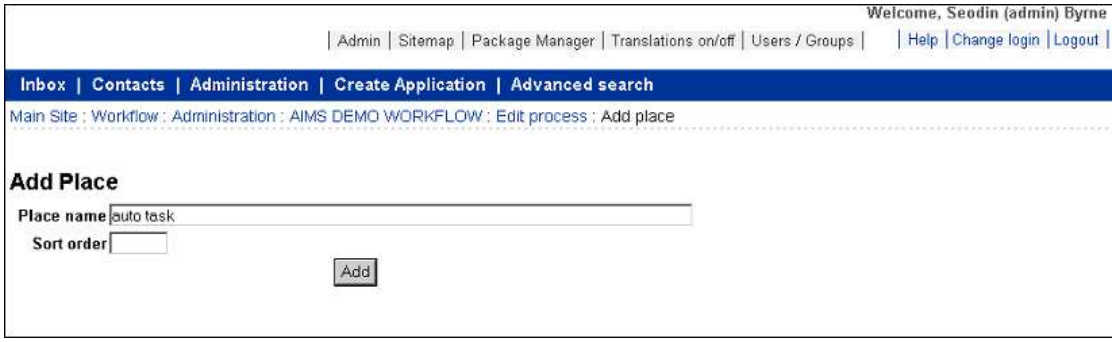


7.6 Create the Places

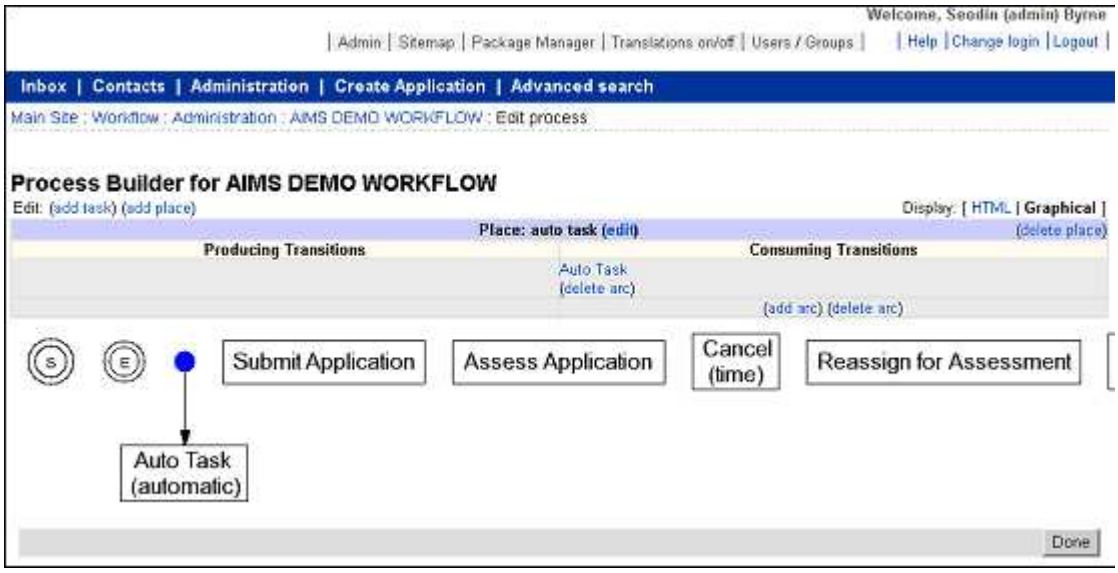
- Select the 'Home' tab and then 'Graphic Process Editor'



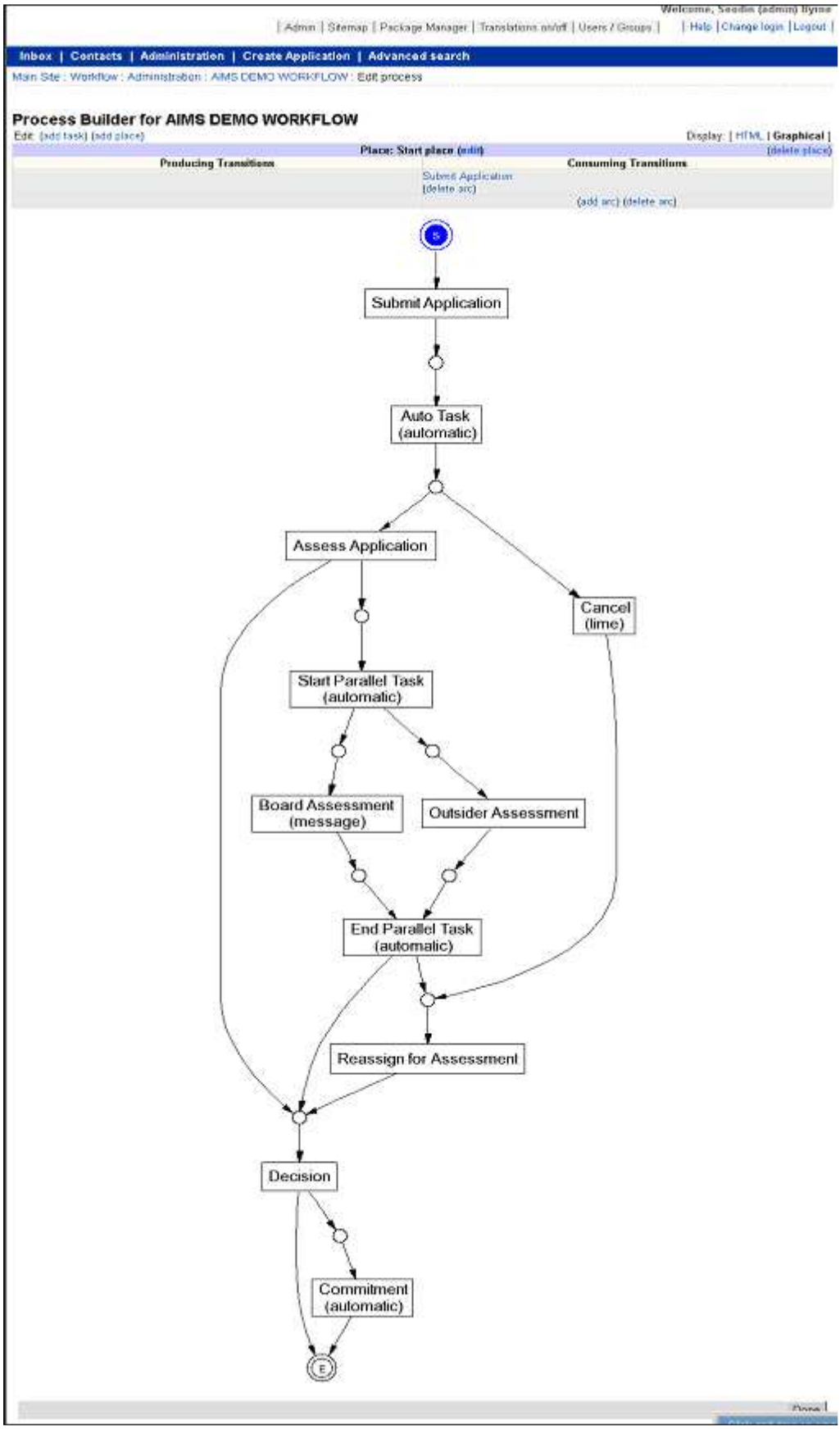
- Select 'Add Place'
- Add a place for each transition except for the first transition, which will use 'Start' as its 'place'.



- Connect the places to the transitions by clicking on the place, then on 'add arc' and then on the transition.

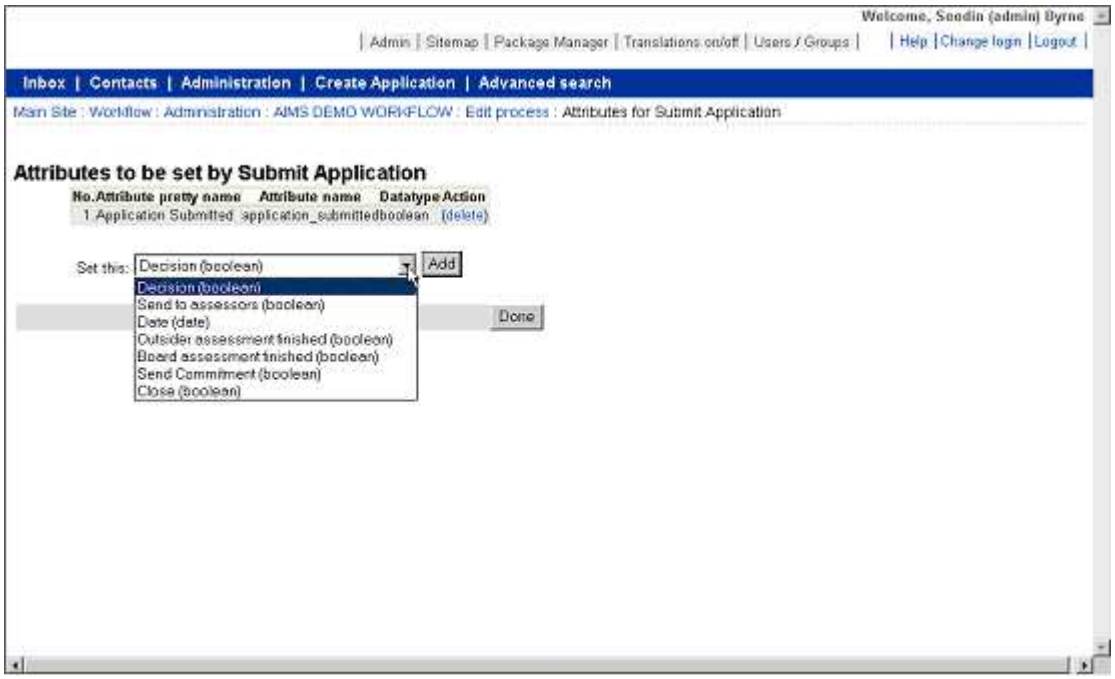

Welcome, Seodin (admin) Byrne
| Admin | Sitemap | Package Manager | Translations on/off | Users / Groups | | Help | Change login | Logout |
Inbox | Contacts | Administration | Create Application | Advanced search
Main Site : Workflow : Administration : AIMS DEMO WORKFLOW : Edit process
Process Builder for AIMS DEMO WORKFLOW
Edit: (add task) (add place)
Display: [HTML | Graphical]
Place: auto task (edit)
(delete place)
Producing Transitions
Consuming Transitions
Auto Task
(delete arc)
(add arc) (delete arc)
S
E
Submit Application
Assess Application
Cancel (time)
Reassign for Assessment
Auto Task (automatic)
Done

Connect the transitions following the workflow process that had been mapped out in the prototype, connect each transition to the appropriate place or places.



7.7 Add the Attributes to the Transitions

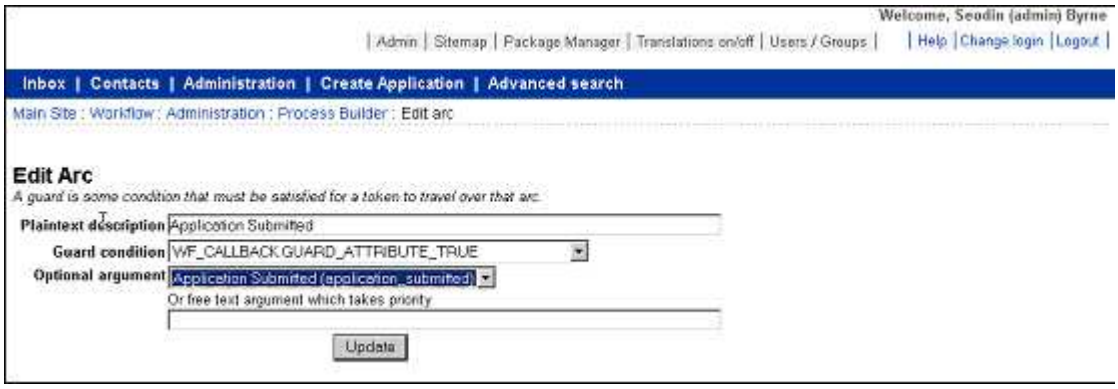
- Select the transition
- Select 'Attributes' at the top of the workflow diagram
- Select the relevant attribute from the 'Set this' drop-down and press 'Add'
- This is shown as a new button on the user Application Processing Screen.
- Click 'Done' to go back to the Graphic process editor



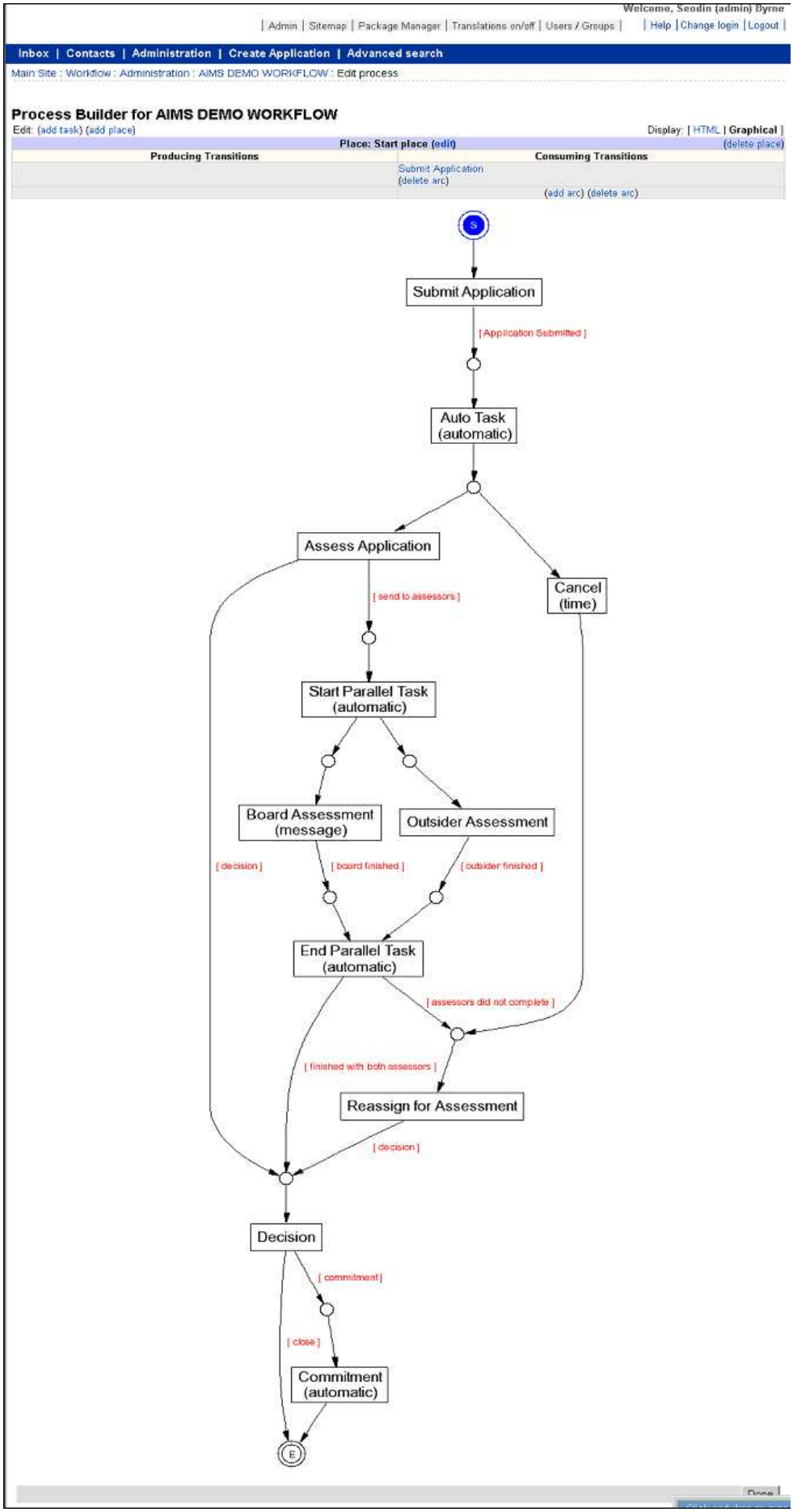
7.8 Add Guards to the Workflow

- From the process editor, select the transition and then select 'Add Guard' for the relevant place.
- Plaintext Description: Describes where the arc leads to
- Guard Condition: If a certain condition must be met before proceeding to the next transition in the workflow or a db record is to be created, the guard or 'callback' is added here.
- If a conditional guard is not required then use wf_callback.guard_attribute_true
- If a conditional guard is not required then use wf_callback.guard_attribute_true. This together with an attribute as the Optional Argument means that an application will only be allowed to proceed through this arc if the button associated with the attribute has been pressed by the user.
- Optional Argument: choose the attribute from the drop down which represents the button.
- Free Text Argument: any attribute name can be specified here. If an attribute name is specified along with wf_callback.guard_attribute_true, then the value of this attribute is evaluated within the callback and the application will move along the arc associated with

this guard if the attribute has the value TRUE for this application. Otherwise the application will not move along the workflow.



When the guard are added, they should appear as red text within square brackets to the side of the arc.

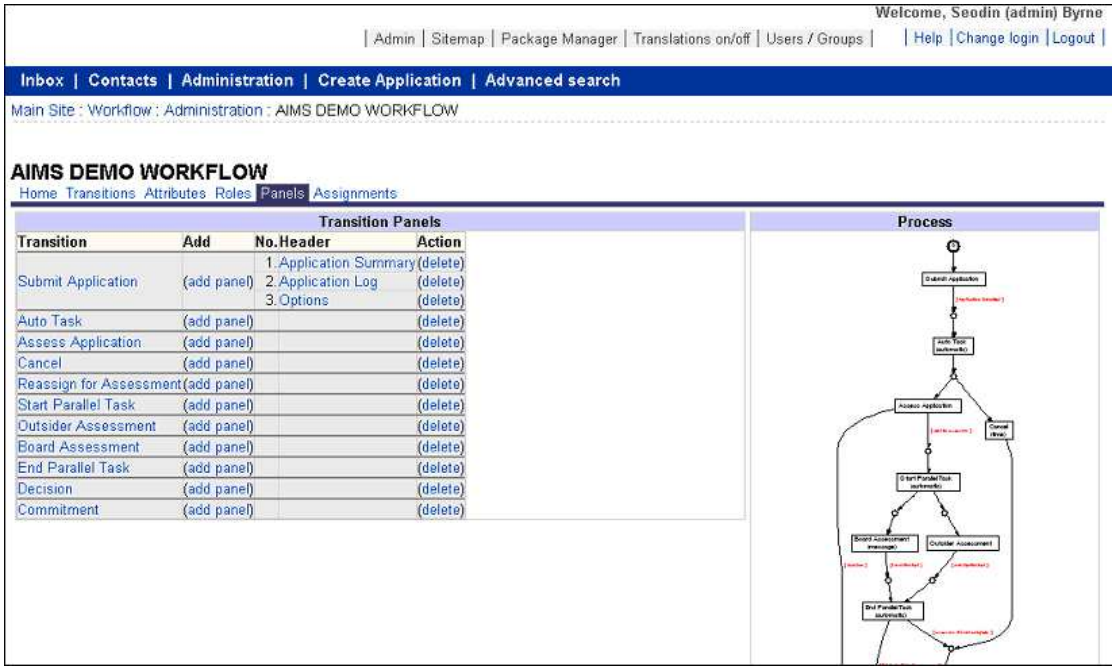

Welcome, Seodin (admin) Byrne
Admin | Sitemap | Package Manager | Translations on/off | Users / Groups | Help | Change login | Logout
Inbox | Contacts | Administration | Create Application | Advanced search
Main Site : Workflow : Administration : AIMS DEMO WORKFLOW : Edit process
Process Builder for AIMS DEMO WORKFLOW
Edit: (add task) (add place)
Display: | HTML | Graphical |
Place: Start place (edit)
(delete place)
Producing Transitions
Consuming Transitions
Submit Application
(delete arc)
(add arc) (delete arc)
S
Submit Application
[Application Submitted]
Auto Task (automatic)
Assess Application
[send to assessors]
Cancel (time)
Start Parallel Task (automatic)
Board Assessment (message)
Outsider Assessment
[decision]
[board finished]
[outsider finished]
End Parallel Task (automatic)
[assessors did not complete]
[finished with both assessors]
Reassign for Assessment
[decision]
Decision
[commitment]
[close]
Commitment (automatic)
E

7.8.1 Suggested Method for Changing an Existing Workflow to Remove a Stage through Which Applications Have Passed

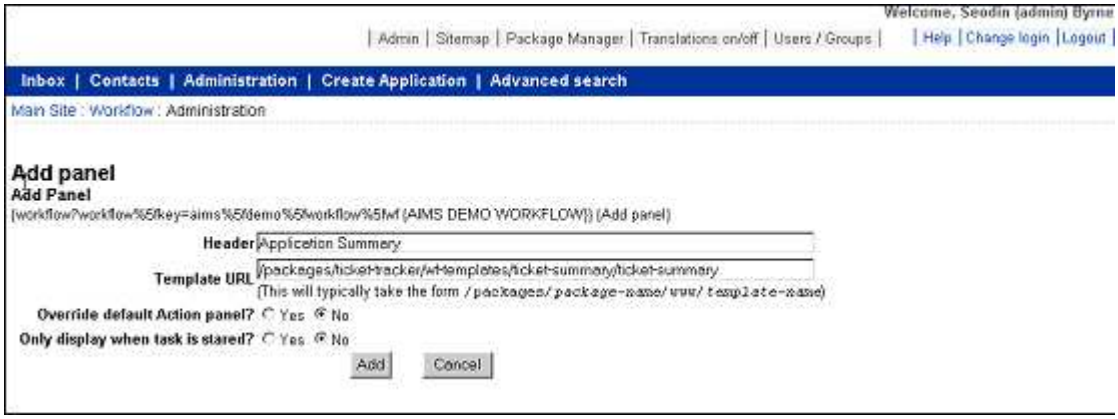
- Under the workflow administration page for this workflow, create an attribute with name always_false and default value 'f'.
- Go to each stage which has an arc leading to the stage which is to be "removed" from the workflow.
- Choose attributes for the stage, and delete the attribute which represents the button leading to the stage which is to be removed.
- Select the arc leading to the stage to be removed and edit the appropriate guard.
- Set the Guard Condition to wf_callback.guard_attribut_true
- Set the Optional Argument dropdown to None
- Set the free text argument to always_false
- As long as this attribute is not used as a button at any stage, the application will never travel down this route.

7.9 Add Panels to the Processing Screen

- Select the 'Panels' tab from the workflow admin screen



- Beside each transition in turn click 'add panel' to add the required panel.
- Header: Application Summary
- Template URL: /packages/ticket-tracker/wf-templates/ticket-summary/ticket-summary
- Header: Application Log
- Template URL: /packages/ticket-tracker/wf-templates/comments/comment-view
- Header: Options
- Template URL: /packages/ticket-tracker/wf-templates/options/user_options


Welcome, Seodin (admin) Byrne
| Admin | Sitemap | Package Manager | Translations on/off | Users / Groups | | Help | Change login | Logout |
Inbox | Contacts | Administration | Create Application | Advanced search
Main Site : Workflow : Administration
Add panel
Add Panel
(workflow?workflow%5fkey=aims%5fdemo%5fworkflow%5fwf (AIMS DEMO WORKFLOW)) (Add panel)
Header
Application Summary
Template URL
/packages/ticket-tracker/www/templates/ticket-summary/ticket-summary
(This will typically take the form /packages/package-name/www/template-name)
Override default Action panel? Yes No
Only display when task is stared? Yes No
Add
Cancel

8 HOW TO RUN A BACKGROUND PROCEDURE BEFORE A TASK IS STARTED

When a task is fired and the guards passed, the following transition becomes enabled. In some cases an extra procedure may be required to be called before the user starts the next task. This extra procedure is declared in the Enable action of a transition and is executed when that transition becomes enabled.

An example of when an Enable action occurs can be seen when a Parent workflow calls a child workflow. When the parent workflow transition is enabled, the procedure called from the Enable action causes the case to be brought directly to the child workflow. Please see Hierarchical Workflows - How to attach a Child Workflow for an example.

8.1 Enable Action

The Enable Action can be accessed by going to the Graphic Process Editor, selecting the transition that the enable action should be set against and then selecting the Actions link.

In the Enable Action Type, a callback must be entered in the PL/SQL proc field. These procedures are generally custom built with the exception of the Parent/Child workflow example above which is included in the]PO[package.

The Custom argument field might contain a parameter that can be passed into the procedure.

The Enable Action values are saved to the following database fields:

WF_CONTEXT_TRANSITION_INFO.ENABLE_CALLBACK
WF_CONTEXT_TRANSITION_INFO. ENABLE _CUSTOM_ARG

9 HOW TO RUN A BACKGROUND PROCEDURE WHEN A TASK IS COMPLETE

In some cases an extra procedure may be required to be called when a transition is fired. This extra procedure is declared in the Fire action of a transition and executes when a task is complete.

An example of when to use a Fire action could be when an application has been approved and a commitment needs to be sent to Finance. This transfer of data could be triggered when the button for 'Approve Application ' was selected.

9.1 Fire Action

The Fire Action can be accessed by going to the Graphic Process Editor, selecting the transition that the fire action should be set against and then selecting the Actions link.

In the Fire Action Type, a callback must be entered in the PL/SQL proc field. These procedures are generally custom built.

The Custom argument field might contain a parameter that can be passed into the procedure.

The Enable Action values are saved to the following database fields:

WF_CONTEXT_TRANSITION_INFO.FIRE_CALLBACK
WF_CONTEXT_TRANSITION_INFO.FIRE_CUSTOM_ARG

10 HOW TO MOVE A TASK ALONG IN THE PROCESS IF A CERTAIN TIME HAS ELAPSED

It is possible to set a task to run after a certain time has passed since the previous transition fired. Such a task might include a task to cancel an application if the application wasn't submitted in a certain amount of time. This can be achieved using a transition of trigger type Time. The Timed transition will only trigger when it's enabled. Therefore if a timed transition is to occur if a task hasn't completed on time, it will only fire if that task hasn't been started at all, i.e. the task is also still enabled and the token is in the place that joins both the task and the timed transition.

When a Time transition is added to the workflow, the time must be set using the Time Action on that transition.

10.1 Time Action

The Time Action can be accessed by going to the Graphic Process Editor, selecting the timed transition that the time should be set against and then selecting the Actions link. This trigger is fired when the transition becomes enabled.

In the Time Action Type, a callback must be entered PL/SQL proc field. A standard callback for calculating the time is provided in the procedure WF_CALLBACK.TIME_SYSDATE_PLUS_X.

The Custom argument field should contain the number of days the time should be calculated from. This number is passed into the callback when executed and the procedure returns the deadline as an Oracle date.

The Time Action values are saved to the following database fields:

WF_CONTEXT_TRANSITION_INFO.TIME_CALLBACK
WF_CONTEXT_TRANSITION_INFO.TIME_CUSTOM_ARG

11 HOW TO SET DEADLINES AGAINST A TASK

Tasks may have deadlines and these may be set in a number of ways. The deadline date can be seen in the users Inbox and also in the Application Processing screen.

11.1 Deadline offset

From the Business Process Administration screen, select the Add Deadline Offset link at the bottom of the page. Scroll down to the relevant workflow and select the transition that the deadline should be set against. Enter the number of days and press Submit.
When the workflow is in action, as soon as the deadline task (transition) has become enabled, the deadline is then calculated from the current time plus the number of days entered into the offset field.

The deadline offset value is saved to the database into
WF_TRANSITIONS.DEADLINE_OFFSET.

11.2 Deadline Action – automatic setting

The Deadline Action can be accessed by going to the Graphic Process Editor, selecting the transition that the deadline should be set against and then selecting the Actions link. This trigger is fired when the transition becomes enabled.

In the Deadline Action Type, a callback must be entered PL/SQL proc field. A standard callback for calculating the deadline is provided in the procedure
WF_CALLBACK.TIME_SYSDATE_PLUS_X.

The Custom argument field should contain the number of days the deadline should be calculated from. This number is passed into the callback when executed and the procedure returns the deadline as an Oracle date.

The Deadline Action values are saved to the following database fields:
WF_CONTEXT_TRANSITION_INFO.DEADLINE_CALLBACK
WF_CONTEXT_TRANSITION_INFO.DEADLINE_CUSTOM_ARG

This entry overrides the Deadline offset and the Deadline Action – attribute setting.

11.3 Deadline Action – attribute setting

The Deadline Action – attribute setting is used in a similar way to the automatic setting. Instead of entering the callback and a number of days, an attribute name is entered into the Attribute name field. This attribute name should correspond to an attribute of date type previously added.

If the deadline should be entered by the user, then the attribute should be assigned to a previous transition to the one the deadline is being set against. A field will appear on the Application Processing screen that the user may enter the deadline date.
N.B. the format of the date should be one of either :
YYYY/MM/DD

Or YYYYMMDD

Alternatively, if the user is not required to manually enter a date, the attribute may have a default value and so the attribute is not assigned to any transition.

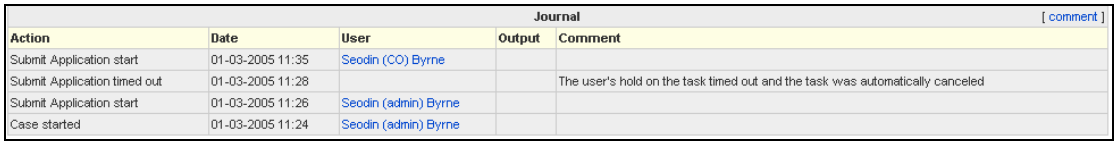
If the attribute value changes, the deadline of the enabled transition does not change, but if the transition becomes enabled again, the new value will be used.

The Deadline Attribute Action is saved to the database to
WF_CONTEXT_TRANSITION_INFO.DEADLINE_ATTRIBUTE_NAME

This entry overrides the Deadline offset but not the Deadline Action – automatic setting.

12 HOW TO REMOVE A TASK FROM A USER'S INBOX IF A CERTAIN TIME HAS ELAPSED

When a user starts a task, a lock is created for that transition so no other users can access it (unless they are assigned to it). By setting a holding timeout date, the system will allow the user to hold onto their task for a particular time. This timeout date can be seen in the Application Processing screen. When that date and time has arrived, the system will cancel that task and will bring the case task back to an enabled state and re-assign it to the person assigned for that task. A comment will be inserted into the journal so people know it was fired. The Timeout date is set using the Hold Timeout Action.



Journal [comment]

Action	Date	User	Output	Comment
Submit Application start	01-03-2005 11:35	Seodin (CO) Byrne		
Submit Application timed out	01-03-2005 11:28			The user's hold on the task timed out and the task was automatically canceled
Submit Application start	01-03-2005 11:26	Seodin (admin) Byrne		
Case started	01-03-2005 11:24	Seodin (admin) Byrne		

12.1 Hold Timeout Action

The Holding Timeout Action can be accessed by going to the Graphic Process Editor, selecting the transition that the timeout should be set against and then selecting the Actions link. This trigger is fired when the transition has started.

In the Holding Timeout Action Type, a callback must be entered in the PL/SQL proc field. A standard callback for calculating the timeout date is provided in the procedure

WF_CALLBACK.TIME_SYSDATE_PLUS_X.

The Custom argument field should contain the number of days the timeout should be calculated from. This number is passed into the callback when executed and the procedure returns the deadline as an Oracle date.

The Holding Timeout Action values are saved to the following database fields:

WF_CONTEXT_TRANSITION_INFO.HOLD_TIMEOUT_CALLBACK
WF_CONTEXT_TRANSITION_INFO. HOLD_TIMEOUT _CUSTOM_ARG

13 HOW TO RUN A BACKGROUND PROCEDURE WHEN A USER IS ASSIGNED TO A TASK

When a user is assigned a task, an email should be sent to that user informing them of their task. As standard, this will occur automatically by the workflow engine. If however a different email is required than the standard one, a PL/SQL procedure may be called to send a customized email. The]PO[system provides an alternative PL/SQL procedure however this is not very informative and will need to be changed per custom email.

In order to assign custom emails per task, a callback can be made in the Notification Action for the required transition.

13.1 Notification Action

Going to the Graphic Process Editor, selecting the transition that the Notification should be set against and then selecting the Actions link can access the Notification Action. This trigger is fired when the transition is enabled.

In the Notification Action Type, a callback must be entered in the PL/SQL proc field. A standard callback for calculating the timeout date is provided in the procedure

WF_ARTICLE_CALLBACK.NOTIFICATION

The Custom argument field is inputted by the callback, however it is not currently used.

The Notification Action values are saved to the following database fields:

WF_CONTEXT_TRANSITION_INFO.NOTIFICATION_CALLBACK
WF_CONTEXT_TRANSITION_INFO.NOTIFICATION _CUSTOM_ARG

14 HOW TO RUN A BACKGROUND PROCEDURE WHEN A TASK HAS BEEN UNASSIGNED

This is not currently used in the]PO[system, however a custom procedure may be written to carry out a particular function when a task is enabled and is not assigned to anyone. In order to run this custom procedure, an entry needs to be inserted into the Unassigned Task Action of a transition.

14.1 Unassigned task Action

Going to the Graphic Process Editor, selecting the transition that the Unassigned procedure should be set against and then selecting the Actions link can access the Unassigned Action. This trigger is fired when the transition is enabled.

In the Unassigned Action Type, a callback must be entered in the PL/SQL proc field.
The Custom argument field can hold a parameter to be inputted into the callback.

The Unassigned Action values are saved to the following database fields:

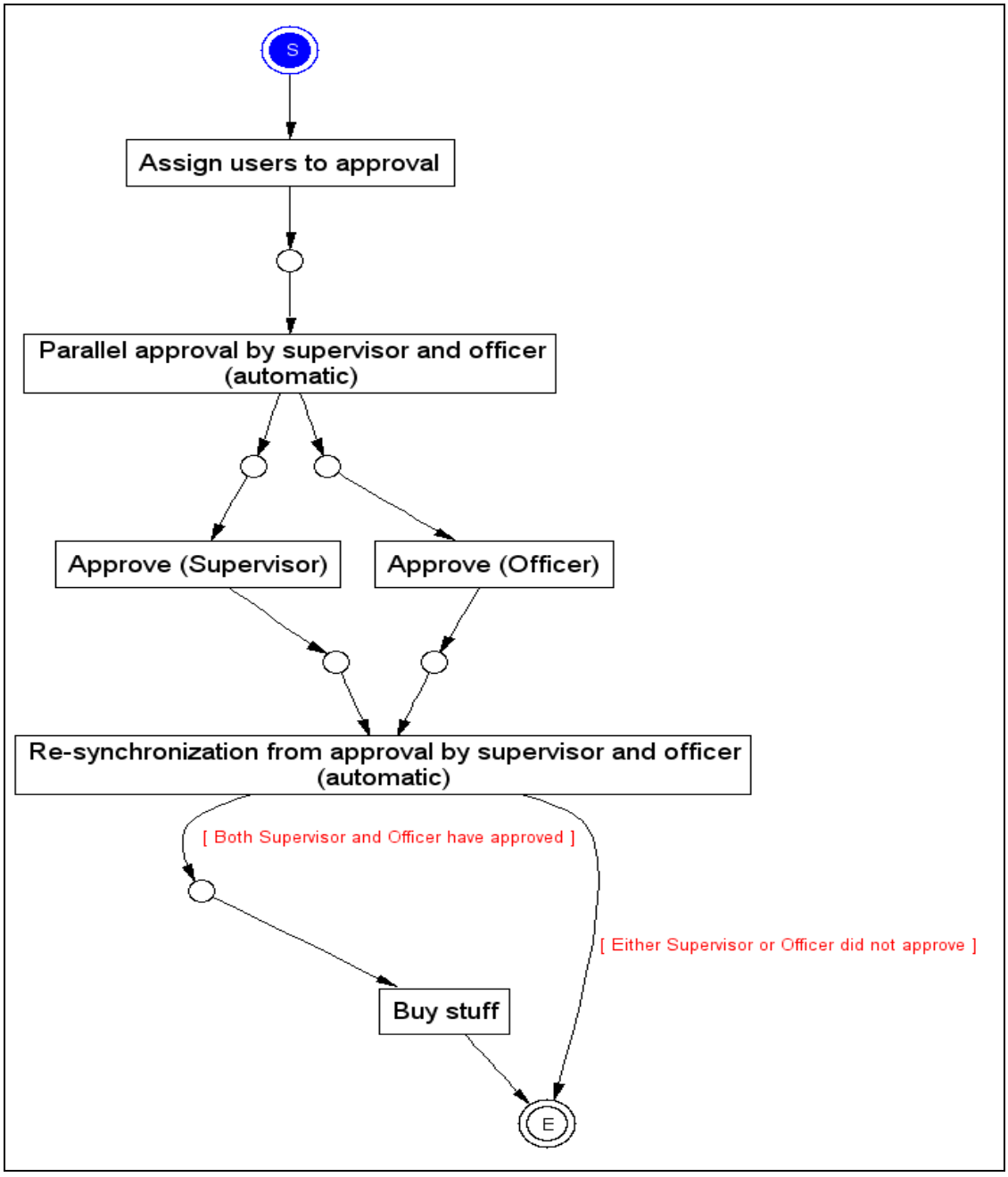
WF_CONTEXT_TRANSITION_INFO.UNASSIGNED_CALLBACK
WF_CONTEXT_TRANSITION_INFO.UNASSIGNED_CUSTOM_ARG

15 HOW TO SET UP A PARALLEL WORKFLOW

In some circumstances a workflow will be required to 'branch' so that 2 or more users may work on a case at the same time (in parallel). This involves sending the case to two or more roles at the same time. The case may then be processed by both roles concurrently before proceeding to the next stage in the workflow.

In the example below the case is sent to a Supervisor and also to an Officer. Both of these roles have the option to either 'Approve' or 'Reject' the case. If both approve then the case is sent on in the workflow to the 'Buy Stuff' transition, if only one or none approve then the case is sent to the End. This is determined with the use of guards and is explained in more detail below.

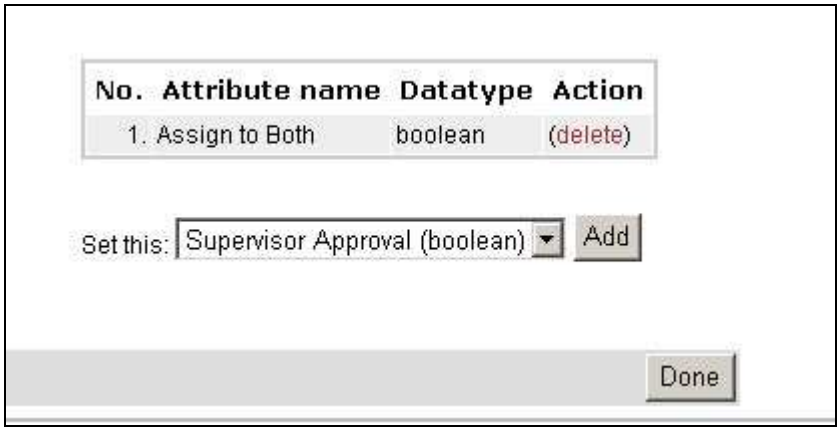
Figure 15-1 Example Workflow



15.1 Assign users to approval

At this transition the user selects the attribute 'Assign to Both' which then sends the case on to the 'Parallel approval by supervisor and officer' transition in order to create the branch.

Figure 15-2 Assign to Both



15.2 Parallel approval by supervisor and officer

When a case reaches this transition it is automatically sent to both the supervisor and the officer.

Figure 15-3 Task Parallel approval by supervisor and officer

15.3 Approve (Supervisor/Officer)

The case will appear in the inbox of both the Supervisor and the Officer. They will both have the option to either 'Approve' or 'Reject' the case. When they have each sent the case on it will arrive at the 'Re-synchronization from approval…' transition. The case will not proceed to the next transition until *both* Supervisor and Officer have sent it on. There are two attributes available at each of the two stages, making four attributes in total.

Figure 15-4 Attributes for Supervisor

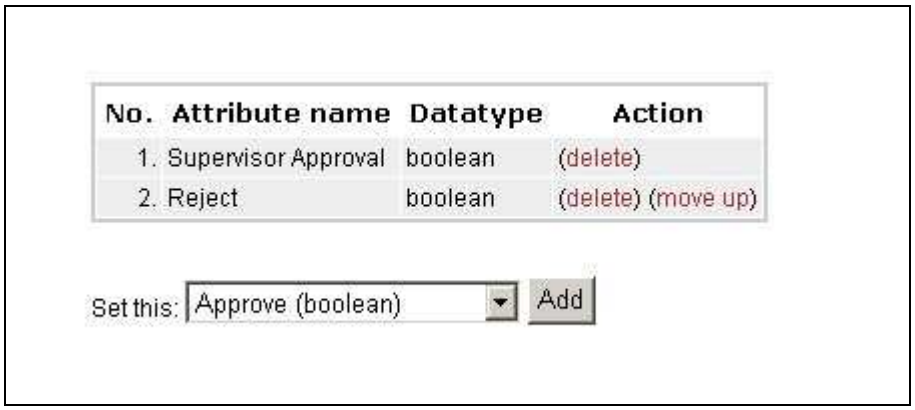


Figure 15-5 Attributes for Officer

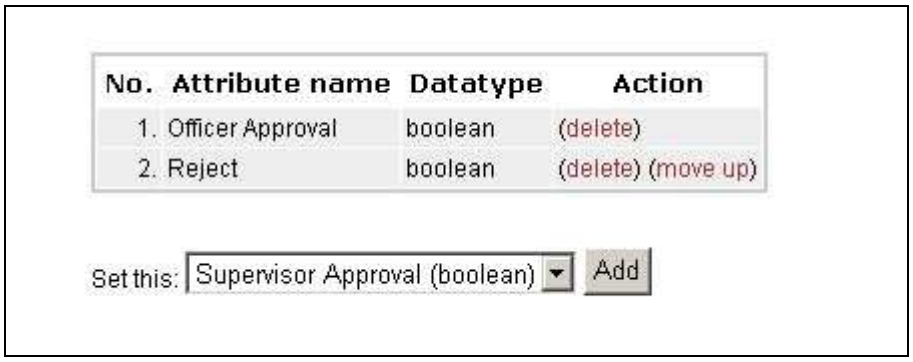


Figure 15-6 Creation of Attribute "Supervisor Approval"

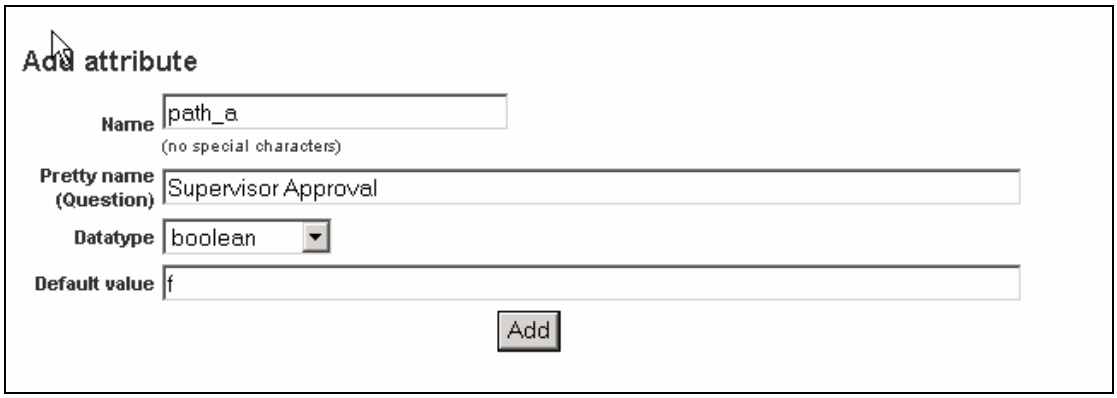
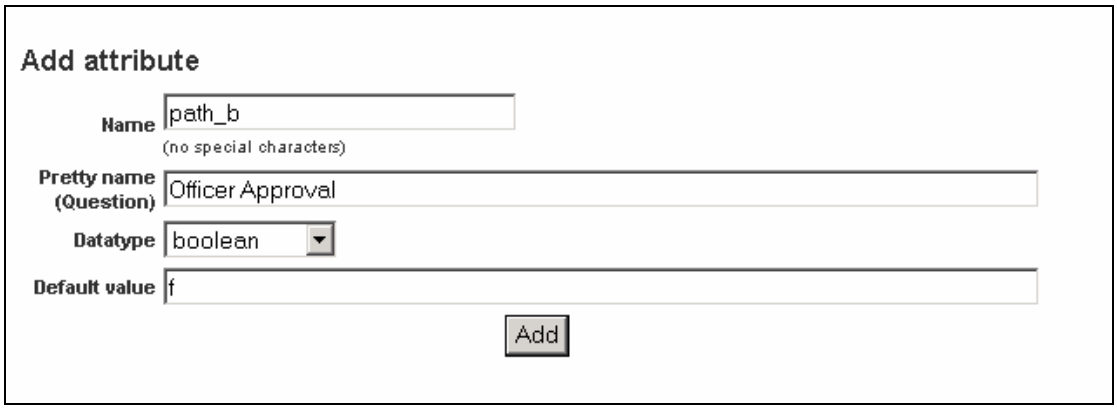


Figure 15-7 Creation of attribute "Officer Approval"



15.4 Re-synchronization from approval by Supervisor and Officer

At this transition the 'Branched' cases are merged into one again. A guard (guard_two_approved_p) on the arc leading to the 'Buy Stuff' transition determines whether both the Supervisor and Officer approved the case by checking the attributes that were set in each of the preceding transitions. If both have approved then the attributes path_a and path_b will be set to True. In this case, the guard will be satisfied and the case will proceed to 'Buy Stuff'. If only one or neither have approved then one of the attributes path_a/path_b will be false and therefore the guard will not be satisfied (i.e. returns false) and the case will proceed down the other arc (No other guards were satisfied).

Figure 15-8 Guard used to determine where to send the case next

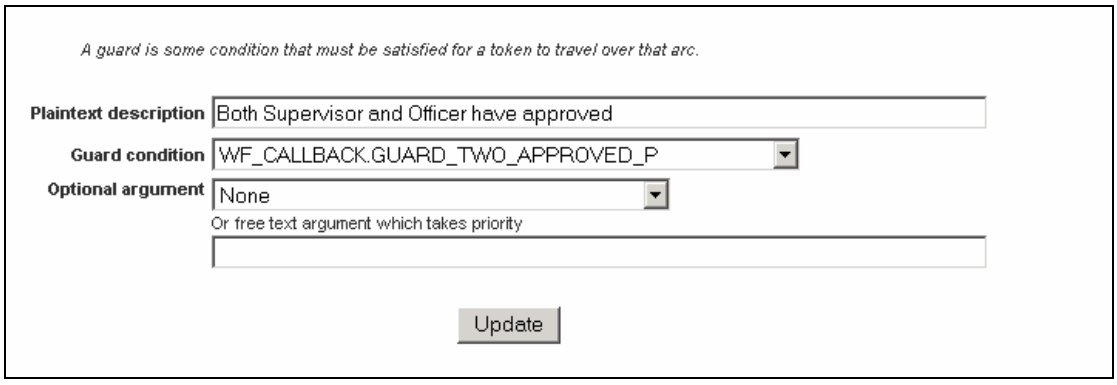
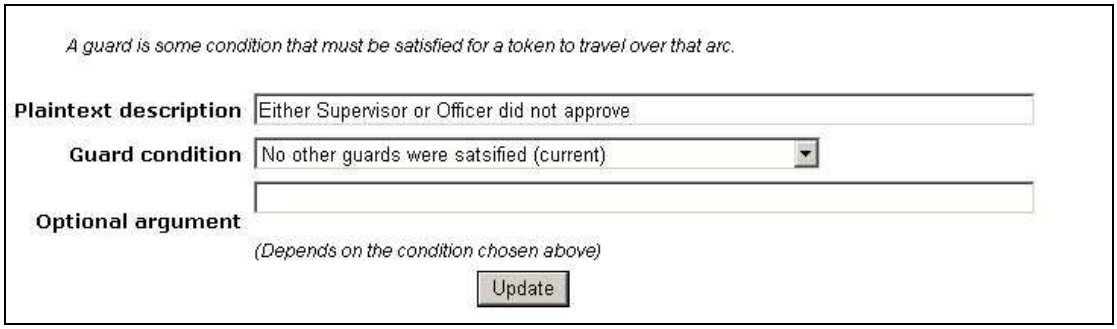


Figure 15-9 No other guards were satisfied



16 HIERARCHICAL WORKFLOWS - HOW TO ATTACH A CHILD WORKFLOW

'Hierarchical Workflows' introduces the notions of a 'Child Workflow' and a 'Parent Workflow'. Parent WFs can call a child WF by including a special type of transition that serves like a subroutine call. This transition defers execution to a child WF and only continues when the child WF has terminated.

16.1 Create a Child Workflow

A Child workflow must be created with a single input place and a single output place. Set up the Workflow as normal assigning attributes and roles etc.

16.1.1 Dynamic Role assignment

Role assignments in a child workflow may depend on the role/user assigned to a task in a parent workflow. E.G. A Payment Process might be represented by a child workflow and the payment might need to be authorised by the supervisor that approved the application. In order to obtain the parents assignment carry out one of the two following procedures:

To inherit the user assigned to the Parent WF:

1. Select the transition that will inherit the assignment
2. Go to the Actions section of that transition
3. In the Enable Action type, a callback must be entered in the PL/SQL proc field. A standard callback for obtaining a assignee is provided in the procedure WF_HIERARCHICAL.ASSIGN_PARENT_ASSIGNEE
4. The Custom argument field should contain the transition key of the parent transition that the child transition wishes to inherit.

To inherit the role assigned to the Parent WF:

1. Select the transition that will inherit the assignment
2. Go to the Actions section of that transition
3. In the Enable Action type, a callback must be entered in the PL/SQL proc field. A standard callback for obtaining a assignee is provided in the procedure WF_HIERARCHICAL.ASSIGN_PARENT_ROLE
4. The Custom argument field should contain the transition key of the parent transition that the child transition wishes to inherit.

16.2 Attach the Child Workflow to a transition in the Parent

To attach the Child WF to the Parent WF:

1. Create a message transition that replaces the procedure defined in the Child WF
2. Select this message transition and go to the Actions section
3. In the Enable Action type, a callback must be entered in the PL/SQL proc field. A standard callback for obtaining an assignee is provided in the procedure WF_HIERARCHICAL.SPAWN_CHILD
4. The Custom argument field should contain the workflow key of the child workflow to be called.

The workflow will execute the ‘spawn_child’ procedure when the parent workflow reaches the message transition. This procedure starts the child WF and tells the child about its parent. The child WF will fire the parent’s message transition once it reaches its ‘end place’.

16.2.1 Setting attributes on returning to the parent

Typically, you will have a default attribute value of true in the parent message transition. However, you can also use the ‘Fire’ action type in the same message transition to set the parent attribute to true on returning from the child, regardless of the outcome of the child or the default attribute value.

You do this with the pl/sql procedure ‘workflow_case.tr_callback_set_attr_true' and by specifying the relevant parent attribute name as the custom argument.

Note: if the default attribute value in the parent is false, and there is a normal wf_attribute_value_true guard checking for true, hung tokens and parallel merging issues may result if the above instructions are not followed.

17 HOW TO SET PERMISSIONS PER ROLE

When a new role is set up in a workflow, their permissions must also be established within the]PO[system. To complete this go to the Roles/Permissions Matrix section from the Administration module.

A list of all workflows and the roles within each are displayed along with check boxes for each task. Roles may be granted permission to carry out the various tasks within each workflow by selecting the relevant check boxes and pressing the 'Save Permissions' button.

Figure 17-1 Roles/Permissions matrix

NESTA : Application tracker : Administration : Role Permissions Matrix 06-09-2004

Workflow	Roles/Functions	Process Application	View Application	Create Application	Edit Application	Create Letters	Reassign Task	Create Payment	Edit Payment	Authorise Payment	Upload a File
Dreamtime V1	Applicant	☑	☑	☑	☐	☐	☐	☐	☐	☐	☐
Dreamtime V1	Programme Officer	☑	☑	☑	☑	☑	☑	☑	☑	☑	☑
Dreamtime V1	Assessor	☑	☑	☐	☐	☐	☐	☐	☐	☐	☐
Dreamtime V1	Mentor	☑	☑	☐	☐	☐	☐	☐	☐	☐	☐
Dreamtime V1	CEO	☑	☑	☐	☐	☐	☐	☐	☐	☐	☐
Dreamtime V1	Programme Director	☑	☑	☐	☐	☐	☐	☐	☐	☐	☐
Dreamtime V1	Finance	☐	☑	☐	☐	☐	☐	☐	☐	☐	☐
Dreamtime V1	Interviewer	☑	☑	☐	☐	☐	☐	☐	☐	☐	☐
Dreamtime V1	Committee	☑	☑	☐	☐	☐	☐	☐	☐	☐	☐
Dreamtime V1	Trustee	☑	☑	☐	☐	☐	☐	☐	☐	☐	☐
Dreamtime V1	Delegate	☑	☑	☐	☐	☐	☐	☐	☐	☐	☐
Dreamtime V1	Strategy & Communications	☐	☐	☐	☐	☐	☐	☐	☐	☐	☐
Fellowship V1	Nominator	☑	☑	☐	☐	☑	☑	☐	☐	☐	☐
Fellowship V1	Programme Officer	☑	☐	☑	☑	☑	☑	☑	☑	☑	☑
Fellowship V1	Nominee	☑	☑	☐	☐	☑	☑	☐	☐	☐	☐
Fellowship V1	Assessor	☑	☐	☐	☐	☐	☐	☐	☐	☐	☐

18 HOW TO ATTACH A SMARTFORM TO A WORKFLOW

Smartforms may be added to any stage in the workflow by doing the following:

- Go to /wf/admin
- At the bottom of the screen select 'Categorise Smartforms'
- Scroll down to the Workflow required
- Select the transition where you want the smartform to appear
- Select the smartform to categorise with the transition

Figure 18-1 Select transition

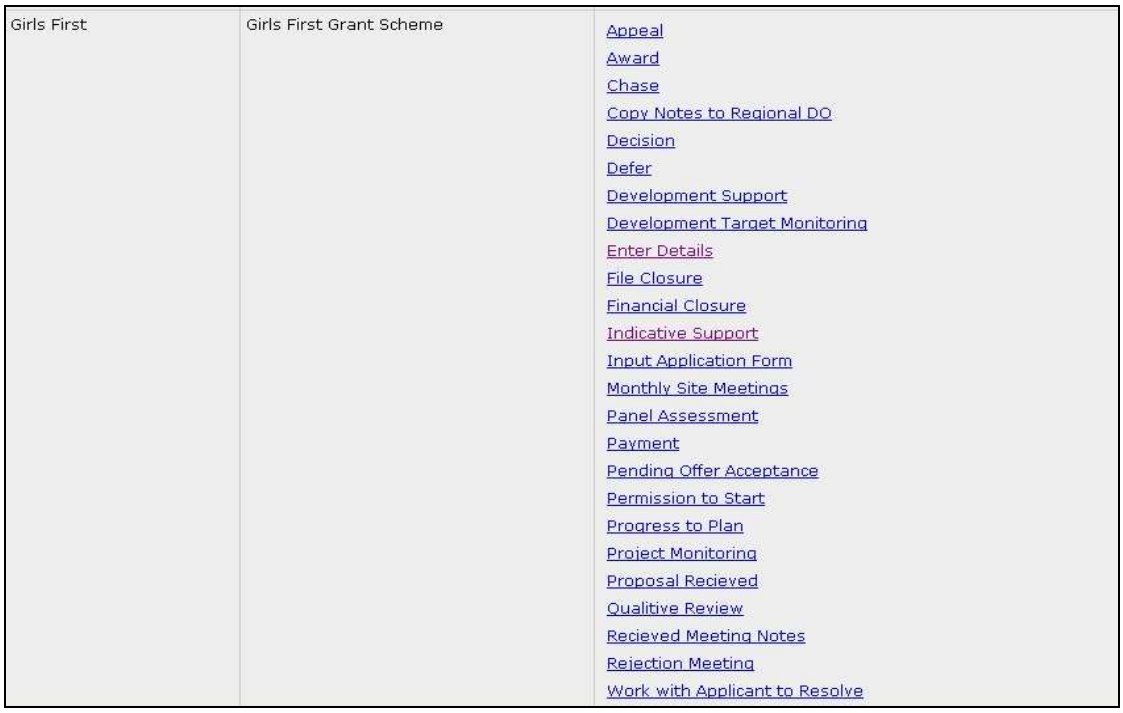
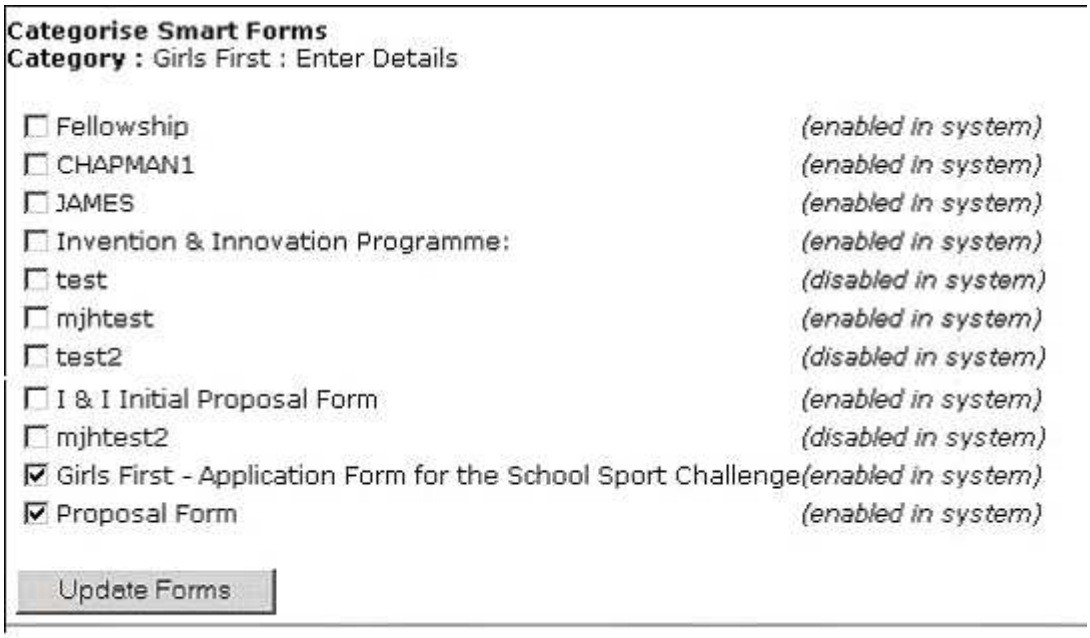


Figure 18-2 Select smartform to categorise with transition

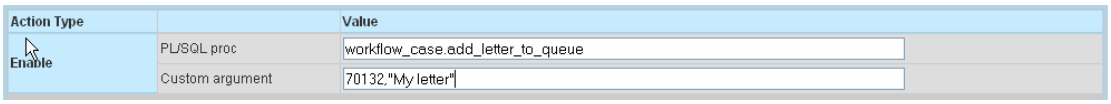


19 HOW TO ATTACH A TEMPLATE LETTER TO A WORKFLOW

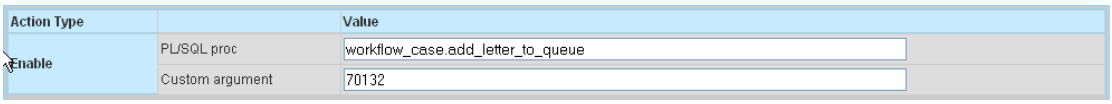
Template letters may be added to any transition in the workflow in the same way as Smartforms, the only difference is that instead of selecting 'Categorise Smartforms' on the workflow admin screen we select 'Categorise Template Letters'. See How to attach a Smartform to a Workflow above.

20 HOW TO QUEUE APPLICATION LETTERS FROM WITHIN A WORKFLOW

You can use a custom workflow procedure to queue letters when a task is enabled for a given transition. The procedure is called workflow_case.add_letter_to_queue. You specify this as the "PL/SQL Proc" for the required action type and specify a custom argument which is the letter id (i.e. template_letters.letter_id) and a reference for the queue entry. E.g.



NOTE: The next instruction is specific to]PO[meetings functionality.
Alternatively, in the case of meetings outcome workflow transitions, you can leave the reference field blank e.g.



In this case, the code will assume a meeting occurred in the previous transition. The code will work out which meeting it was, and create a reference string of "<meeting body> <office> <date/time>" for that meeting. This feature is designed such that you print all the letters for a given meeting as a batch by filtering the letter queue on that reference.

21 HOW TO DELETE A WORKFLOW

Before deleting a Workflow, it is important to disassociate the workflow from the Ticket Tracker (in admin/ticket tracker - make sure there's no scheme/category associated with the workflow).

All roles that appear in the Role Permission Matrix for the workflow must also be unchecked.

When this is complete go to Administration followed by Workflow. Select the correct workflow from the list and select the Delete all cases link and then the Delete Process entirely link.

Please note this procedure should be treated with great care as all cases will be entirely deleted from the]PO[system.

22 HOW TO IMPORT AN EXPORTED WORKFLOW

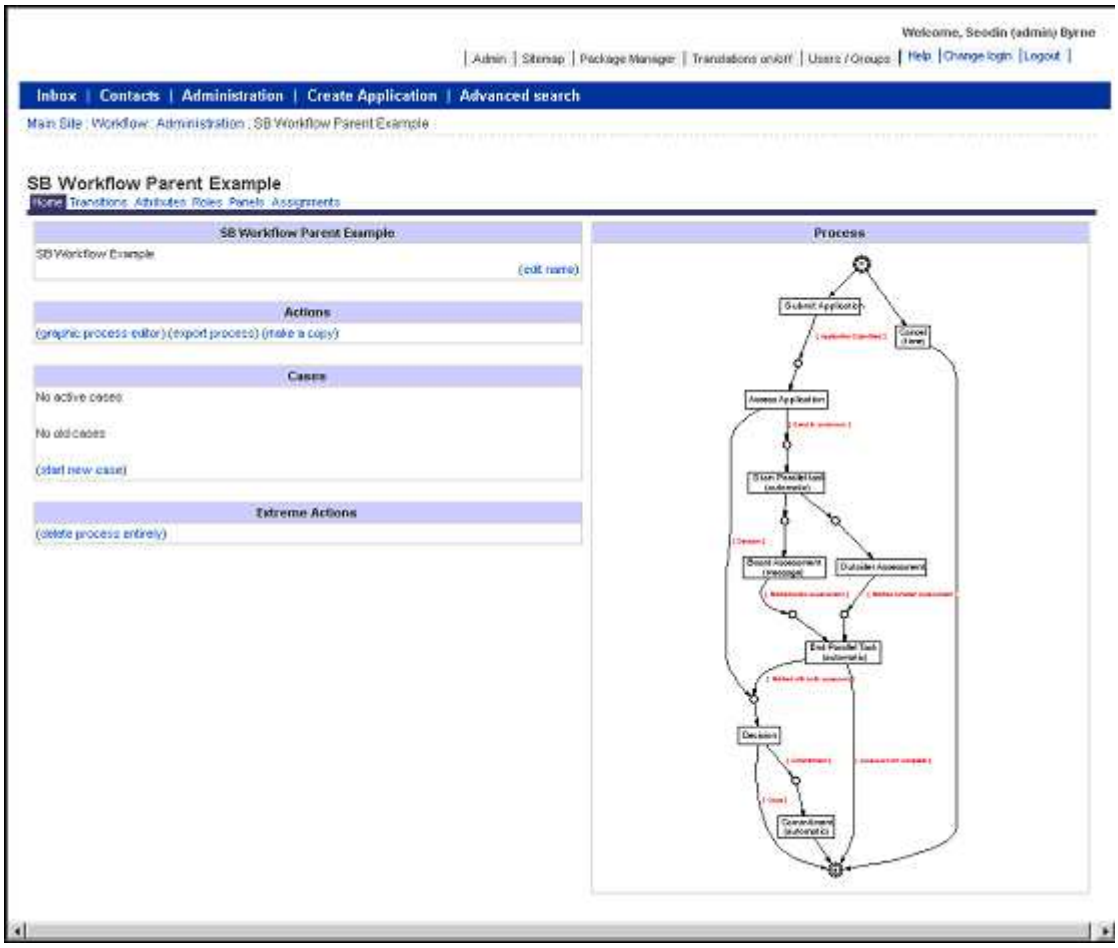
This task can only be carried out by an administrator who is a member of the technical team.

The admin must open a SQL session that is connected to the]PO[database. The workflow SQL procedure saved from the Export can then be executed. This will create the workflow on the system.

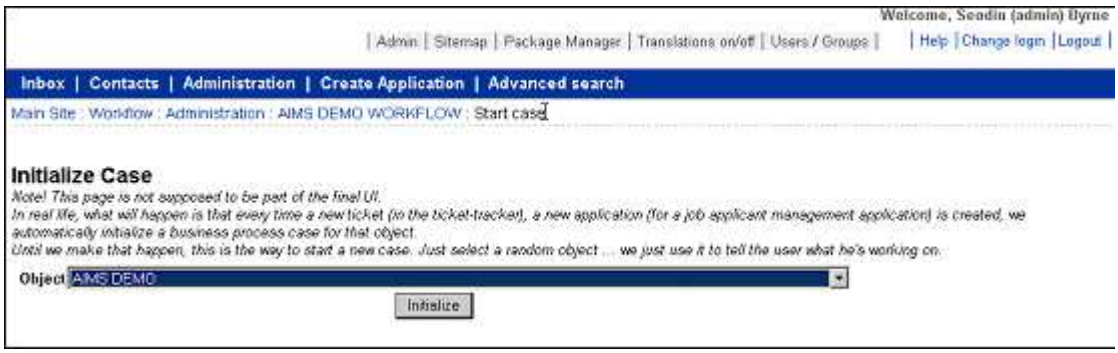
23 HOW TO DEBUG A WORKFLOW

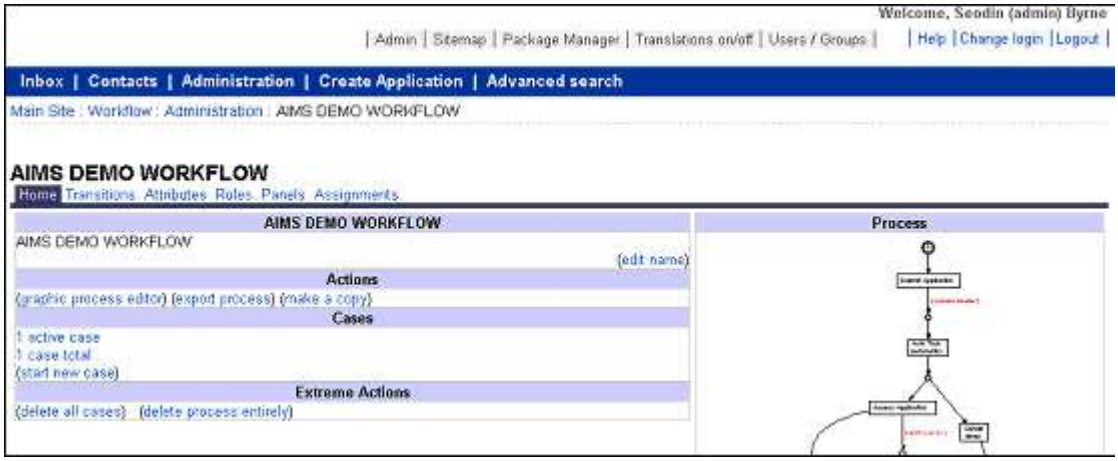
Note: When going through a newly created workflow make sure that no panels have been added. These will cause an error during this process. Once the workflow is routing correctly go back and add in the panels.

Select the Start new case from the Workflow admin home page.



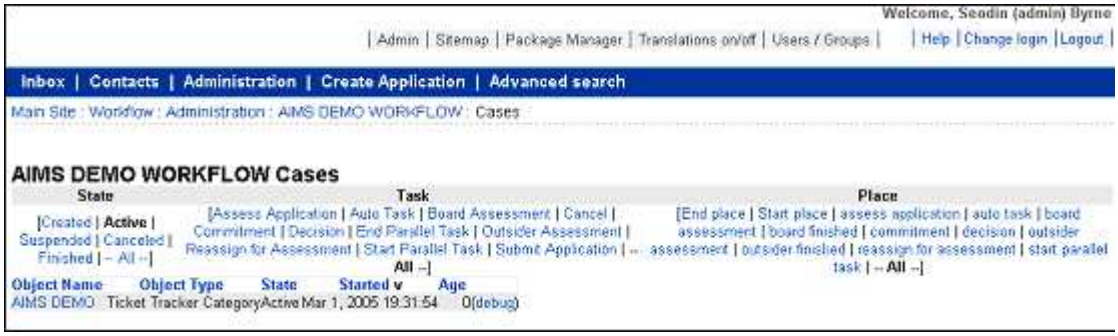
Select the ACS object - which in this case, will be the Category, set up for the new workflow and click Initialize.



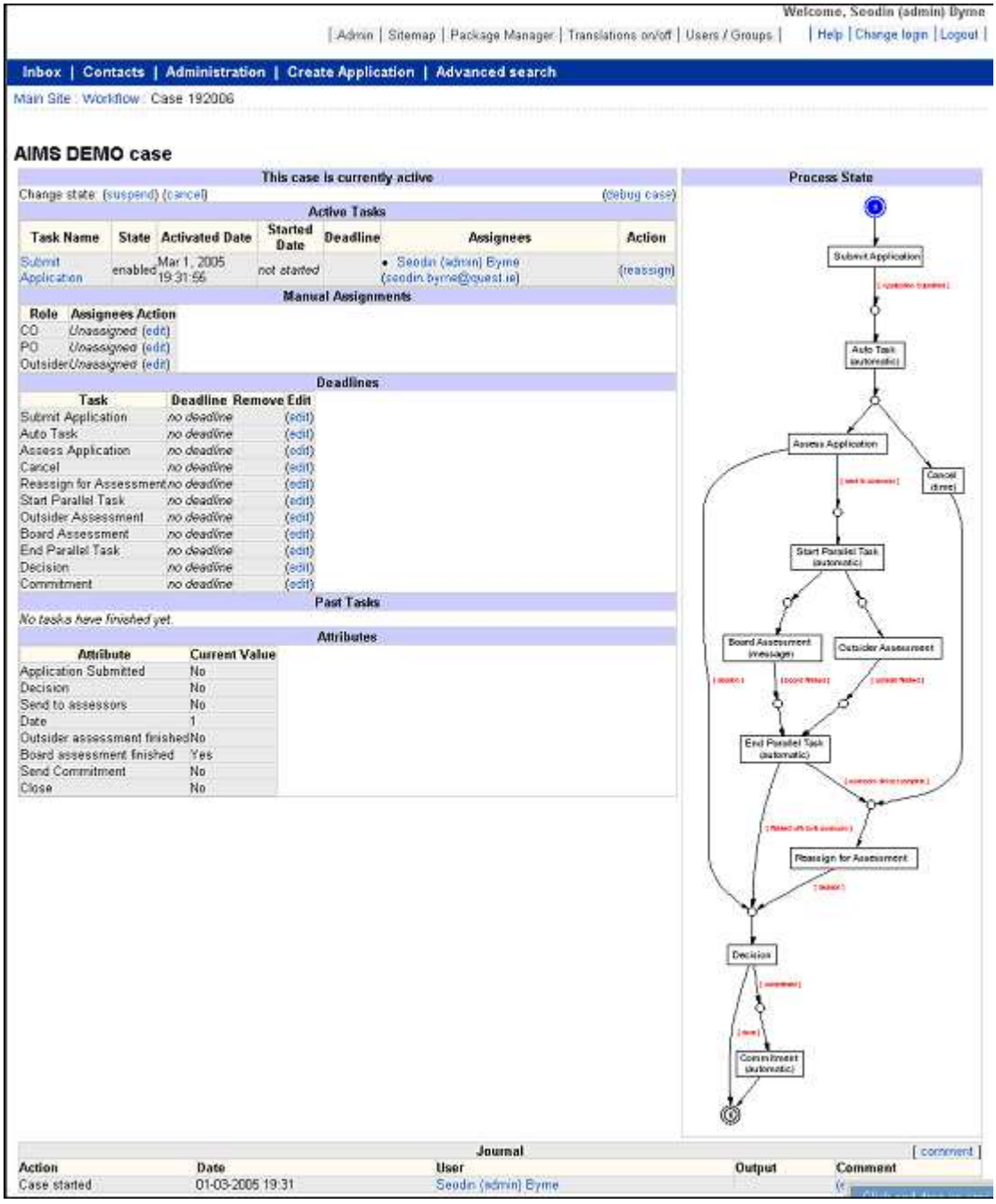


The Workflow admin home page will now appear with an active case.

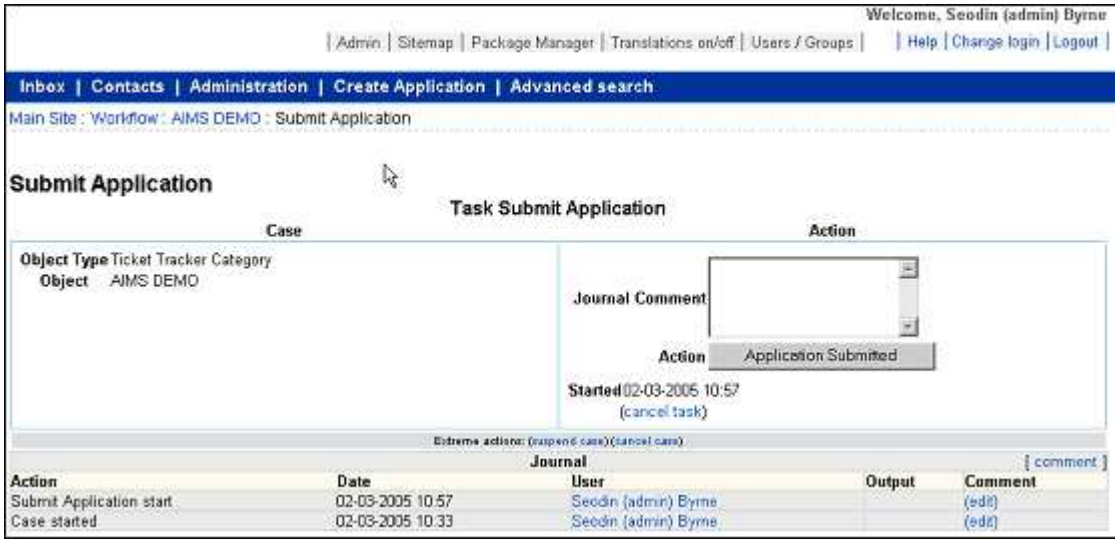
Select the active case link, a list of all active cases are shown. Cases of all statuses can be selected to view.



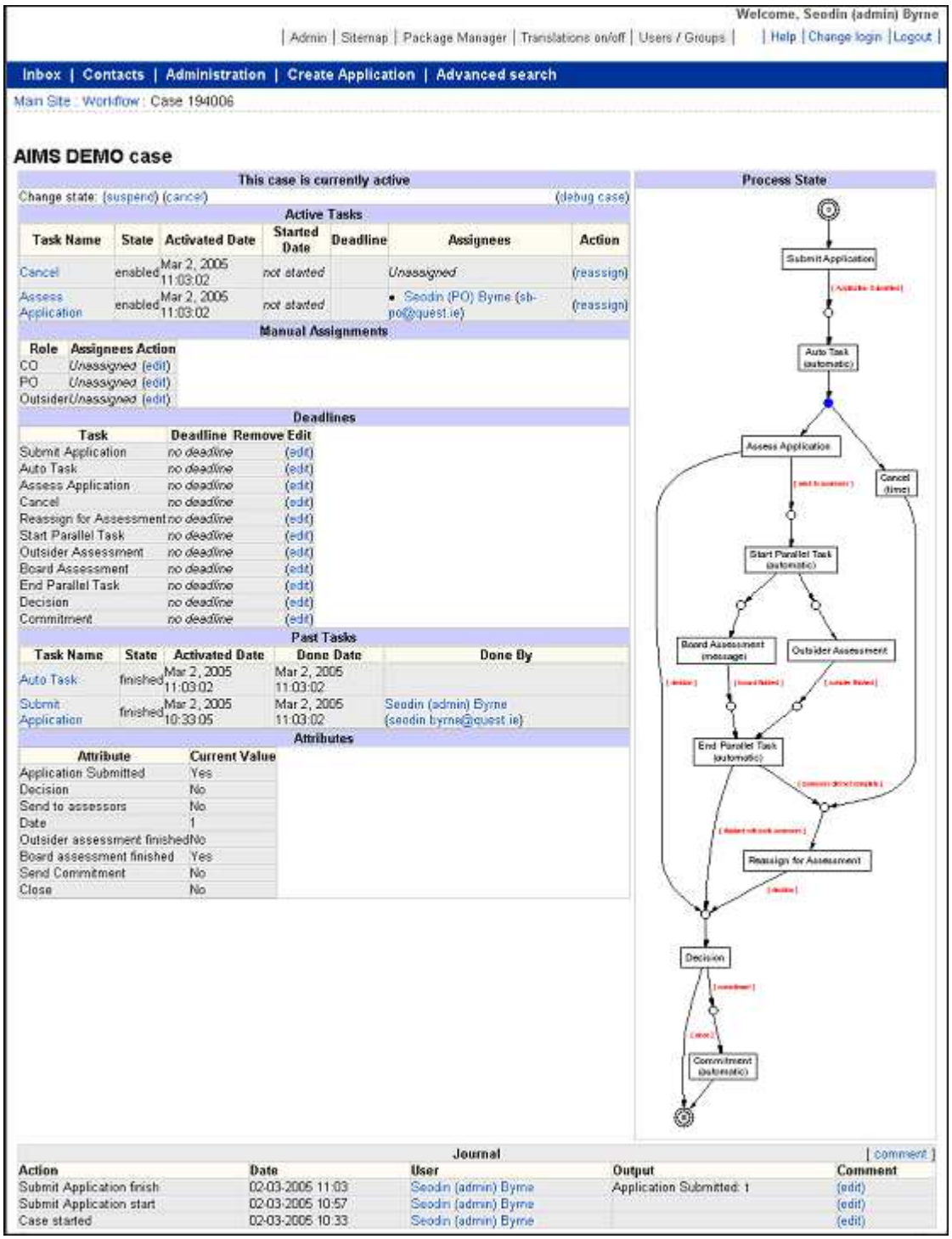
Select the relevant object name link and all details of that case can be viewed. In the process state, a diagram of the workflow is shown and either a place or a transition will be highlighted in blue depending on where in the process the case is. If a transition is enabled, the place will be highlighted, whereas if it has started, the transition is blue.



Select the task name link that you wish to start. The Application processing screen is shown and an attribute can be selected (if that transition was a user one).



Once the attribute is selected, the process moves on and the next transitions are enabled.



Continue this process until all paths have been tested and the Workflow is ready for use.

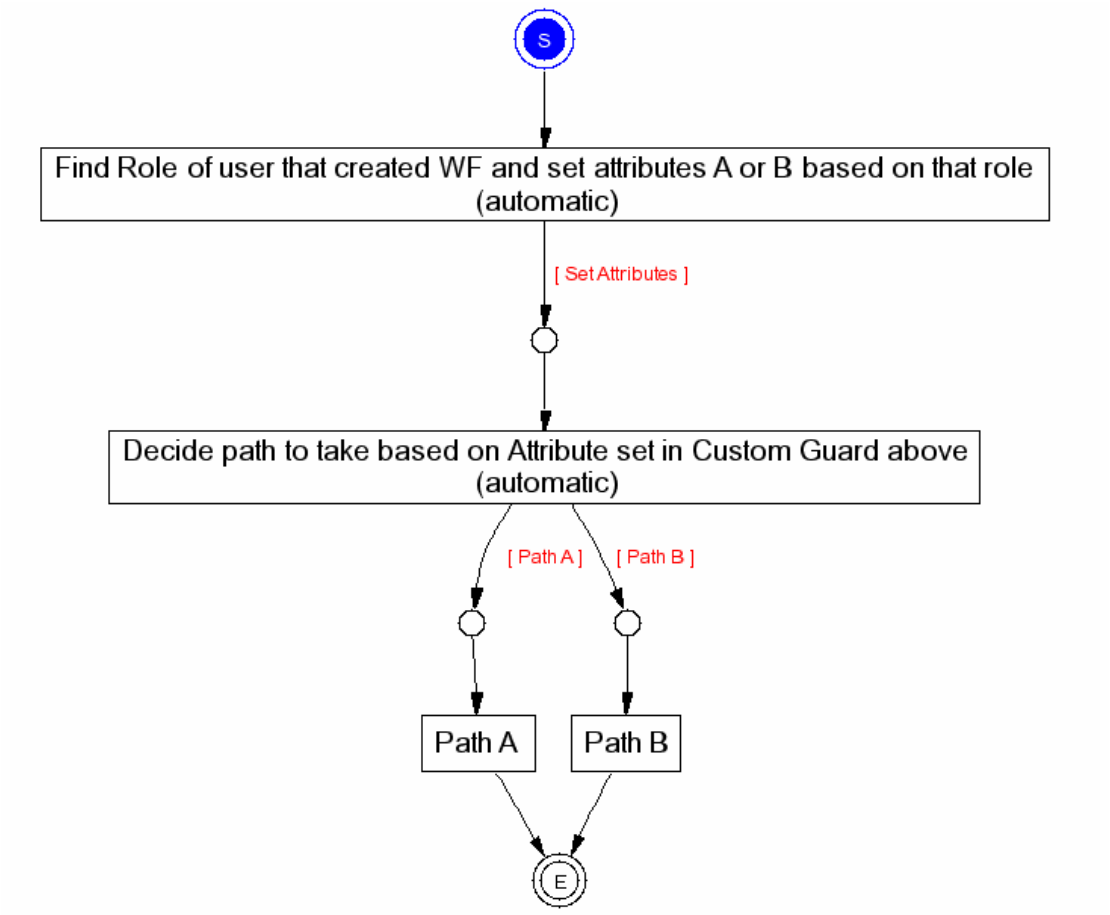
24 HOW TO DETERMINE THE FIRST USER TRANSITION DEPENDING ON THE ROLE OF THE USER

1. Set up WF similar to example below.

2. Need to set up Attributes 'Path A' and 'Path B' – default to False.

3. For transition 'Find Role …' Set it up as Automatic. Assign a guard with condition = custom guard.

4. Need to write a custom guard that will
   - Identify the user that started the WF
   - Find the user's Role
   - Set attribute 'Path A' or 'Path B' as True based on Role – see Proc WORKFLOW_CASE.SET_ATTRIBUTE_VALUE
   -
5. Assign attributes to transition to 'Decide path to ….. When the WF gets to this task, it will take whatever path the Guard satisfies i.e. whichever attribute is set to True.



25 PROTOTYPE EXAMPLE

This is a sample of a workflow prototype. It includes two workflows, a parent one and a child one which will be nested into the parent one.

It includes the following details:

25.1 Workflow Roles and Users Assigned

25.1.1 Child Workflow

Role	**Users**
Board CO	USER1@project-open.com

25.1.2 Parent Workflow

Role	**Users**
CO	user1@project-open.com
PO	user1@project-open.com
outsider	user2@project-open.com

25.2 Workflow Stages (roles, transitions and attributes assigned)

25.2.1 Child Workflow

Transition Name	Trigger Type	Role
Assessment by Board	User	Board CO
Board Decision	User	Board CO

Attributes:

Transition	Attribute name	Attribute Pretty name	Data type
Assessment by Board	decision_made	Decision Made	Logical
Board Decision	board_decision	Board Decision	Logical

25.2.2 Parent Workflow

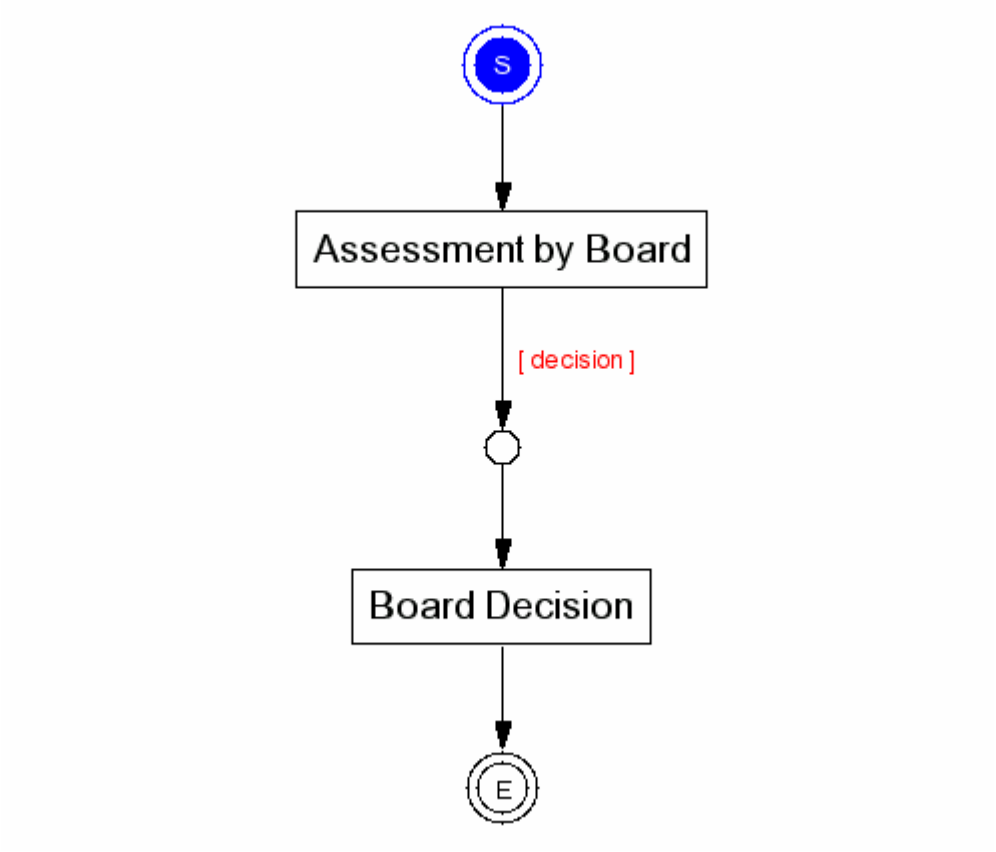
	Transition Name	Trigger Type	Role
1	Submit Application	User	CO
2	Auto Task	Automatic	
3	Assess Application	User	PO
4	Cancel	Time	
5	Reassign for Assessment	User	Outsider
6	Start Parallel Task	Auto	
7	Outsider Assessment	User	Outsider
8	Board Assessment	Message	
9	End Parallel Task	Auto	
10	Decision	User	PO
11	Commitment	Auto	

Attributes:

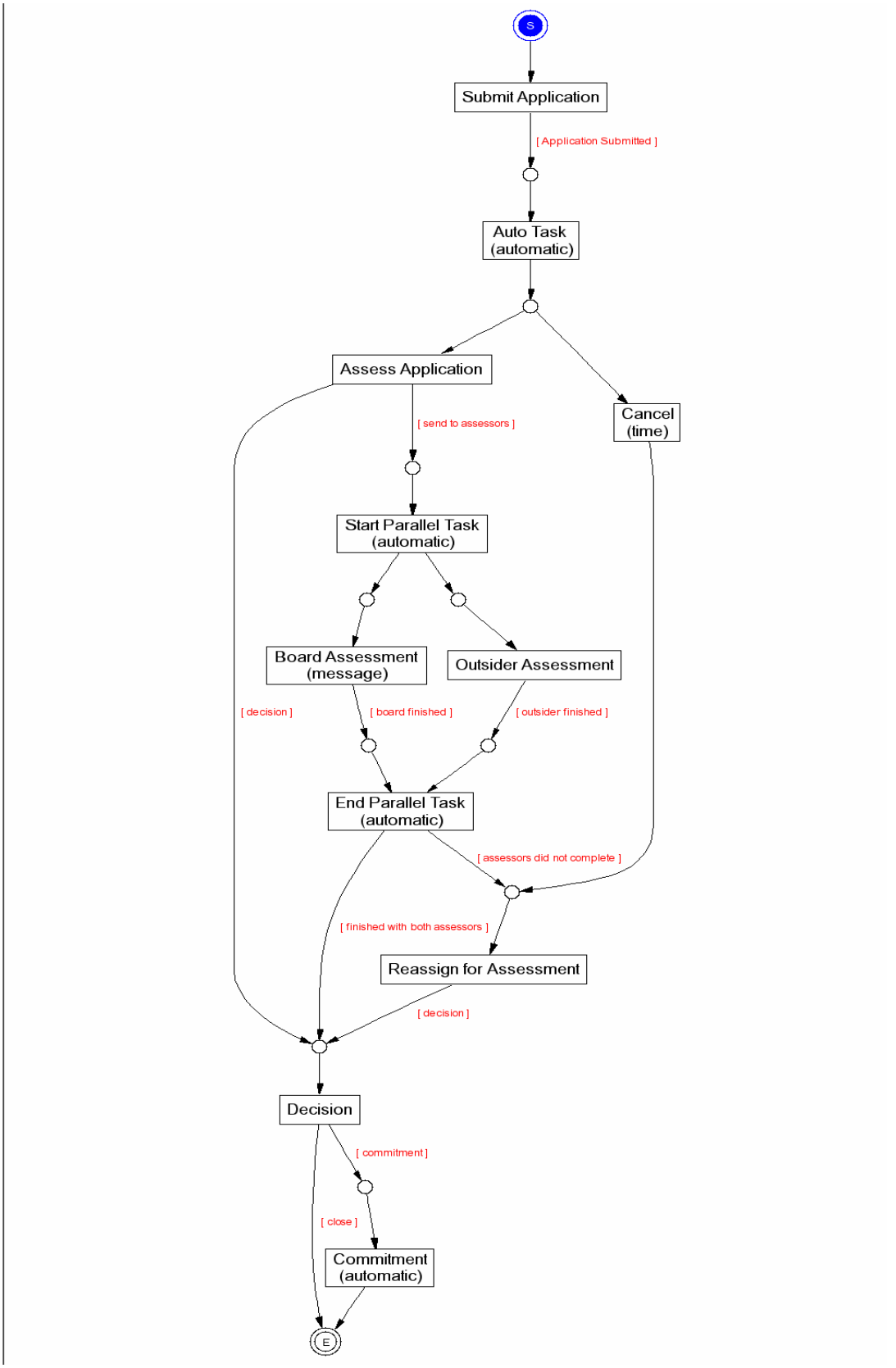
Transition	Attribute Name	Attribute Pretty name	Data type	Default Value
Submit Application	application_submitted	Application Submitted	Boolean	f
Assess Application	decision send_to_assessors	Decision Send to Assessors	boolean boolean	f f
Reassign for Assessment	decision date	Decision Date	Boolean date	f 1
Outsider Assessment	path_a	Outsider assessment finished	boolean	f
Board Assessment	path_b	Board assessment finished	boolean	t
Decision	send_commitment close	Send Commitment Close	Boolean boolean	f f

25.3 Arcs between transitions

25.3.1 Child Workflow



25.3.2 Parent Workflow



25.4 Extra Actions to occur on transitions

Transition	Action Type	Callback	Custom Argument	Attribute name
Submit application	Deadline	wf_callback.time_sysdate_plus_x	0.0006	
Submit application	Hold timeout	wf_callback.time_sysdate_plus_x	0.0006	
Board Assessment	Enable	wf_hierarchical.spawn_child	]po[_demo_child_workflow_wf	
Cancel	Time	wf_callback.time_sysdate_plus_x	0.0006	
Reassign for assessment	Notification	wf_article_callback.notification		
Decision	Deadline			date