Configurable Workflow

Introduction

Workflows can be used to define how documents should be reviewed or edited after being submitted and/or imported into DSpace. The primary focus of the workflow framework is to create a more flexible solution for the administrator to configure, and even to allow an application developer to implement custom steps, which may be configured in the workflow for the collection through a simple configuration file. Each workflow can be compared to an action that is performed on an item between its submission to the repository and the moment it is archived and published in the repository. The concept behind this approach was modeled on the configurable submission system already present in DSpace.

How to Configure your Workflows

Every submission to DSpace goes through a workflow before it is published in the repository. A workflow consists of a series of steps, each of which is an opportunity for a reviewer, editor or collection administrator to modify and/or approve/reject the submission. A step may also begin by running a {Curation Task} over the submission.

Each collection is associated with a workflow. If no explicit association is made, the collection is assigned the default workflow. These associations are configured in `config/spring/api/workflow.xml` using the workflowMapping property of the XmlWorkflowFactory bean. To make an explicit association, add an entry to the list with the collection's Handle as the 'key' and the 'name' of a Workflow bean as the 'value-ref'.

Each step in a workflow is associated with a "role" which defines who can perform that step. Role members will be notified when a new submission needs their attention. Roles are defined by DSpace user groups. If you wish to have reviewers interact with incoming submissions, you must create and fill the necessary groups. See below for details.

WORKFLOWS

To create a new workflow, add another bean with the 'class' 'org.dspace.xmlworkflow.state.Workflow' and a unique 'name'. Give it a 'steps' property containing a list of the steps that should be entered in sequence, and a 'firstStep' property which names the step to be entered first. See the default workflow for an example. An existing step may be re-used if appropriate, or you can create one to suit.

STEPS

Aside from its name, a step has a "user selection method", a "role", "actions" and "outcomes".

A step's 'userSelectionMethod' is the name of an "action" of the user-selection type. A step may, for example, let itself be claimed (for a given submission) by a single user, or it may combine the actions of multiple users. A step has exactly one 'userSelectionMethod'. See more on actions below.

A step's role defines the set of users who may perform actions on a submission that has entered that step. See more on roles below.

A step's actions are the types of work that are done in the step. See more on actions below. More than one action may be listed.

A step's outcomes connect the role members' decisions with the next step to be performed. For example, this allows a role member to accept a submission and skip subsequent steps by going directly to the final step in the workflow.

To create a new step, add a bean with 'class' org.dspace.xmlworkflow.state.Step and the necessary properties, as discussed above. See the existing steps in workflow.xml for examples.

ROLES

You may re-use existing roles, or add your own. A role has a 'name', a 'scope', and optionally a 'description'. There are three kinds of roles:

- A COLLECTION role refers to a user group associated with a specific collection. It will be named {collectionID}_{roleName}. For example, a role 'editor' with COLLECTION scope, applied to collection 123, will refer to the user group named 'editor_123', while the same role applied to collection 456 will refer to the user group 'editor_456'.
- A REPOSITORY role refers to a fixed user group, whose name is the role's name. A REPOSITORY role named 'fred' will always refer to the user group 'fred'.
- An ITEM role is assigned by a previous action in the workflow. [NEEDS MORE EXPLANATION]

To create a new role, add a bean with 'class' org.dspace.xmlworkflow.Role, the appropriate 'scope', and a unique 'name'. Be sure that the related groups exist.

ACTIONS

Actions are defined separately in 'config/spring/api/workflow-actions.xml'.

A number of actions are already defined, and these should serve most needs. Actions are implemented in Java code, so if you need a new one then you will need to write some Java in addition to configuring it here.

There are two kinds of actions: user assignment and processing. A user assignment action selects one or more role members to execute a step. A processing action modifies the state of the submission.

To configure a new Action, create a bean with a unique 'id', 'class' equal to the fully qualified name of the Java class which implements the action, and 'scope' "prototype". Add properties, constructor arguments, etc. as required by the code.

CURATION

To attach a Curation Task to a workflow step, see Curation System. Tasks are executed at the beginning of a step, before role members are notified.

HOW IT WORKS

For details of how these concepts are implemented (for example, to create new actions) see the Workflow page under DSpace Development.

Data Migration

As of DSpace 7, Configurable Workflow is the only workflow system available in DSpace. It has fully replaced the older "traditional/basic workflow" system. One major difference is that Configurable Workflow is dynamic – if a user is added to a workflow approval task *after* a workflow has already begun, they will immediately get access to any existing items in workflow. Previously, this was not possible in the "traditional" workflow system.

Workflowitem conversion/migration scripts

Depending on the workflow that is used by a DSpace installation, different scripts can be used when migrating to the new workflow.

Automatic migration

As part of the upgrade to DSpace 7 or above, all your old policies, roles, tasks and workflowitems will be automatically updated from the original workflow to the Configurable Workflow framework. This is done via this command:

```
[dspace]/bin/dspace database migrate ignored
```

The "ignored" parameter will tell DSpace to run any previously-ignored migrations on your database. As the Configurable Workflow migrations have existed in the DSpace codebase for some time, this is the only way to force them to be run.

For more information on the "database migrate" command, please see Database Utilities.

Java based migration

In case your DSpace installation uses a customized version of the workflow, the migration script might not work properly and a different approach is recommended. Therefore, an additional Java based script has been created that restarts the workflow for all the workflowitems that exist in the original workflow framework. The script will take all the existing workflowitems and place them in the first step of the configurable workflow framework thereby taking into account the XML configuration that exists at that time for the collection to which the item has been submitted. This script can also be used to restart the workflow for workflowitems in the original workflow but not to restart the workflow for items in the configurable workflow.

To execute the script, run the following CLI command:

```
[dspace]/bin/dspace dsrun org.dspace.xmlworkflow.migration.RestartWorkflow -e admin@myrespository.org
```

The following arguments can be specified when running the script:

- -e: specifies the username of an administrator user
- -n: if sending submissions through the workflow, send notification emails
- -p: the provenance description to be added to the item
- -h: help

Configuration

Main workflow configuration

As of DSpace 7, the `workflow.xml` configuration file has been migrated to use Spring Bean syntax (instead of a custom XML format). The structure of this XML has changed. If you need help migrating your old `workflow.xml` file (which started with a `<wf-config>` tag) to the new format (using `<bean>` tags), an XSLT script is available: workflow-migration.xsl

The workflow main configuration can be found in the workflow.xml file, located in **`[dspace]/config/spring/api/workflow.xml`** . An example of this workflow configuration file can be found below.

```
<beans>
    <bean class="org.dspace.xmlworkflow.XmlWorkflowFactoryImpl">
        <property name="workflowMapping">
            <util:map>
                <entry key="defaultWorkflow" value-ref="defaultWorkflow"/>
<!--            <entry key="123456789/4" value-ref="selectSingleReviewer"/>-->
<!--            <entry key="123456789/5" value-ref="scoreReview"/>-->
            </util:map>
        </property>
    </bean>

    <!--Standard DSpace workflow-->
    <bean name="defaultWorkflow" class="org.dspace.xmlworkflow.state.Workflow">
        <property name="firstStep" ref="reviewstep"/>
        <property name="steps">
            <util:list>
                <ref bean="reviewstep"/>
                <ref bean="editstep"/>
                <ref bean="finaleditstep"/>
            </util:list>
        </property>
    </bean>

    <bean id="{workflow.id}"
                  class="org.dspace.xmlworkflow.state.Workflow">
            <!-- Another workflow configuration-->
    </bean>

        <!-- Role beans.  See below. -->

        <!-- Step beans.  See below. -->

</beans>
```

workflowFactory bean (org.dspace.xmlworkflow.XmlWorkflowFactoryImpl)

The workflow map contains a mapping between collections in DSpace and a workflow configuration, and is defined by the `workflowMapping` property of the workflow factory. Similar to the configuration of the submission process, the mapping can be done based on the handle of the collection. The mapping with "defaultWorkflow" as the value for the collection mapping, will be used for the collections not occurring in other mapping tags. Each mapping is defined by a "`entry`" element with two attributes:

- key: can either be a collection handle or "defaultWorkflow"
- value-ref: the value of this attribute points to one of the workflow configurations defined by the "Workflow" beans

workflow beans (org.dspace.xmlworkflow.state.Workflow)

The workflow bean is a repeatable XML element and represents one workflow process. It requires the following:

- "`name`" attribute: a unique name used for the identification of the workflow and used in the workflow to collection mapping
- "`firstStep`" property: the identifier of the first step of the workflow. This step will be the entry point of this workflow-process. When a new item has been committed to a collection that uses this workflow, the step configured in the "`firstStep`" property will he the first step the item will go through.
- "`steps`" property: a list of all steps within this workflow (in the order they will be processed).

role beans (org.dspace.xmlworkflow.Role)

Each workflow step has defined "role" property. A role represents one or more existing DSpace EPersons or Groups and can be used to assign them to one or more steps in the workflow process. One role is represented by one "role" bean and has the following:

- "`id`" attribute: a unique identifier (in one workflow process) for the role
- "`description`" property: optional attribute to describe the role
- "`scope`" property: optional attribute that is used to find our group and must have one of the following values, which are defined as constant fields of `org.dspace.xmlworkflow.Role.Scope`:
    - COLLECTION: The collection value specifies that the group will be configured at the level of the collection. This type of groups is the same as the type that existed in the original workflow system. In case no value is specified for the scope attribute, the workflow framework assumes the role is a collection role.
    - REPOSITORY: The repository scope uses groups that are defined at repository level in DSpace. The name attribute should exactly match the name of a group in DSpace.
    - ITEM: The item scope assumes that a different action in the workflow will assign a number of EPersons or Groups to a specific workflow-item in order to perform a step. These assignees can be different for each workflow item.
- "`name`" property: The name specified in the name attribute of a role will be used to lookup an eperson group in DSpace. The lookup will depend on the scope specified in the "`scope`" attribute:
    - COLLECTION: The workflow framework will look for a group containing the name specified in the name attribute and the ID of the collection for which this role is used.
    - REPOSITORY: The workflow framework will look for a group with the same name as the name specified in the `name` attribute.
    - ITEM: in case the item scope is selected, the name of the role attribute is not required.

```
<bean id="reviewer" class="org.dspace.xmlworkflow.Role">
    <property name="scope" value="#{ T(org.dspace.xmlworkflow.Role.Scope).COLLECTION}"/>
    <property name="name" value="Reviewer"/>
    <property name="description" value="The people responsible for this step are able to edit the metadata of
incoming submissions, and then accept or reject them."/>
</bean>
```

step beans (org.dspace.xmlworkflow.state.Step)

The step element represents one step in the workflow process. A step represents a number of actions that must be executed by one specified role. In case no `role` attribute is specified, the workflow framework assumes that the DSpace system is responsible for the execution of the step and that no user interface will be available for each of the actions in this step. The step element has the following in order to further configure it:

- "`name`" attribute: The name attribute specifies a unique identifier for the step. This identifier will be used when configuring other steps in order to point to this step. This identifier can also be used when configuring the start step of the workflow item.
- "`userSelectionMethod`" property: This attribute defines the `UserSelectionAction` that will be used to determine how to attach users to this step for a workflow-item. The value of this attribute must refer to the identifier of an action bean in the workflow-actions.xml. Examples of the user attachment to a step are the currently used system of a task pool or as an alternative directly assigning a user to a task.
- "`role`" property: optional attribute that must point to the `id` attribute of a role element specified for the workflow. This role will be used to define the epersons and groups used by the `userSelectionMethod`.
- RequiredUsers

```
<bean name="reviewstep" class="org.dspace.xmlworkflow.state.Step">
    <property name="userSelectionMethod" ref="claimaction"/>
    <property name="role" ref="reviewer"/>
    <property name="outcomes">
        <util:map>
            <entry key="#{ T(org.dspace.xmlworkflow.state.actions.ActionResult).OUTCOME_COMPLETE}"
                   value-ref="editstep"/>
        </util:map>
    </property>
    <property name="actions">
        <util:list>
            <ref bean="reviewaction"/>
        </util:list>
    </property>
</bean>
```

Each step contains a number of actions that the workflow item will go through. In case the action has a user interface, the users responsible for the exectution of this step will have to execute these actions before the workflow item can proceed to the next action or the end of the step.

There is also an optional subsection that can be defined for a step part called "`outcomes`". This can be used to define outcomes for the step that differ from the one specified in the nextStep attribute. Each action returns an integer depending on the result of the action. The default value is "0" and will make the workflow item proceed to the next action or to the end of the step.
In case an action returns a different outcome than the default "0", the alternative outcomes will be used to lookup the next step. The "`outcomes`" element contains a number of steps, each having a status attribute. This status attribute defines the return value of an action. The value of the element will be used to lookup the next step the workflow item will go through in case an action returns that specified status.

Workflow actions configuration

API configuration

The workflow actions configuration is located in the `[dspace]/config/spring/api/` directory and is named "`workflow-actions.xml`". This configuration file describes the different Action Java classes that are used by the workflow framework. Because the workflow framework uses Spring framework for loading these action classes, this configuration file contains Spring configuration.

This file contains the beans for the actions and user selection methods referred to in the `workflow.xml`. In order for the workflow framework to work properly, each of the required actions must be part of this configuration.

```
<beans
    xmlns="http://www.springframework.org/schema/beans"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xmlns:util="http://www.springframework.org/schema/util"
    xsi:schemaLocation="http://www.springframework.org/schema/beans http://www.springframework.org/schema/beans
/spring-beans-2.0.xsd
                           http://www.springframework.org/schema/util http://www.springframework.org/schema/util
/spring-util-2.0.xsd">

    <!-- At the top are our bean class identifiers --->
    <bean id="{action.api.id}" class="{class.path}" scope="prototype"/>
    <bean id="{action.api.id.2}" class="{class.path}" scope="prototype"/>

         <!-- Below the class identifiers come the declarations for out actions/userSelectionMethods -->

         <!-- Use class workflowActionConfig for an action -->
         <bean id="{action.id}" class="oorg.dspace.xmlworkflow.state.actions.WorkflowActionConfig" scope="
prototype">
             <constructor-arg type="java.lang.String" value="{action.id}"/>

             <property name="processingAction" ref="{action.api.id}"/>
             <property name="requiresUI" value="{true/false}"/>
         </bean>

         <!-- Use class UserSelectionActionConfig for a user selection method -->
         <!--User selection actions-->
         <bean id="{action.api.id.2}" class="org.dspace.xmlworkflow.state.actions.UserSelectionActionConfig"
scope="prototype">
             <constructor-arg type="java.lang.String" value="{action.api.id.2}"/>

             <property name="processingAction" ref="{user.selection.bean.id}"/>
             <property name="requiresUI" value="{true/false}"/>
         </bean>
</beans>
```

Two types of actions are configured in this Spring configuration file:

- User selection action: This type of action is always the first action of a step and is responsible for the user selection process of that step. In case a step has no role attached, no user will be selected and the `NoUserSelectionAction` is used.
- Processing action: This type of action is used for the actual processing of a step. Processing actions contain the logic required to execute the required operations in each step. Multiple processing actions can be defined in one step. These user and the workflow item will go through these actions in the order they are specified in the workflow configuration unless an alternative outcome is returned by one of them.

User Selection Action

Each user selection action that is used in the workflow configuration refers to a bean definition in the `workflow-actions.xml` file. In order to define a new user selection action, the following XML code is used:

```
<bean id="{action.api.id.2}" class="org.dspace.xmlworkflow.state.actions.UserSelectionActionConfig" scope="
prototype">
    <constructor-arg type="java.lang.String" value="{action.api.id.2}"/>

    <property name="processingAction" ref="{user.selection.bean.id}"/>
    <property name="requiresUI" value="{true/false}"/>
</bean>
```

This bean defines a new UserSelectionActionConfig and the following child tags:

- constructor-arg: This is a constructor argument containing the ID of the task. This is the same as the `id` attribute of the bean and is used by the workflow configuration to refer to this action.
- property `processingAction`: This tag refers the the ID of the API bean, responsible for the implementation of the API side of this action. This bean should also be configured in this XML.
- property `requiresUI`: In case this property is true, the workflow framework will expect a user interface for the action. Otherwise the framework will automatically execute the action and proceed to the next one.

Processing Action

Processing actions are configured similarly to the user selection actions. The only difference is that these processing action beans are implementations of the WorkflowActionConfig class instead of the UserSelectionActionConfig class.

Authorizations

Currently, the authorizations are always granted and revoked based on the tasks that are available for certain users and groups. The types of authorization policies that is granted for each of these is always the same:

- READ
- WRITE
- ADD
- DELETE

Database

The workflow uses a separate metadata schema named `workflow`. The fields this schema contains can be found in the `[dspace]/config/registries` directory and in the file `workflow-types.xml`. This schema is only used when using the score reviewing system at the moment, but one could always use this schema if metadata is required for custom workflow steps.

The following tables have been added to the DSpace database. All tables are prefixed with 'cwf_' to avoid any confusion with the existing workflow related database tables:

cwf_workflowitem

The cwf_workflowitem table contains the different workflowitems in the workflow. This table has the following columns:

- workflowitem_id: The identifier of the workflowitem and primary key of this table
- item_id: The identifier of the DSpace item to which this workflowitem refers.
- collection_id: The collection to which this workflowitem is submitted.
- multiple_titles: Specifies whether the submission has multiple titles (important for submission steps)
- published_before: Specifies whether the submission has been published before (important for submission steps)
- multiple_files: Specifies whether the submission has multiple files attached (important for submission steps)

cwf_collectionrole

The cwf_collectionrole table represents a workflow role for one collection. This type of role is the same as the roles that existed in the original workflow meaning that for each collection a separate group is defined to described the role. The cwf_collectionrole table has the following columns:

- collectionrol_id: The identifier of the collectionrole and the primaty key of this table
- role_id: The identifier/name used by the workflow configuration to refer to the collectionrole
- collection_id: The collection identifier for which this collectionrole has been defined
- group_id: The group identifier of the group that defines the collection role

cwf_workflowitemrole

The cwf_workflowitemrole table represents roles that are defined at the level of an item. These roles are temporary roles and only exist during the execution of the workflow for that specific item. Once the item is archived, the workflowitemrole is deleted. Multiple rows can exist for one workflowitem with e.g. one row containing a group and a few containing epersons. All these rows together make up the workflowitemrole The cwf_workflowitemrole table has the following columns:

- workflowitemrole_id: The identifier of the workflowitemrole and the primaty key of this table
- role_id: The identifier/name used by the workflow configuration to refer to the workflowitemrole
- workflowitem_id: The cwf_workflowitem identifier for which this workflowitemrole has been defined
- group_id: The group identifier of the group that defines the workflowitemrole role
- eperson_id: The eperson identifier of the eperson that defines the workflowitemrole role

cwf_pooltask

The cwf_pooltask table represents the different task pools that exist for a workflowitem. These task pools can be available at the beginning of a step and contain all the users that are allowed to claim a task in this step. Multiple rows can exist for one task pool containing multiple groups and epersons. The cwf_pooltask table has the following columns:

- pooltask_id: The identifier of the pooltask and the primaty key of this table
- workflowitem_id: The identifier of the workflowitem for which this task pool exists
- workflow_id: The identifier of the workflow configuration used for this workflowitem
- step_id: The identifier of the step for which this task pool was created
- action_id: The identifier of the action that needs to be displayed/executed when the user selects the task from the task pool
- eperson_id: The identifier of an eperson that is part of the task pool
- group_id: The identifier of a group that is part of the task pool

cwf_claimtask

The cwf_claimtask table represents a task that has been claimed by a user. Claimed tasks can be assigned to users or can be the result of a claim from the task pool. Because a step can contain multiple actions, the claimed task defines the action at which the user has arrived in a particular step. This makes it possible to stop working halfway the step and continue later. The cwf_claimtask table contains the following columns:

- claimtask_id: The identifier of the claimtask and the primary key of this table
- workflowitem_id: The identifier of the workflowitem for which this task exists
- workflow_id: The id of the workflow configuration that was used for this workflowitem
- step_id: The step that is currenlty processing the workflowitem
- action_id: The action that should be executed by the owner of this claimtask
- owner_id: References the eperson that is responsible for the execution of this task

cwf_in_progress_user

The cwf_in_progess_user table keeps track of the different users that are performing a certain step. This table is used because some steps might require multiple users to perform the step before the workflowitem can proceed. The cwf_in_progress_user table contains the following columns:

- in_progress_user_id: The identifier of the in progress user and the primary key of this table
- workflowitem_id: The identifier of the workflowitem for which the user is performing or has performed the step.
- user_id: The identifier of the eperson that is performing or has performe the task
- finished: Keeps track of the fact that the user has finished the step or is still in progress of the execution

Additional workflow steps/actions and features

These optional steps are only supported in 7.5 and later.

These optional workflow steps are pre-defined in the "workflow.xml" but are not used by default.

Optional workflow steps: Select single reviewer workflow

This workflow makes it possible to assign a single user to review an item. This workflow configuration skips the task pool option meaning that the assigned reviewer no longer needs to claim the task. The configuration consists of the following 2 steps.

- selectReviewerStep: During this step, a user has the ability to select a responsible user to review the workflowitem. This means that for each workflowitem, a different user can be selected. Because a user is assigned, the task pool is no longer required.
    - The available users to select from are defined in the "action.selectrevieweraction.group" setting in workflow.cfg. This setting must list the name of a group of reviewers to select from (default value = "Reviewers" group).
- singleUserReviewStep: The start of the reviewstep is different than the typical task pool. Instead of having a task pool, the user will be automatically assigned to the task. However, the user still has the option to reject the task (in case he or she is not responsible for the assigned task) or review the item. In case the user rejects the task, the workflowitem will be sent to the another step in the workflow as an alternative to the default outcome.

Optional workflow steps: Score review workflow

The score review system allows reviewers to give the reviewed item a rating. Depending on the results of the rating, the item will be approved to go to the next workflow step or will be sent to an alternative step. The scrore review workflow consists of the following 2 steps.

- scoreReviewStep: The group of responsible users for the score reviewing will be able to claim the task from the taskpool. Depending on the configuration, a different number of users can be required to execute the task (default is requiredusers=2). This means that the task will be available in the task pool until the required number of users has at least claimed the task. Once everyone of them has finished the task, the next (automatic) processing step is activated.
- evaluationStep: During the evaluationstep, no user interface is required. The workflow system will automatically execute the step that evaluates the different scores (which corresponds to a rating from 1-5). In case the average score is greater than the average "minimumAcceptanceScore", the item is approved, otherwise it is rejected. (The minimum average score is set by adjusting the `minimumAcceptanceScore` property passed to `evaluationactionAPI` in `config/spring/api/workflow-actions.xml`.)

Workflow overview features

The DSpace UI also provides a feature to allow Administrators to see & administer all active workflows (workitems). This feature is provided in the "Administer Workflow" menu option. Currently, the Administrator has the ability to permanently delete the workflowitem, or to send it back to the original submitter.