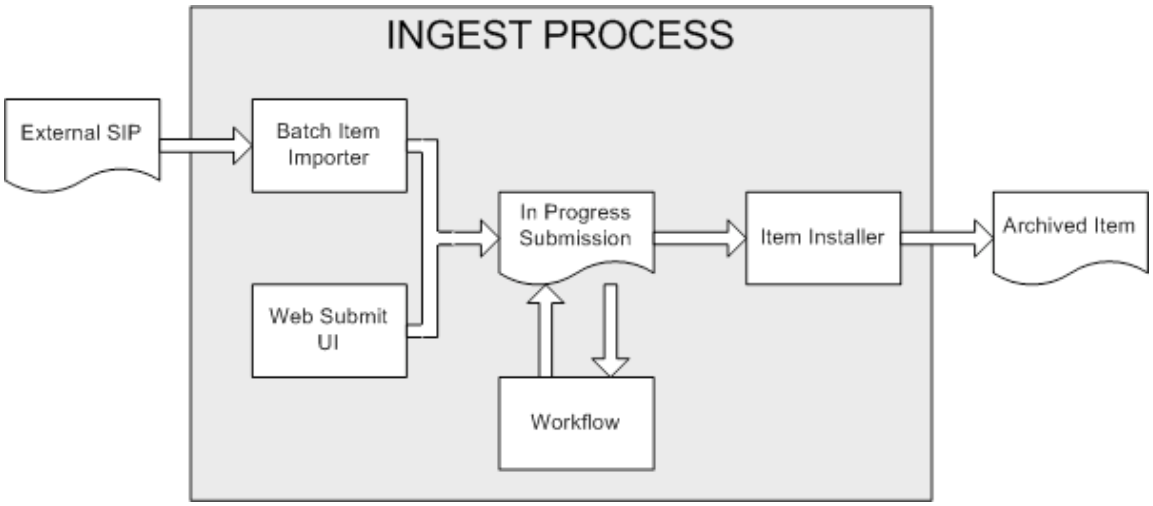
Configuring DSpace Item Submissions

1. Submission Input Forms

Manual Item Submission

The following diagram outlines the different ways to deposit content into DSpace. This session will go into detail about the Web Submit UI and the workflow.



In DSpace an item is made up of: Metadata, Bundles (e.g. ORIGINAL / LICENCE / TEXT), and Bitstreams.

A typical submission process through the web UI is as follows. First choose a collection to submit the item to. Then answer initial questions, such as whether or not the item has been publishedbefore or has more than one title. Proceed to enter metadata for the item in one ore more steps/pages (edit metadata steps). Then upload one or more files, verify the submission and finally you will have to agree to a deposit licence.

Metadata Input

The “Edit Metadata” steps in the web submission process are fully cusomizable. Any field in the DSpace metadata registry can be added to the forms in the edit metadata steps and these steps are configurable in the input-forms.xml file. The metadata registry can be accessed through the Dspace admin interface and supports multiple schemas. By default the metadata registry will contain a set of qualified dublin core fields, and the admin can add, change or delete any of these fields in the web user interface. This session will not go into detail on how to manage the metadata registry, for this the reader is reffered to the dspace documentation. The next section will outline how to configure the edit metadata steps.

Configuring input-forms.xml

The XML configuration file has a single top-level element, `input-forms`, which contains three elements in a specific order. The outline is as follows:

```
<input-forms>
  <form-map>
    <name-map collection-handle="default" form-name="traditional"/>
    ...
  </form-map>
  <form-definitions>
    <form name="traditional">
    ...
    </form>
  </form-definitions>
  <form-value-pairs>
    <value-pairs value-pairs-name="common_iso_languages"
```



```
      dc-term="language_iso">
    ...
    </value-pairs>
  </form-value-pairs>
</input-forms>
```

Adding a Collection Map

Each name-map element within `form-map` associates a collection with the name of a form set. Its `collection-handle` attribute is the Handle of the collection, and its `form-name` attribute is the form set name, which must match the name attribute of a `form` element.

For example, the following fragment shows how the collection with handle "123456789/42" is attached to the "TechRpt" form set:

```
<form-map>
  <name-map collection-handle="123456789/42" form-name=" TechRpt"/>
  ...
</form-map>

<form-definitions>
  <form name="TechRept">
  ...
 </form>
</form-definitions>
```

It's a good idea to keep the definition of the `default` name-map from the example `input-forms.xml` so there is always a default for collections which do not have a custom form set.

Adding a Form Set

You can add a new form set by creating a new form element within the `form-definitions` element. It has one attribute, `name`, which as seen above must match the value of the `name-map` for the collections it is to be used for.

Forms and Pages

The content of the `form` is a sequence of `page` elements. Each of these corresponds to a Web page of forms for entering metadata elements, presented in sequence between the initial "Describe" page and the final "Verify" page (which presents a summary of all the metadata collected).

A form must contain at least one and at most six pages. They are presented in the order they appear in the XML. Each `page` element must include a `number` attribute, that should be its sequence number, e.g.

```
<page number="1">
```

The `page` element, in turn, contains a sequence of `field` elements. Each field defines an interactive dialog where the submitter enters one of the Dublin Core metadata items.

Composition of a Field

Each `field` contains the following elements, in the order indicated. The required sub-elements are so marked:

dc-schema *(Required)*

Name of metadata schema employed, e.g. dc for Dublin Core. This value must match the value of the schema element defined in `dublin-core-types.xml`

dc-element *(Required)*

Name of the Dublin Core element entered in this field, e.g. `contributor`.

dc-qualifier

Qualifier of the Dublin Core element entered in this field, e.g. when the field is `contributor.advisor` the value of this element would be `advisor`. Leaving this out means the input is for an unqualified DC element.

repeatable

Value is `true` when multiple values of this field are allowed, `false` otherwise. When you mark a field repeatable, the UI servlet will add a control to let the user ask for more fields to enter additional values. Intended to be used for arbitrarily-repeating fields such as subject keywords, when it is impossible to know in advance how many input boxes to provide.

label *(Required)*

Text to display as the label of this field, describing what to enter, e.g. "`Your Advisor's Name`".

input-type *(Required)*

Defines the kind of interactive widget to put in the form to collect the Dublin Core value. Content must be one of the following keywords:

- **onebox** - A single text-entry box.
- **twobox** - A pair of simple text-entry boxes, used for repeatable values such as the DC subject item. Note: The 'twobox' input type is rendered the same as a 'onebox' in the XML-UI, but both allow for ease of adding multiple values.
- **textarea** - Large block of text that can be entered on multiple lines, e.g. for an abstract.
- **name** - Personal name, with separate fields for family name and first name. When saved they are appended in the format 'LastName, FirstName'
- **date** - Calendar date. When required, demands that at least the year be entered.
- **series** - Series/Report name and number. Separate fields are provided for series name and series number, but they are appended (with a semicolon between) when saved.
- **dropdown** - Choose value(s) from a "drop-down" menu list. Note: You must also include a value for the `value-pairs-name` attribute to specify a list of menu entries from which to choose. Use this to make a choice from a restricted set of options, such as for the language item.
- **qualdrop_value** - Enter a "qualified value", which includes both a qualifier from a drop-down menu and a free-text value. Used to enter items like alternate identifiers and codes for a submitted item, e.g. the DC identifier field. Note: As for the dropdown type, you must include the `value-pairs-name` attribute to specify a menu choice list.
- **list** - Choose value(s) from a checkbox or radio button list. If the `repeatable` attribute is set to `true`, a list of checkboxes is displayed. If the `repeatable` attribute is set to `false`, a list of radio buttons is displayed. Note: You must also include a value for the `value-pairs-name` attribute to specify a list of values from which to choose.

hint *(Required)*

Content is the text that will appear as a "hint", or instructions, next to the input fields. Can be left empty, but it must be present.

required

When this element is included with any content, it marks the field as a required input. If the user tries to leave the page without entering a value for this field, that text is displayed as a warning message. For example,

```
<required>You must enter a title.</required>
```

Note that leaving the required element empty will not mark a field as required, e.g.:

```
<required></required>
```

visibility

When this optional element is included with a value, it restricts the visibility of the field to the scope defined by that value. If the element is missing or empty, the field is visible in all scopes. Currently supported scopes are:

- **workflow** : the field will only be visible in the workflow stages of submission. This is good for hiding difficult fields for users, such as subject classifications, thereby easing the use of the submission system.
- **submit** : the field will only be visible in the initial submission, and not in the workflow stages.

In addition, you can decide which type of restriction apply: read-only or hiding the field (default behaviour) using the `otherwise` attribute of the `visibility` XML element. For example:

```
<visibility otherwise="readonly">workflow</visibility>
```

Note that it is considered a configuration error to limit a field's scope while also requiring it - an exception will be generated when this combination is detected.

vocabulary

When this optional element is included, the taxonomy defined by the value will be assigned to the field. This element can be used with the onebox and twobox input-type. The taxonomy definition will be retrieved from `[dspace]/config/controlled-vocabularies/value.xml` where value is the contents of the vocabulary element. The vocabulary is only provided for the JSPUI, it has not been ported to the XMLUI. During the submission, a link to the controlled vocabulary will be included in the field. This link will lead to a pop-up containing a collapsable tree of the values in the taxonomy definition. In order to enable vocabularies, `webui.controlledvocabulary.enable = true` must be included in the dspace.cfg file.

closedVocabulary

When this optional element is included and contains the value "`true`", the field value can only be the value chosen by the vocabulary. By default, the taxonomy can be used, but the user can also choose to insert text independent of the taxonomy. Configuring `closedVocabulary` to "`true`" will ensure only a value of the taxonomy is inserted. The element can only be used when a `vocabulary` is defined.

Automatically omitted Fields (initial questions)

You may notice that some fields are automatically skipped when a custom form page is displayed, depending on the kind of item being submitted. This is because the DSpace user-interface engine skips Dublin Core fields which are not needed, according to the initial description of the item. For example, if the user indicates there are no alternate titles on the first "Describe" page (the one with a few checkboxes), the input for the `title.alternative` DC element is automatically removed, even on custom submission pages.

When a user initiates a submission, DSpace first displays what we'll call the "initial-questions page". By default, it contains three questions with check-boxes:

1. The item has more than one title, e.g. a translated title
   Controls `title.alternative` field.

2. The item has been published or publicly distributed before
   Controls DC fields:
   - `date.issued`
   - `publisher`
   - `identifier.citation`

3. The item consists of more than one file
   Does not affect any metadata input fields.

The answers to the first two questions control whether inputs for certain of the DC metadata fields will displayed, even if they are defined as fields in a custom page. Conversely, if the metadata fields controlled by a checkbox are not mentioned in the custom form, the checkbox is not displayed in the initial page to avoid confusing or misleading the user.

Adding Value-Pairs

Finally, your custom form description needs to define the "value pairs" for any fields with input types that refer to them. Do this by adding a `value-pairs` element to the contents of `form-value-pairs`. It has the following required attributes:

- **value-pairs-name** -- Name by which an `input-type` refers to this list.
- **dc-term** -- Qualified Dublin Core field for which this choice list is selecting a value.

Each `value-pairs` element contains a sequence of `pair` sub-elements, each of which in turn contains two elements:
- **displayed-value** -- Name shown (on the web page) for the menu entry.
- **stored-value** -- Value stored in the DC element when this entry is chosen.

Unlike the HTML `select` tag, there is no way to indicate one of the entries should be the default, so the first entry is always the default choice.

Here is a menu of types of common identifiers:

```
<value-pairs value-pairs-name="common_identifiers" dc-term="identifier">
  <pair>
    <displayed-value>Gov't Doc#</displayed-value>
    <stored-value>govdoc</stored-value>
  </pair>
  <pair>
    <displayed-value>URI</displayed-value>
    <stored-value>uri</stored-value>
  </pair>
  <pair>
    <displayed-value>ISBN</displayed-value>
    <stored-value>isbn</stored-value>
  </pair>
</value-pairs>
```

dspace.cfg

Configuring the File Upload step

The Upload step in the DSpace submission process has two configuration options which can be set with your `[dspace]/config/dspace.cfg` configuration file. They are as follows:

- upload.max - The maximum size of a file (in bytes) that can be uploaded from the JSPUI (not applicable for the XMLUI). It defaults to 536870912 bytes (512MB). You may set this to -1 to disable any file size limitation.
- webui.submit.upload.required - Whether or not all users are required to upload a file when they submit an item to DSpace. It defaults to 'true'. When set to 'false' users will see an option to skip the upload step when they submit a new item.

Exercises

Look at the example input-forms.xml file and experiment with a custom form to learn this specification language thoroughly. It is a very simple way to express the layout of data-entry forms, but the only way to learn all its subtleties is to use it.

2. Submission Steps

Submission Stages

item-submission.xml

The `[dspace]/config/item-submission.xml` contains the submission configurations for both the DSpace JSP user interface (JSPUI) or the DSpace XML user interface (XMLUI or Manakin). This configuration file contains detailed documentation within the file itself, which should help you better understand how to best utilize it.

The Structure of item-submission.xml

```
<item-submission>
 <submission-map>
  <name-map collection-handle="default" submission-name="traditional" />
 </submission-map>
 <step-definitions>
  <step id="collection">
   <heading></heading>
     <processing-class>
       org.dspace.submit.step.SelectCollectionStep
     </processing-class>
     <jspui-binding>
       org.dspace.app.webui.submit.step.JSPSelectCollectionStep
     </jspui-binding>
     <xmlui-binding>
       org.dspace.app.xmlui.aspect.submission.submit.SelectCollectionStep
     </xmlui-binding>
     <workflow-editable>false</workflow-editable>
  </step>
 </step-definitions>
 <submission-definitions>
  <submission-process name="traditional">
        <step>
          …
        </step>
  </submission-process>
 </submission-definitions>
</item-submission>
```

Because this file is in XML format, you should be familiar with XML before editing this file. By default, this file contains the "traditional" Item Submission Process for DSpace, which consists of the following Steps (in this order):

Select Collection -> Initial Questions -> Describe -> Upload -> Verify -> License -> Complete

If you would like to customize the steps used or the ordering of the steps, you can do so within the `<submission-definition>` section of the `item-submission.xml`.

In addition, you may also specify different Submission Processes for different DSpace Collections. This can be done in the `<submission-map>` section. The `item-submission.xml` file itself documents the syntax required to perform these configuration changes.

Defining Steps (<step>) within the item-submission.xml

This section describes how Steps of the Submission Process are defined within the `item-submission.xml`.

XPath is an important part of the XSLT standard. XPath knowledge is necessary to be able to manipulate XML with XSLT documents. XSLT will perform operations on an XML file to manipulate and transform it into some other form (eg. XHTML) and XPath is essential to XSLT in the sense that XPath is the technique used to select portions of the input XML file to be transformed by XSLT templates. More about XPath in XSLT in section “3. XSLT”.

The Upload step in the DSpace submission process has two configuration options which can be set with your `[dspace]/config/dspace.cfg` configuration file. They are as follows:

- upload.max - The maximum size of a file (in bytes) that can be uploaded from the JSPUI (not applicable for the XMLUI). It defaults to 536870912 bytes (512MB). You may set this to -1 to disable any file size limitation.
- webui.submit.upload.required - Whether or not all users are required to upload a file when they submit an item to DSpace. It defaults to 'true'. When set to 'false' users will see an option to skip the upload step when they submit a new item.

Exercises

Replace the initial questions step by a step without a userinterface which preconfigures the values to set multiple files to true, multiple titles to false, and published before to true. Check whether multiple files can be added, the issue date can be added, and the alternative title cannot be added despite the fact that it is configured in input-forms.xml.

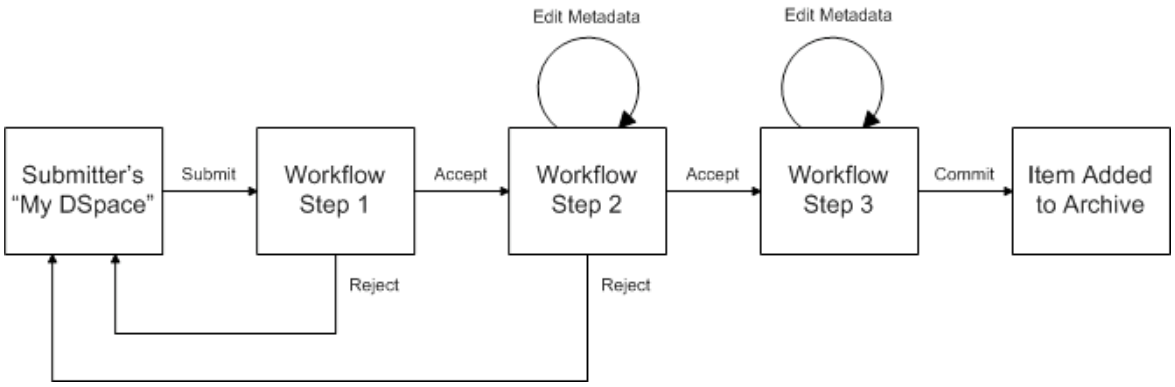
3. Configuring Workflow

This section summarizes and expands on the DSpace documentation regarding workflow.

Workflow Steps

A collection's workflow can have up to three steps. Each collection may have an associated e-person group assigned for performing each step; if no group is associated with a certain step, that step is skipped. If a collection has no e-person groups associated with any step, submissions to that collection are committed straight into the archive.

When a step is invoked, the task of performing that workflow step put in the task pool of the associated group. If any one member of that group takes the task from the pool, it is then removed from the task pool.



Workflow Roles

Submitter

The E-People and Groups that have permission to submit new items to this collection.

Reviewer

Accept/Reject Step: The people responsible for this step are able to accept or reject incoming submissions. However, they are not able to edit the submission's metadata.

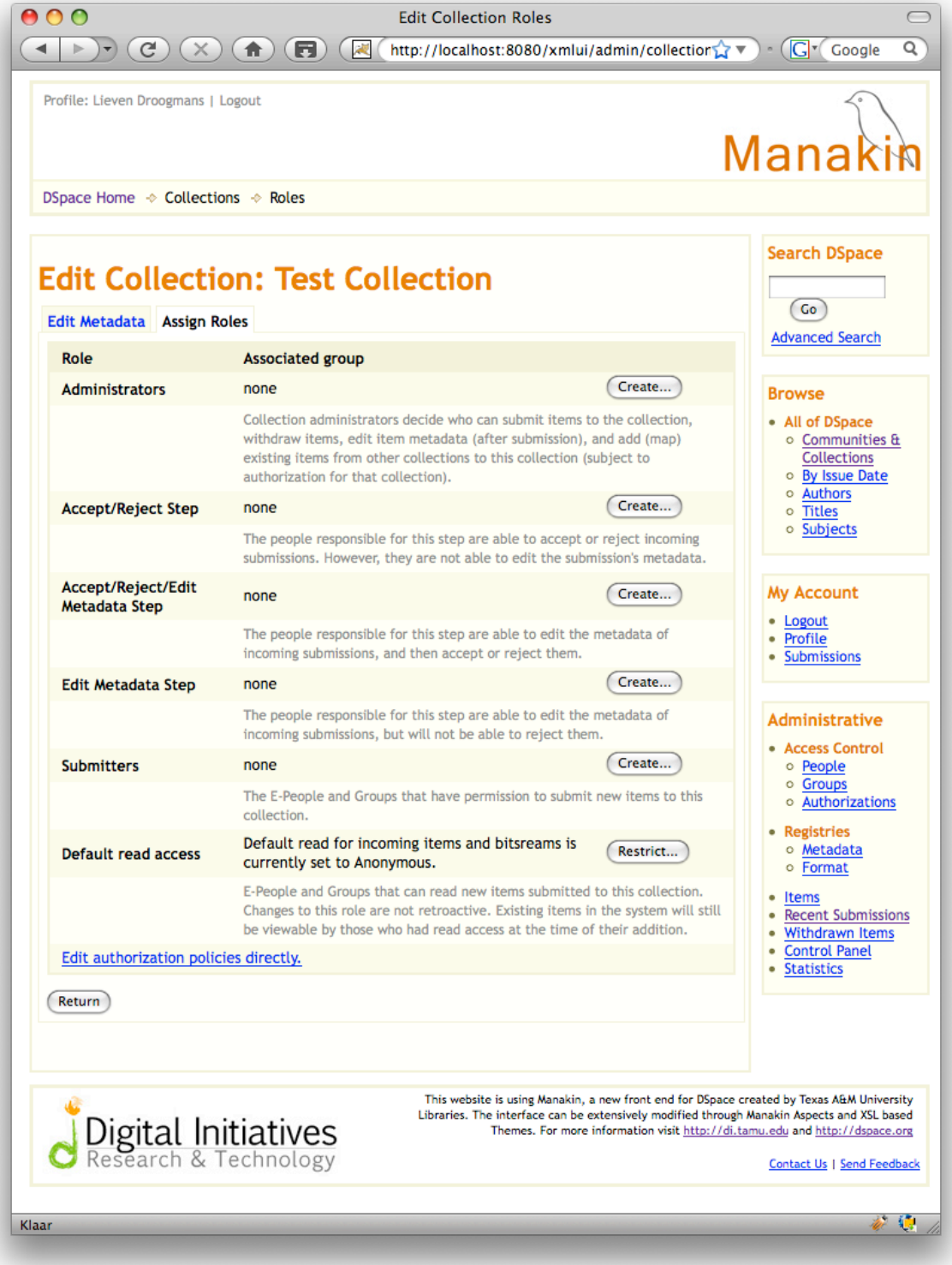
Metadata Editor

Edit Metadata Step: The people responsible for this step are able to edit the metadata of incoming submissions, but will not be able to reject them.

Collection Administrator

This role is not actually part of the workflow, but can configure the other roles in the workflow. Collection administrators decide who can submit items to the collection, withdraw items, edit item metadata (after submission), and add (map) existing items from other collections to this collection (subject to authorization for that collection).

These workflow roles can be configured in the DSpace web interface as shown in the screenshot below.



TERMS OF USE

PLEASE READ THESE TERMS OF USE CAREFULLY BEFORE USING THESE COURSE MATERIALS. By using these course materials, you signify your assent to these terms of use. If you do not agree to these terms of use, please do not use these course materials.

RESTRICTIONS ON USE OF MATERIALS. These course materials are owned by @mire NV, Technologielaan 9, 3001 Heverlee (Belgium).

No components from these course materials owned, licensed or controlled by @mire NV may be copied, reproduced, republished, uploaded, posted, transmitted, or distributed in any way, except that you may download one copy of the materials on any single computer for your personal, non-commercial home use only, provided you keep intact all copyright and other proprietary notices.

Modification of the materials or use of the materials for any other purpose is a violation of @mire's copyright and other proprietary rights. The use of any such material on any other web site or networked computer environment is prohibited.

To request permission to reproduce materials,
call +32 2 888 29 56,
email info@atmire.com,
or write to @mire NV, Technologielaan 9, 3001 Heverlee, Belgium.

