Confederation of
Open Access Repositories

Implementation guide for COAR Controlled Vocabularies

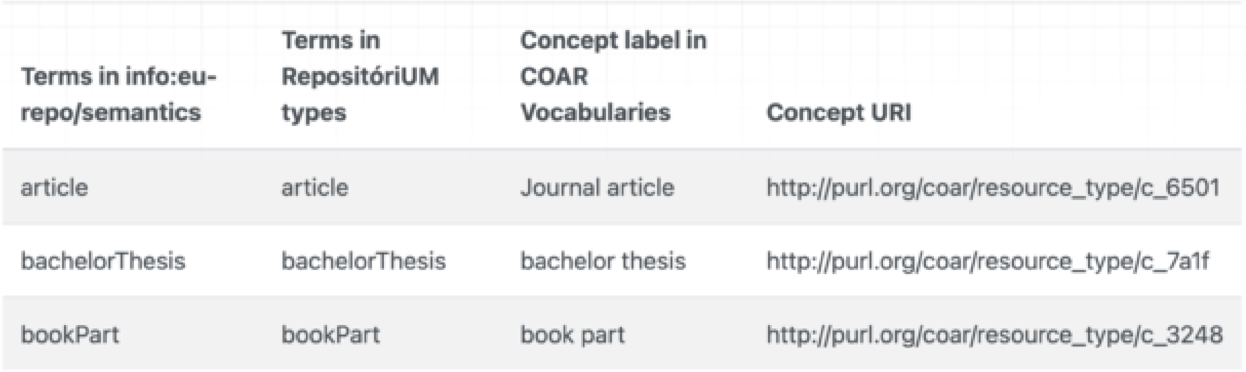
The continuous review update and maintenance of the COAR Controlled Vocabularies and their adoption by the most commonly used open repository software is a way to enhance the interoperability across repositories and with other related systems such as harvesters, CRIS systems, data repositories and publishers.

COAR developed and maintains three vocabularies: Resource types vocabulary, access rights vocabulary and version types vocabulary. For the latest release and other details, please click on the hyperlink for the relevant vocabularies. On this page, you will find guidance on how to implement COAR's controlled vocabularies.

Mapping vocabularies

To implement vocabularies locally, you may need to map the other terms of locally used vocabularies with COAR Controlled Vocabularies concepts.

Please note that it is not mandatory to use the whole set of concepts in COAR Vocabularies. COAR Vocabularies are offered in a hierarchical structure and depending on the needs of the repository, you may decide not to use some of the concepts and/or labels. See an example of mapping table below:



Terms in info:eu-repo/semantics	Terms in RepositóriUM types	Concept label in COAR Vocabularies	Concept URI
article	article	Journal article	http://purl.org/coar/resource_type/c_6501
bachelorThesis	bachelorThesis	bachelor thesis	http://purl.org/coar/resource_type/c_7a1f
bookPart	bookPart	book part	http://purl.org/coar/resource_type/c_3248

Implementation in DSpace and DSpace-CRIS

COAR vocabularies can be implemented in repositories operating on DSpace by customising input forms or using the Dspace software functionality.

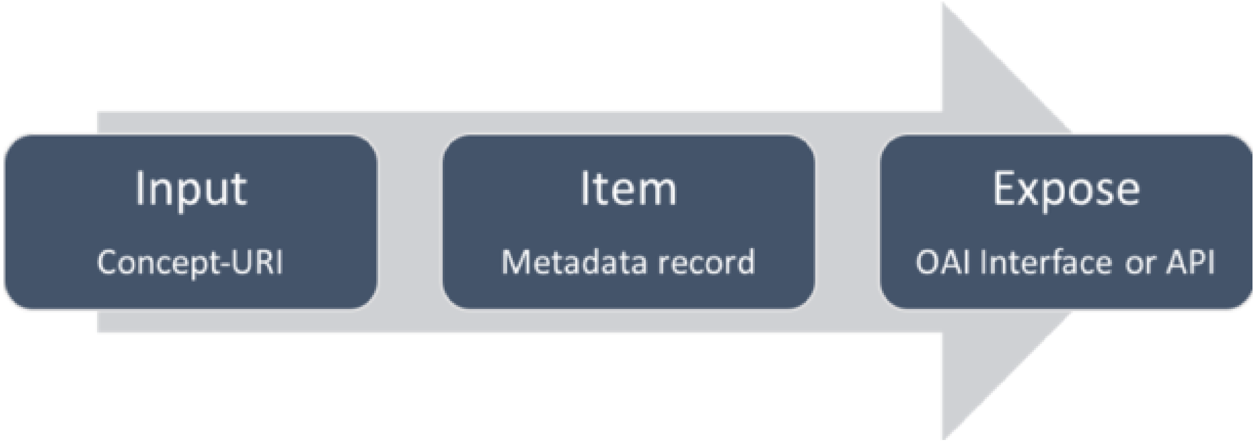


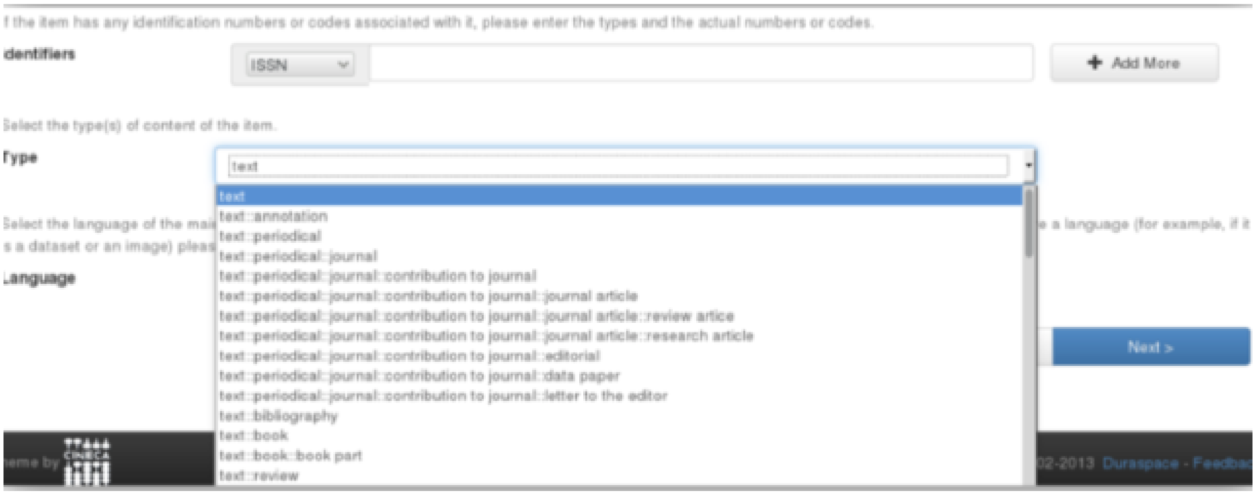
Figure 1. DSpace workflow approach

DSpace supports controlled vocabularies in search and submission process. Supported controlled vocabularies are expressed in a simple XML format (“DSpace node schema”). All information about a term is enclosed in a element. Only the expression of a hierarchical relationship is allowed through the use of the sub element. By using a simple annotation mechanism becomes possible. Hierarchical Taxonomies and Controlled Vocabularies are well explained in DuraSpace wiki pages. DSpace developer community forums may also helpful for relevant update patches.

In DSpace, you have two options to implement COAR Controlled Vocabularies. Option 1 includes a dropdown list of the fields to allow the submitter to select one option. Option 2 uses the internal controlled vocabularies functionality in DSpace.

Option 1: Dropdown

This option includes a dropdown list of the fields to allow the submitter to select one option. You may find the implementation steps below:



f the item has any identification numbers or codes associated with it, please enter the types and the actual numbers or codes.

dentifiers: ISSN

Select the type(s) of content of the item.

Type: text

Select the language of the mai... e a language (for example, if it s a dataset or an image) pleas...

Language

+ Add More
text
text::annotation
text::periodical
text::periodical::journal
text::periodical::journal::contribution to journal
text::periodical::journal::contribution to journal::journal article
text::periodical::journal::contribution to journal::journal article::review artice
text::periodical::journal::contribution to journal::journal article::research article
text::periodical::journal::contribution to journal::editorial
text::periodical::journal::contribution to journal::data paper
text::periodical::journal::contribution to journal::letter to the editor
text::bibliography
text::book
text::book::book part
text::review
Next >
eme by CINECA
02-2013 Duraspace - Feedbac

Figure 2. Option 1: Dropdown

1.Create a separate for each community per type inside the input-forms.xml file.

```
<name-map collection-handle="1111/33333" form-name="book" />
```

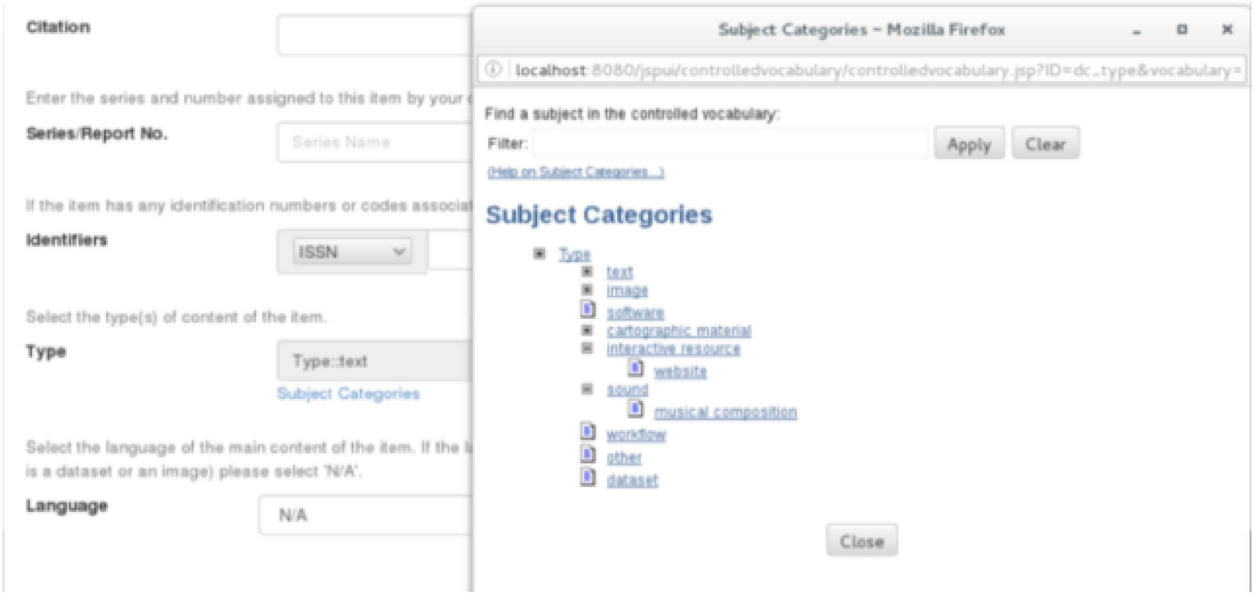
2.Example of the field in input-forms.xml

```
<field>
<dc-schema>dc</dc-schema>
<dc-element>type</dc-element>
<dc-qualifier></dc-qualifier>
<repeatable>false</repeatable>
<label>Tipo</label>
<input-type value-pairs- name="common_types">dropdown</input- type>
<hint>xyz.</hint>
<required></required>
</field>
```

3.Provide a way to show (flat term) type name instead of URL eg. Show “software” instead of > http://purl.org/coar/resource_type/c_5ce6

Option 2: Using Dspace Functionality

This option uses the internal functionality of controlled vocabularies in Dspace to present the COAR types. Note that this prototype implementation is valid Dspace version 5 and above. You may find the implementation steps below:



Citation

Enter the series and number assigned to this item by your

Series/Report No.: Series Name

If the item has any identification numbers or codes associat

Identifiers: ISSN

Select the type(s) of content of the item.

Type: Type::text

Subject Categories

Select the language of the main content of the item. If the l
is a dataset or an image) please select 'N/A'.

Language: N/A

Subject Categories – Mozilla Firefox

localhost:8080/jspui/controlledvocabulary/controlledvocabulary.jsp?ID=dc_type&vocabulary=

Find a subject in the controlled vocabulary:

Filter: Apply Clear

(Help on Subject Categories...)

Subject Categories

- Type
  - text
  - image
  - software
  - cartographic material
  - interactive resource
    - website
  - sound
    - musical composition
  - workflow
  - other
  - dataset

Close

Figure 3. Option 2: Using DSpace Functionality

1.Insert provided coar-types.xml in ${DSpace_HOME}/dspace/config/controlled-vocabularies/

2.Switch in ${DSpace_HOME}/dspace/config/input-forms.xml at dc:type field and change input type for onebox and insert in the same field the tag coar-types, this tag forces the input to be only through the plugin.

3.Example of the field in input-forms.xml:

```
<field>
<dc-schema>dc</dc-schema>
<dc-element>type</dc-element>
<dc-qualifier></dc-qualifier>
<repeatable>false</repeatable>
<label>Type</label>
<input-type>onebox</input-type>
<hint>Select the type(s) of content of the item. </hint>
<required></required>
<vocabulary closed="true"> coar-types</ vocabulary>
</field>
```

4.In dspace.cfg Edit the value to “true”: webui.controlledvocabulary.enable = true # allows the use of the controlled vocabularies add-on to allow the tree visualization mode;

5.Add the following lines:

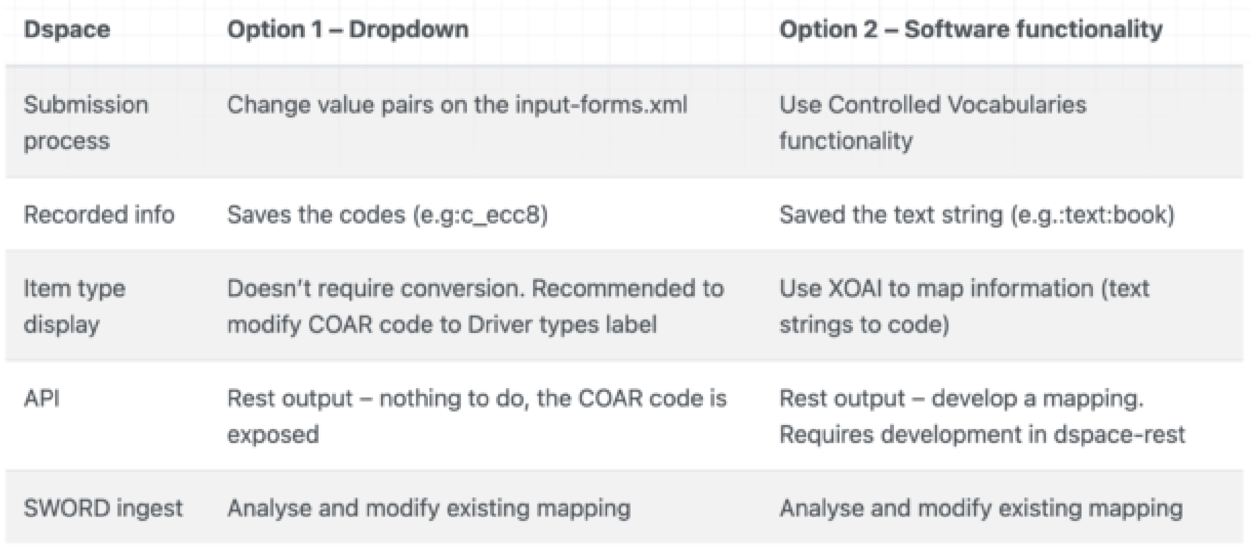
```
 choices.plugin.dc.type = coar-types # field (dc.type) to use the plugin and name of file with the vocabularies;
  choices.presentation.dc.type = select # field presentation mode;
  vocabulary.plugin.coar.hierarchy.store = true # saves in database the tree hierarchy;
  vocabulary.plugin.coar.hierarchy.suggest = true # show selected type with its hierarchy;
  vocabulary.plugin.coar.delimiter = "::" # delimiter to use between different hierarchic levels;
```

How to encode it?

Text or figure to be added Provide at least two values: 1. the concept URI – http://purl.org/coar/resource_type/c_6501 2. a label belongs to the concept

Changes required for each implementation

The following table describes the changes in different levels on DSpace regarding each implementation. Repository managers can make all changes or just a basic implementation by changing only the submission process in one of the two options.



Dspace	Option 1 – Dropdown	Option 2 – Software functionality
Submission process	Change value pairs on the input-forms.xml	Use Controlled Vocabularies functionality
Recorded info	Saves the codes (e.g:c_ecc8)	Saved the text string (e.g.:text:book)
Item type display	Doesn’t require conversion. Recommended to modify COAR code to Driver types label	Use XOAI to map information (text strings to code)
API	Rest output – nothing to do, the COAR code is exposed	Rest output – develop a mapping. Requires development in dspace-rest
SWORD ingest	Analyse and modify existing mapping	Analyse and modify existing mapping

Implementation in Samvera (including Hyrax)

Samvera (and some of its variants, including Hyrax and Haiku), is designed, ‘out of the box’ to be able to exploit the use of controlled vocabularies in both its data model and in its user interfaces for input (ingest), browsing and searching. This is facilitated by a component with the amusing name Questioning Authority (QA). There are, essentially, two ways to implement a controlled vocabulary in Samvera with QA – either to connect QA directly to a remote web service providing the vocabulary, or to copy the vocabulary into a file within the repository. The second of these two options is much simpler to configure, and the Samvera community has provided clear documentation about how to do this . It has the advantage of no reliance on any external resource once configured, but the disadvantage that the file must be manually updated if the vocabulary is revised.

The format of the file used by QA is very simple, essentially linking the URI of each term in the vocabulary to the label which will be used in the repository’s user interface. The data model in Samvera is already a linked-data graph, so the URIs in the vocabulary are used directly by the repository in metadata records.

Implementation in Eprints

Eprints community has developed a plug-in for COAR’s Resource Type Controlled Vocabulary. The documentation, file links and other details are available at the given link.

Implementation in Haplo

Install the [hres_repo_openaire] plugin. This implements metadata conversion to the OpenAIRE v4 application profile, including COAR vocabularies, and declares it as a metadata format for OAI-PMH.

While the schema can be configured through the administrative user interface, [Haplo Repository] prefers to implement schemas as plugins. These provide schema definitions and any associated metadata conversions.

A typical repository may use over 100 plugins, providing building blocks of repository functionality, and cooperating to implement the overall repository schema. The flexibility of this architecture enables a single repository to manage research data, text based and non-text based outputs, and portfolios of outputs, all within the same system.

In the case of the OpenAIRE plugin, it only needs to deliver the metadata conversion, as it can share the underlying schema with the core repository schema.

Implementation in Open Journals System

Open Journal Systems (OJS) is an open source journal management and publishing software, developed by the Public Knowledge Project (PKP).

Since 2011 an OpenAIRE plugin helps OJS journals to become compliant with the OpenAIRE infrastructure in terms of comprehensive metadata descriptions of open access articles. A new OpenAIRE plugin for OJS 3.1 was released in January 2019 which is a complete rewrite and is designed to replace the old plugin.

It adds JATS as a new XML metadata format in OAI-PMH and complements the OpenAIRE Guidelines for Literature Repository Managers v4. JATS (Article Tagging Suite) is a NISO standard and is widely used as an archiving and interchange XML format for scientific articles.

Besides adding a new endpoint to the OAI interface, the plugin implements COAR controlled vocabularies. The journal section settings are extended to let journal managers choose the proper publication type from the COAR resource type vocabulary that best describes the articles published in a specific section of a journal. COAR vocabularies are following the SKOS standard, and therefore a concept is described by a pair of URL and label. They can also be represented in custom elements in JATS metadata.

In section settings OJS-journals can choose a publication type from the COAR Resource Type vocabulary that best describes the articles published in that section.

The access right of a journal article is shown by using the COAR Access Rights Vocabulary.

Installing using a release from Github

- Download the latest compatible release from https://github.com/ojsde/openAIRE/releases. Unzip.
- Disable the old OpenAIRE plugin from Settings -> Website -> Plugins -> Generic Plugin -> OpenAIRE Plugin.
- Remove the old OpenAIRE plugin folder from plugins/generic/.
- Move the new OpenAIRE plugin folder to OJS plugins/generic/ folder.
- Go to Settings -> Website -> Plugins -> Generic Plugin -> OpenAIRE Plugin and enable the plugin.

If you are hosting a single site with several OJS journals, you can enable the plugin from the site settings. This enables the custom OAI-PMH metadata format for the whole site. However, other features, like allowing journals to define content types for articles within specific sections, requires that journals enable the plugin from their own settings.

How to use the plugin?

After enabling the plugin you will see a new metadata format called oai_openaire_jats in your OAI-PMH.

You can go to Settings > Journal > Sections and edit the section settings. The settings form has a pull down menu that allows you to select a COAR Resource Type that best describes the articles published in that section. The selected resource type will be shown in the OAI-PMH metadata.

For more about the implementation, please visit OJS-de.net Project's Github page.

Repositories that have Implemented a Controlled Vocabulary

Country	Institution	Software	Repository name	Type of implementation
Portugal	University of Minho	DSpace	Repositorium	Dropdown list
Spain	Spanish National Research Council	DSpace-CRIS	DIGITAL.CSIC	DSpace functionality
United Kingdom	Rothamsted Research	Haplo	Rothamsted Research	Out-of-the-box or plugin

COAR Confederation of Open Access Repositories

These resources were developed by COAR in 2020