EMiCdora Technical Documentation

EMiCdora

EMiCdora is a collection of Drupal modules designed to pull together the open-source software that comprises Modernist Commons into a cohesive suite of tools that can be used to create, work on, evaluate and publish digital critical editions.

These modules function as an extension of Islandora, an extensible framework designed to store, manage, manipulate, and retrieve digital assets being maintained by a Fedora Repository, as well as to handle how those actions are taken for each content type handled by Fedora.

The purpose of this document is to outline how EMiCdora extends Islandora, what kind of content types are provided by EMiCdora through Islandora, how they are stored, how they are managed, and how they relate to each other.

Installation and Dependencies

The EMiCdora Drupal module and its included submodules require the following module dependencies:

- Islandora - Provides the connection to Fedora and the ability to create context for EMiCdora content types stored in Fedora
- Islandora CWRC Writer - Provides instances of the CWRC-Writer to edit TEI-RDF content
- Islandora Image Annotation - Provides image annotation functionality for image and page sources
- Islandora XML Forms - Provides the ability to convert Drupal forms into XML metadata
- Islandora Large Image Solution Pack - Provides the image content type that can be used as a source
- Islandora Video Solution Pack - Provides the video content type that can be used as a source
- Islandora Audio Solution Pack - Provides the audio content type that can be used as a source
- Islandora Book Solution Pack - Provides the book content type that can be used as a source, and includes a dependency list of additional modules to provide functionality such as OCR generation and PDF conversion
- Islandora XACML API - Provides the ability to restrict object and datastream viewing and management to specific users or groups
- Islandora Solr Search - Provides a connection to Solr so that any available metadata can be stored; drives the display of items in the CO-OP and items related to a critical edition

Additionally, the 'modules' folder included in the EMiCdora package contains additional pieces of functionality that will need to be enabled, including:

- Collation - allows for collations to be generated and viewed from transcriptions; requires MongoDB for collation storage and retrieval.
- Version Viewer - integrates the CWRC-Writer with EMiCdora expressly for the purpose of viewing version TEI-RDF.

Some background on Fedora objects

Fedora stores content quite a bit differently than most file and operating systems. Most of these differences are invisible to the end user, but some context is useful in understanding how they work, especially in an object structure as complex as EMiCdora's.

Content in Fedora is *abstract*, *acontextual* and *asemantic*. This means that Fedora almost never understands or cares what

it contains, and everything it *does* contain starts off pretty much the same. Content inside Fedora begins as an *Abstract Object*, to which is attached *properties*, *relationships* and *datastreams*. These *properties* describe some information about the object, such as its label or owner. The *datastreams* are files attached to the objects, such as XML files describing metadata, or JPG files representing a photograph, or a text file describing the textual content of the object. Many *datastreams* can be attached to a single Fedora object, meaning that, for example, a scan of a book page can be associated to the extracted OCR from that page simply by virtue of them both being attached to the same object as *datastreams*. An additional RDF *datastream* describes *relationships*, which explain how the object relates to other objects in Fedora (e.g., *X* `isMemberOfCollection` *Y* to describe the collections *X* is contained in, or *X* `hasModel` *Z* to describe the object's content type as conforming to *Z* model). Fedora does not necessarily understand what these relationships mean; it simply knows how to tie objects together by virtue of their resource identifiers.

The purpose of Islandora, then is to provide context for the *properties*, *relationships*, and *datastreams* of Fedora objects so that they can be worked with. While Fedora may not understand what a thumbnail image or a collection or the extracted OCR of the fifth page in a book is, Islandora is set up to do just that. The process looks a bit like this:

- Islandora asks for an object out of Fedora
- Islandora asks the object what kind of content it is
- Drupal asks Islandora what the user would like to do with the object, such as viewing or editing, and reports this to Islandora
- Islandora responds with a web page tailored both to the action the user requested, as well as the type of content they requested, compiled from the *properties*, *relationships*, and *datastreams* on that object as well as related objects.

The result is a webpage generated on the fly that amalgamates *datastreams* and *properties* of an object, based on the type of request the user made and the type of object requested, bolstered by the suite of tools that comprises Modernist Commons. These manifest themselves in Modernist Commons in many ways, such as TEI *datastreams* on TEI-RDF objects being read into the CWRC-Writer, or TRANSCRIPTION *datastreams* on transcription-type objects being available for edit.

Content types in EMiCdora

EMiCdora and its associated modules define the following content types:

Type	Provided By	Description
Critical Edition Container	EMiCdora	A base content type intended as a springboard towards content asssociated with the critical edition. Defines and contains the portions of the apparatus for the content.
Version	EMiCdora	A single version of a piece of source content associated with a critical edition. Contains a consolidated copy of the version's TEI to be used for reading and evaluation.
TEI-RDF	EMiCdora	Individual pages of a version. Contains a single page of the version's TEI to be used for modification.
Transcription	EMiCdora	The transcription of an individual page of a version. Contains a copy of the page's transcription in plain text.
CWRC Schema	Islandora CWRC Writer	A schema that can be associated with TEI-RDF. Contains a copy of an RNG XML file, as well as a CSS stylesheet to drive the display of the TEI.
Collection	Islandora	A container-type object to which other objects can be associated to relate them together as members of a single collection
Person Entity	Islandora	An object representing a person. Contains XML metadata about that person.

Type	Provided By	Description
	CWRC Writer	
Organization Entity	Islandora CWRC Writer	An object representing an organization. Contains XML metadata about that organization.
Title Entity	Islandora CWRC Writer	An object representing the title of a work. Contains XML metadata about that title.
Place Entity	Islandora CWRC Writer	An object representing a geographical location. Contains XML metadata about that location.
Event Entity	Islandora CWRC Writer	An object representing an event. Contains XML metadata about that event.
Image Annotation	Islandora	An abstract representation of the annotations that can be applied to an image. Contains an OAC EAD representation of a series of annotations.
Video	Islandora	An object that understands how to store and display video content. Contains metadata about the video, as well as an uploaded video and a display version of the video.
Audio	Islandora	An object that understands how to store and display audio content. Contains metadata about the audio, as well as an uploaded audio file and a playback version of the audio.
Large Image	Islandora	An object that understands how to store and display high-definition still images. Contains metadata about the image, as well as an uploaded image and a display version of the image.
Book	Islandora	An abstract representation of a book. Contains no pages, but contains metadata about the book, and can also contain a concatenated PDF version of the book.
Page	Islandora	An object representing an individual page of a book. Contains an uploaded image of the page, as well as display versions of the page, and possible extracted OCR and PDF copies of the page.

Structure of content types used by EMiCdora

Critical editions reference a source object of some kind. These source objects are contained inside the CO-OP collection like so, using RDF `isMemberOfCollection` relationships:

> EMiC CO-OP Collection
>
> > Video Objects
> >
> > Audio Objects
> >
> > Large Image Objects
> >
> > Book Objects
> >
> > > Book Pages
> >
> > ____ Entity Collection (all the varieties described above)
> >
> > > ____ Entity

From the workbench, a Critical Edition Container can be created to work with sources in the CO-OP (or even add new sources to the CO-OP that are automatically associated with the critical edition). This structure looks like so, with the verb representing the RDF relationship in parentheses:

> Critical Edition Container (`isMemberOfCollection` EMiC Editions Collection)
>
> > Source object associated from the CO-OP collection (`isMemberOfCollection` CO-OP, `isMemberOf` Critical Edition Container)
> >
> > > Version minted from the source (`isMemberOf` Critical Edition Container, `isDerivationOf` Version)
> > >
> > > > TEI-RDF representation of an individual page of the source; videos, Audio and images only have one of these, under the version itself as opposed to the pages (`isMemberOf` Version, `isDerivationOf` the page or source, `isPageOf` Version, `hasSchema` Schema object; also contains relationships to entities directly inside the TEI-RDF content)
> > > >
> > > > > Transcription of the TEI-RDF (`isMemberOf` Version)
> > > > >
> > > > > CWRC Schema to be used with the TEI-RDF (`isMemberOf` CWRC Schema Collection)
> > > > >
> > > > > Entities of different kinds (`isMemberOfCollection` its respective entity collection)

Metadata stored by EMiCdora

EMiCdora stores four types of XML metadata:

Type	Description	Stored with?	Indexed in Solr?
RDF	Describes how each object relates to others	Objects of every type	Yes
TEI-RDF	Describes how a version of a source is marked up	Version objects or their pages	No
OAC EAD	Describes how an image is marked up	Potentially any type; however, it is enabled on pages and image sources by default	No
MODS	Stores general information about an object that has been input from a form	Critical edition containers and sources	Yes

The metadata stored is used in two ways:

- It is stored directly on an object to provide information for display
- It is indexed and stored in Solr to run searches and provide contextual lists of objects, filtered or otherwise

Metadata indexed into Solr

The Solr configuration used on the EMiCdora site automatically generates Solr field names based on the structure of XML. These fields are then stored multiple times in Solr using different field types. Solr field names use the following conventions:

RDF field names

- Prefix: `RELS_EXT_` (relationships to other objects) or `RELS_INT_` (relationships to different parts of the object itself)
- Middle: relationship name, generally camel-cased (e.g., `isMemberOfCollection_`), possibly with resource location

- RELS type (`literal_` , `uri_` , etc.)
- Suffix: Solr type

Example: `RELS_EXT_isMemberOfCollection_uri_ms` , indexing any values of the RELS-EXT isMemberOfCollection URI for the object in question.

MODS field names

- Prefix: `mods_`
- Middle: strung together hierarchy of MODS elements and attributes separated by underscores
- Suffix: Solr type

Example: `mods_name_personal_namePart_ss` , indexing a sortable version of any `mods/name[@type="personal"]/namePart` fields found on an object's MODS.

Common Solr suffix types for string values

- `_s` : single string; stores an individual index for a field; best used to drive the display of labels and other text content.
- `_ms` : multivalued string; stores any content that can be found for this particular field; best for providing search results, and cannot be used for faceting.
- `_ss` : sortable string; stores string content prepared for alphanumerical sorting; best used for sort parameters.

Using the CO-OP

The CO-OP can be accessed by navigating to http//modernistcommons.ca/co-op.

The CO-OP is a special collection that houses source content to be made available to critical edition containers. Video, Audio, Image and Book content types can be added to this collection from several places:

- From the CO-OP itself
- From the Workbench
- From a Critical Edition

Items added to the CO-OP can be chosen to be added to the currently-logged-in user's Workbench.

Items in the CO-OP can be searched for by authors (Solr field: `mods_name_personal_namePart_ms`) or genres (Solr field: `mods_genre_s`). Care should be taken that forms created providing metadata for items in the CO-OP also provide these two Solr fields, or they will not show up in the CO-OP. Authors don't necessarily have to be tied to person entities; these are instead authors that were included in the form that was originally filled out when adding the source to the CO-OP. These fields are stored in Solr, as the CO-OP navigation is driven by entries in the Solr index.

After clicking on an author or genre, users are directed to a set of Solr search results for items in the CO-OP collection filtered by that author or genre. Two of the Islandora Solr Search blocks are configured for use, by default, and can be seen on these search results pages along the right-hand side:

- *Islandora Sort* - Defines the sort order for search results. Re-titled as *Sort*. Search results can be sorted by Title (Solr field: `fsg_label_ss`), Genre (Solr field: `mods_genre_ss`), or Type (Solr field: `mods_typeOfResource_ss`).
- *Islandora Facets* - Defines filter criteria that can be used to narrow down result sets. Re-titled as *Refine by*. Search results can be faceted by Title (Solr field: `fgs_label_s`) or Creator (`mods_name_personal_namePart_ms`).

These blocks can be configured at http://modernistcommons.ca/admin/islandora/search/islandora_solr/settings, in the *Sort settings* and *Facet settings* sections.

The form for adding items to the CO-OP is split into two parts; the top half allows for the upload of files to attach to the

item, and the bottom half allows metadata to be filled out. File upload is required for all content types except for text, which defines no files for upload itself, but rather requires files to be uploaded for individual pages.

Using the Workbench

The Workbench is where objects owned by the currently-logged-in user are aggregated, and where more can be added. The workbench supports the addition of the following content types:

- Critical Edition Containers
- Person, Organization, Title and Place entities
- Sources of all types
- Schemas to be used with TEI-RDF

Critical editions and sources that have been added to the workbench can removed using the *Remove* button.

Content that is added from the workbench is added to the user's workbench automatically.

Restricting object access

Restriction of objects is done via XACML policies attached to them. When a POLICY XML datastream is found attached to an object, Islandora automatically handles the processing of access restrictions contained in that XML.

A predefined set of XACML restrictions are added by checking off the *Keep object private?* box when filling out the object upload or critical edition container addition forms. The applied policy is a whitelist; it allows the user who created the object to view and manage it, whereas any other non-administrator may not.

An object's restrictions can be fine-tuned by clicking *Object Policy* in the *Operations* context menu when viewing an object. This includes removing existing restriction policies, or adding more users and groups to the whitelist. These context menus appear on, and can be used to restrict, sources, versions and critical edition containers.

Working with a Critical Edition Container

Critical edition containers are units for handling all the different associations and relationships a critical edition may have.

Critical edition containers house:

- Sources (as a related object)
- Versions (as a related object)
- Collations (as a database definition of linked transcriptions)
- An Apparatus (as datastreams on the container itself)

Blocks and Context Menus

The left-hand side of a critical edition container displays several blocks that assist in navigation around different parts of the container object. Users with permission to manage the container object can also access a context menu for each block, which appears as a gear icon when mousing over the top-right corner of the block; clicking on it brings up a suite of actions that can be taken for the part of the container described by this particular block.

The bottom context menu, *Operations*, defines a set of actions that can be taken for the object currently being viewed. Because of the nodular nature of RDF objects, this context menu must be accessed *with context*, meaning that it is only useful for the object currently being worked with. Since using an RDF object relationship structure means that objects like versions or sources can be related to multiple parent or child collections, containers, or other structural elements, it is

neither possible nor useful to try to provide operations for all potential objects with possible relationships to a critical edition container.

EMiCdora defines the following contextual blocks:

Name	Links provisioned	Appears On
Critical Apparatus	Apparatus datastream content	All types
Versions	Filtered sets of versions associated with the current container	All types
Collations	Any collations created from versions associated with the current container	All types
Transcriptions	Transcriptions associated with the current version	Version objects
Source Material	Filtered sets of sources associated with the current container	All types
Operations	No links (only provides a context menu for operations)	All types

The Apparatus

An apparatus is comprised of the following datastreams attached to a critical edition container:

- ACKNOWLEDGEMENTS
- AFTERWORD
- APPENDIX
- BIBLIOGRAPHY
- CHRONOLOGY
- EDITORIAL_PROCEDURES
- EPIGRAPH
- EPILOGUE
- FOREWORD
- ILLUSTRATIONS
- INTRODUCTION
- PREFACE

These datastreams must be `text/plain` files.

Individual apparatus items should be edited by clicking the context menu on the *Apparatus* block.

Sources

Sources are added to critical edition containers. They can be added from the context menu on the *Source Material* block. New sources of each type can be added using this context menu, or existing sources can be searched for. New sources will automatically be added to the container they are added from.

When adding an existing source, Solr is used to populate the list of potential sources to add. Sources are gathered from the CO-OP, and can be filtered by Title (Solr field: `fgs_label_s`) or Creator (`mods_name_personal_namePart_ms`). Multiple sources can be added to a container.

Versions

While versions can be viewed from any place inside a critical edition container, they can only be created from the context of an individual source.

To add a new version, click *Add new version* in the *Operations* context menu when viewing a source from the context of a critical edition container.

The version addition page contains a form that defines how the TEI-RDF and version transcription are created. A TEI-RDF object can be automatically generated from OCR datastreams if they exist on pages of a text object, or a blank one can be created for other source types. The transcription can then be automatically generated from the TEI-RDF, or uploaded from a file, or born-digital from a textfield on the form.

Existing versions associated with the source but not in the current container can be added by clicking *Link existing version* in the same *Operations* context menu. Any available existing versions that fit that criteria will be listed on the resulting page.

Editing TEI-RDF using the CWRC-Writer

TEI-RDF can be edited by clicking "Edit TEI-RDF" in the *Operations* context menu when viewing a version from the context of a critical edition.

The CWRC-Writer is loaded with the CWRC datastream of the TEI-RDF object related to the version, or the TEI-RDF object related to the first page of the version. Changes made in the CWRC-Writer are saved as new versions of the CWRC datastream.

The CWRC-Writer loads the CWRC datastream using the default schema. This schema can be changed using the CWRC-Writer menu, and saving the document writes that schema change back to the TEI-RDF object.

Transcriptions

Transcriptions can be managed from the version they are associated with. They can be edited or deleted from the *Operations* context menu when viewing a transcription from the context of a critical edition container.

Transcriptions are not updated when their related version's TEI-RDF is updated. They can be updated by selecting *Refresh transcription from Edited TEI* from that same context menu.

Version and Source Links

The *Versions* and *Source Material* blocks contain links to filtered Solr searches for versions and sources unique to each critical edition container. By default, a new critical edition container comes with an empty filter linking to 'all' versions or sources. By clicking *Configure Links* in the context menu for *Versions* or *Source Material*, new links can be generated that connect to filtered sets of search results.

For example, a link to all textual sources of a container might be added as a new source category called "Textual Sources". After clicking "Configure", a new field could be added, `mods_typeOfResource_ms` , with the value of "text". Saving this field configuration would provide a second link in the *Source Material* block linking to search results for any sources that are part of this critical edition container that have a "text" resource type.

Collations

Collations are created by clicking *Add New Collation* from the context menu for *Collations*. These can be generated from a selection of multiple transcriptions associated with version objects within the current container.

Clicking a collation loads the associated collation into the collation viewer.

Working with data inside the container

By design, work saved to one content type in a critical edition container is *NOT* propagated to other objects related to it; this prevents quashing work done on one type of data with work done on another type related to it, as well as preventing

overwriting other users' derivations of that object.

The following content types can be stored, modified, saved, and possibly derived from other objects:

Data Type	Associated object	Storage datastream ID	Modified by	Can be derived from
Apparatus	Critical edition container	Several, listed above	Text-editing form accessible from the *Critical Apparatus* context menu	N/A
OCR	Page source	OCR	N/A	A version's TEI-RDF, accessible from a version *Operations* menu
TEI-RDF	TEI-RDF object	CWRC	CWRC-Writer, accessible from a version operations menu	Transcriptions when the version is initially created, but cannot be re-derived
Consolidated TEI-RDF	Version object	TEI-RDF	N/A	TEI-RDF objects associated with a version, accessible from the *Operations* menu for that version
Transcription	Transcription object	TRANSCRIPTION	Text-editing form accessible from the *Operations* menu for that transcription	Consolidated TEI-RDF from the associated version, accessible from the transcription *Operations* menu
Collation	Critical edition container	N/A; stored as Austese collation data in MongoDB	Collation editor, accessible by clicking the link for a collation	Transcriptions originally used to create the collation, accessible from the *Operations* menu for that collation