

DCFUG Training - 2 Migrating from Fedora 3 to Fedora 4

These training archives may be out of date, but have been retained and kept available for the community's benefit in reviewing previous sessions.

 Current training documentation can be found here: [Training](#)

- [Learning Outcomes](#)
- [Differences Between Fedora 3 and Fedora 4](#)
 - [XML Objects vs. Resources](#)
 - [Flat vs. Hierarchy](#)
 - [File System Differences](#)
 - [PIDs vs. Path](#)
- [Data Modeling](#)
 - [Object Properties](#)
 - [Datastream Properties](#)
 - [Fedora 3 Data Models](#)
 - [Fedora 4 Data Models](#)
 - [Portland Common Data Model](#)
 - [Islandora Data Model](#)
 - [UNSW Data Model](#)
- [Data Migration Tool](#)
 - [Motivations](#)
 - [Proposal](#)
- [Enhancements](#)
 - [Taking Advantage of Properties](#)
 - [Enhancing Your Metadata](#)

Learning Outcomes

- Understand the main differences between Fedora 3 and Fedora 4
- Learn about the current state of migration tools and planning in the Fedora community
- Explore new possibilities for enhancing data in Fedora 4

Differences Between Fedora 3 and Fedora 4

XML Objects vs. Resources

Fedora 3	Fedora 4
FOXML objects	Web resources
Inline and managed XML	RDF properties*

**XML Datastreams are still supported*

Flat vs. Hierarchy

Fedora 3	Fedora 4
Objects and Datastreams at the root level	Resources in a hierarchy
No inherent hierarchy	All resources descend from a root resource

File System Differences

Fedora 3	Fedora 4
Objects directory and data streams directory	Containers stored in a database
Objects and datastreams stored in a PairTree	Binaries stored in a PairTree

PIDs vs. Path

Fedora 3	Fedora 4
----------	----------

Objects have Persistent Identifiers (PIDs)	Objects have a path (including a UUID) based on their location in the file system hierarchy
	Objects can also have other identifiers (DOIs, Handles, PIDs, etc.)

Data Modeling

Object Properties

	Fedora 3	Fedora 4	Example	Notes
PID	PID	dc:identifier	someprefix:1234	Fedora 3 Legacy PID
State	state	fedora:status	active	The default values are active and deleted -- but additional values can be created
Label	label	dc:title	Some title	
Created Date	createdDate	fedora:created	2014-01-20T04:34:26.331Z	Automatically added by Fedora 4
Last Modified Date	lastModifiedDate	fedora:lastModified	2014-01-20T05:39:08.601Z	Automatically added by Fedora 4
Owner	ownerId	fedora:createdBy	Chuck Norris	

Datastream Properties

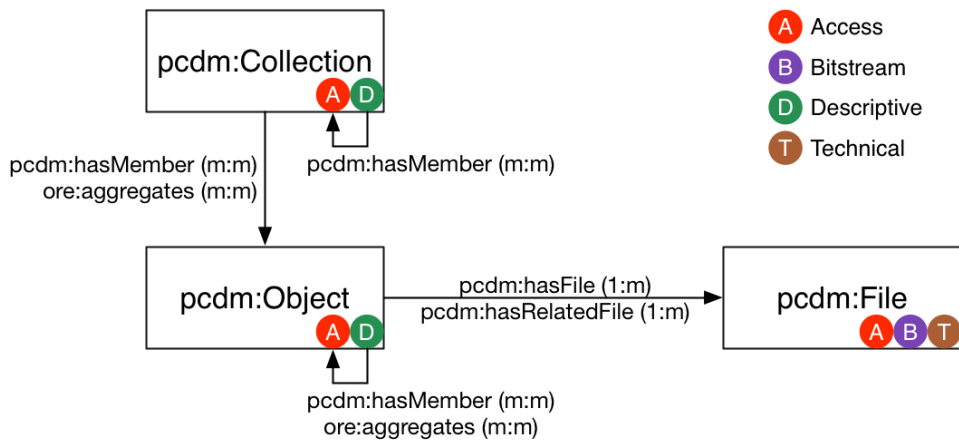
	Fedora 3	Fedora 4	Example	Note
DSID	ID	dc:identifier	MODS	Fedora 3 Legacy DSID. May not need to be preserved
State	state	fedora:status	active	The default values are active and deleted -- but additional values can be created
Versionable	VERSIONABLE	fedora:hasVersions	true	Versions are supported in Fedora 4
Label	LABEL	dc:title	MODS Metadata	
Creation Date	CREATED	fedora:created	2014-01-20T04:34:26.331Z	Automatically added by Fedora 4
Last Modified Date	N/A	fedora:lastModified	2014-01-20T05:39:08.601Z	Automatically added by Fedora 4
Mime Type	MIMETYPE	fedora:mimeType	text/xml	Automatically added by Fedora 4
Size	SIZE	premis:hasSize	50000	Automatically added by Fedora 4
Alternate ID	AltIds	premis:hasOriginalName	sample_file.pdf	Automatically added by Fedora 4

Fedora 3 Data Models

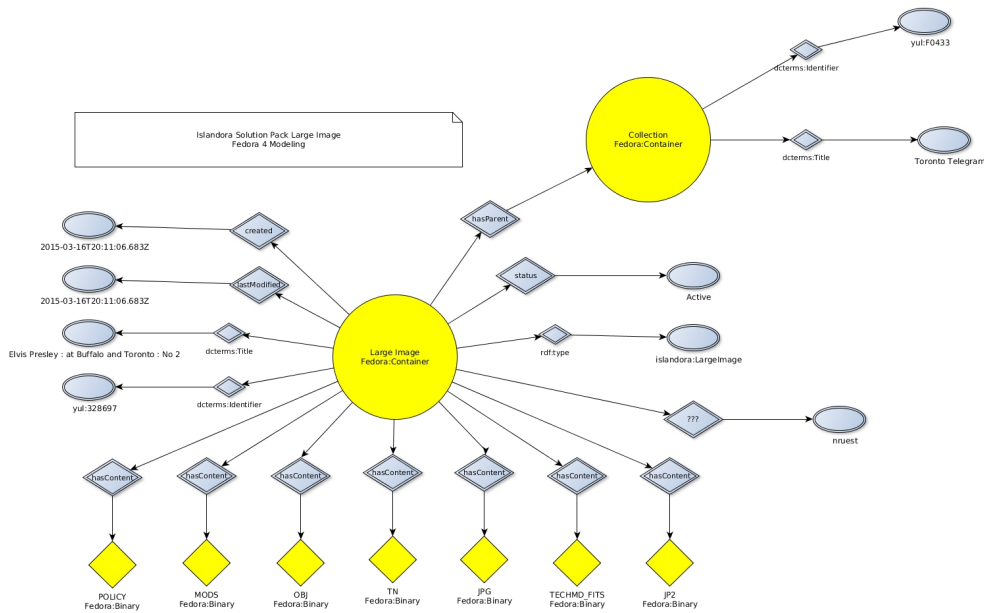
- Hydra
- Islandora
- Custom

Fedora 4 Data Models

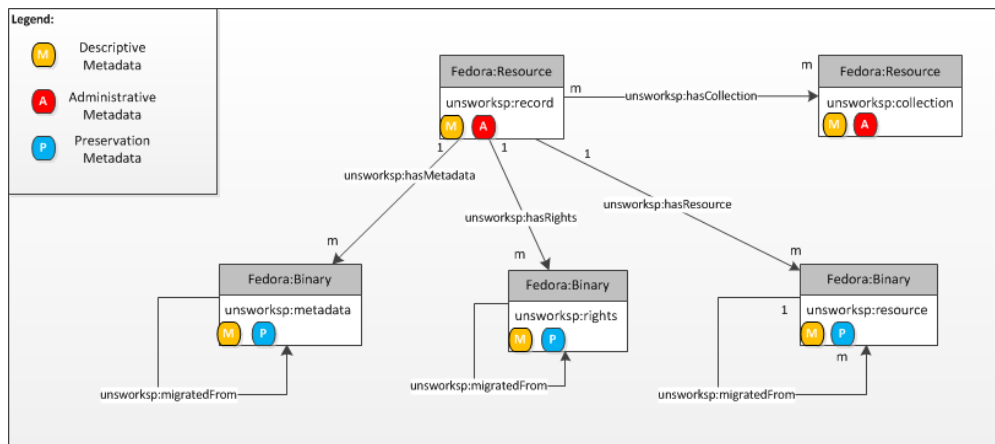
Portland Common Data Model



Islandora Data Model



UNSW Data Model



Data Migration Tool

Motivations

- need to **preserve** fedora 3 content, history and audit trail
- ability to leverage fedora 4 features
- need to make data **accessible and functional** in the new environment
- desire to make migration easier, **faster** and less error-prone

Proposal

- Process FOXML and convert to Fedora 4 resources
 - foxml (when exported in the "archive" context, or persisted in the low level store) is a complete representation of the object
 - foxml offers a wide range of compatibility with various versions of Fedora
 - foxml migration doesn't require the fedora 3 repository software to be running
 - large number of existing frameworks for efficiently processing XML
- Considerations
 - migration of data that's not in the repository (like configuration, global xacml policies, etc.) will require special handling
 - ability to write and use plugins (special configurations) for mapping complex metadata or fedora 3 constructs into fedora 4 must be made as easy as possible since most institutions will need to write their own or adapt existing ones
- Process
 1. Read and process FOXML documents
 2. Migrate PIDs
 3. Convert inline XML to managed XML or RDF properties
 4. Convert datastreams to binaries or RDF properties
 5. Convert or map access controls to Fedora 4
 6. Migrate versions

Enhancements

Taking Advantage of Properties

- Converting Inline XML and/or XML Datastreams (e.g. RELS-EXT, RELS-INT) to RDF properties.
 - Inline XML is no longer supported.
- Lightweight compared to XML.
- New possibilities for complex queries that extend beyond the limits of the repository.
 - Linked data relationships can be exposed via a standardized HTTP requests
 - Web applications can take advantage of these standardized representations.
 - Data can be shared and manipulated in new and interesting ways.

Enhancing Your Metadata

- XML metadata datastreams are still supported, but there are new opportunities to explore!
- XML metadata can be converted into RDF metadata using an RDF-based schema.
- RDF metadata is easier to query and share.
- Take advantage of linked data by pointing to authority URIs.