Migrating from Fedora 3 to 4

David Wilcox, DuraSpace
Learning Outcomes

- Understand the main differences between Fedora 3 and 4
- Learn about the current state of migration tools and data modeling in the Fedora community
- Explore new possibilities for enhancing data in Fedora 4
Differences between Fedora 3 and Fedora 4
Objects, Datastreams, & Resources

● Fedora 3
  ○ FOXML objects
  ○ Inline XML and XML datastreams

● Fedora 4
  ○ Web resources (containers & binaries)
  ○ RDF properties and XML binaries
Flat vs. Hierarchy

- **Fedora 3**
  - Objects and datastreams at the top level
  - No inherent tree structure

- **Fedora 4**
  - Containers and binaries in a hierarchy
  - All resources descend from a root resource
File System

- **Fedora 3**
  - Objects directory and datastreams directory
  - Both objects and datastreams are in a PairTree

- **Fedora 4**
  - Containers directory and binaries directory
  - Containers in a database (e.g. LevelDB)
  - Datastreams in a PairTree
PID vs. Path

- **Fedora 3**
  - Objects have Persistent Identifiers (PID)
  - An object’s PID can never be altered

- **Fedora 4**
  - Resources have an internal UUID
  - Resources have a repository path
    - This can be user-defined or generated via a PID-minter
Data Modeling
# Mapping properties - objects

<table>
<thead>
<tr>
<th></th>
<th>Fedora 3</th>
<th>Fedora 4</th>
<th>Example</th>
</tr>
</thead>
<tbody>
<tr>
<td>PID</td>
<td>PID</td>
<td>dcterms:identifier</td>
<td>prefix:1234</td>
</tr>
<tr>
<td>State</td>
<td>state</td>
<td>fedora:status</td>
<td>active</td>
</tr>
<tr>
<td>Label</td>
<td>label</td>
<td>dcterms:title</td>
<td>Some Title</td>
</tr>
<tr>
<td>Created Date</td>
<td>createdDate</td>
<td>fedora:created</td>
<td>2014-01-20T04:34:26.331Z</td>
</tr>
<tr>
<td>Modified Date</td>
<td>lastModifiday</td>
<td>fedora:lastModified</td>
<td>2014-01-20T04:34:26.331Z</td>
</tr>
<tr>
<td>Owner</td>
<td>ownerID</td>
<td>fedora:createdBy</td>
<td>Chuck Norris</td>
</tr>
</tbody>
</table>
# Mapping properties - datastreams

<table>
<thead>
<tr>
<th></th>
<th>Fedora 3</th>
<th>Fedora 4</th>
<th>Example</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>DSID</strong></td>
<td>ID</td>
<td>dcterms:identifier</td>
<td>prefix:1234</td>
</tr>
<tr>
<td><strong>State</strong></td>
<td>state</td>
<td>fedora:status</td>
<td>active</td>
</tr>
<tr>
<td><strong>Versionable</strong></td>
<td>VERSIONABLE</td>
<td>fedora:hasVersions</td>
<td>true</td>
</tr>
<tr>
<td><strong>Label</strong></td>
<td>LABEL</td>
<td>dcterms:title</td>
<td>Some Title</td>
</tr>
<tr>
<td><strong>Created Date</strong></td>
<td>CREATED</td>
<td>fedora:created</td>
<td>2014-01-20T04:34:26.331Z</td>
</tr>
<tr>
<td><strong>Modified Date</strong></td>
<td>N/A</td>
<td>fedora:lastModified</td>
<td>2014-01-20T04:34:26.331Z</td>
</tr>
<tr>
<td><strong>Mimetype</strong></td>
<td>MIMETYPE</td>
<td>fedora:mimeType</td>
<td>text/xml</td>
</tr>
<tr>
<td><strong>Size</strong></td>
<td>SIZE</td>
<td>premis:hasSize</td>
<td>50000</td>
</tr>
</tbody>
</table>
PCDM - Examples

- Book
- Disk Image
- Geo-data
- Postcard
- Sufia

https://wiki.duraspace.org/display/FF/PCDM+Mappings+-+Reference+Diagrams+for+Comment
Portland Common Data Model (PCDM)
Data Migration Tools
Motivations

- Preserve Fedora 3 content, history and audit log
- Leverage Fedora 4 features
- Make data accessible and functional in the new environment
- Make migration easier, faster and less error-prone
Initiatives

- Fedora-based “migration-utils”
- Hydra-based “fedora-migrate”
migration-utils - philosophy

- FOXML 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
migration-utils - considerations

- Migration of non-repo data (configuration, global XACML policies, etc.) will need special handling
- Writing and using plugins for mapping complex metadata must be easy
migration-utils - 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

- Lightweight and granular compared to XML
- Inline XML is no longer applicable
- Converting Inline XML and/or XML datastreams (e.g. RELS-EXT, RELS-INT) to RDF properties
New Query Possibilities

- New possibilities for complex queries that extend beyond the limits of the repository
  - Linked data relationships can be exposed via a standardized SPARQL-Query
  - Web applications can take advantage of these standardized representations
  - Data can be shared and queried 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