DCFUG Training - 1 Introduction and Feature Tour

These training archives may be out of date, but have been retained and kept available for the community's benefit in reviewing previous sessions.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

- Understand the purpose of a repository
- Learn what Fedora can do for you
- Understand the key capabilities of the software

Course Outline

Introduction to Fedora 4

What is a Repository?

- Secure software that stores, preserves, and provides access to digital materials
- Supports complex semantic relationships between objects both within and outside the repository
- Supports millions of objects, both large and small
- Capable of interoperating with other applications and services

Fedora 4 Guiding Principles

- Improved performance, enhanced vertical and horizontal scalability
- More flexible storage options
- Features to accommodate research data management
- Better capabilities for participating in the world of linked open data
- An improved platform for developers—one that is easier to work with and which will attract a larger core of developers.

Exposing and Connecting Content with Fedora 4

- Flexible, extensible object modelling
- Atomic objects with semantic connections using standard ontologies

- RDF-based metadata using Linked Data
- RESTful API with native RDF response format

New Vocabulary

Fedora 3	Fedora 4
Objects and datastreams	Resources
Object	Container
Datastream	Binary

Core Components

Durable Storage

One of the core components of Fedora 4 is its long-term storage and preservation capability. A number of features support this capability; they have been grouped here under the notion of Durable Storage.

Fixity

- Over time, digital objects can become corrupt and unusable by suffering from bit rot and other digital preservation dangers
- Fixity checks help preserve digital objects by verifying their integrity using techniques such as checksumming
- On content ingest, Fedora can verify a user-provided checksum against the calculated value
- A checksum can be recalculated and compared at any time via a REST-API request

Backup and Restore

- A full backup, including all binaries as well as a compact serialization of all containers, can be performed at any time
- A full restore from a repository backup can be performed at any time

Export and Import

- A specific Fedora resource, its children resources, and associated binaries can be exported
    - The serialization of the Fedora resource is more portable than the compact form found in the backup/restore feature
    - Exported resources are serialized in a standard JCR/XML format
- An exported resource or hierarchy of resources can be imported at any time

Versioning

- Versions can be created across the entire repository or on particular API calls.
- A previous version can be restored via the REST-API.

Data Modelling

Resources

- Both containers and binaries are represented as resources.
- Container nodes can have both containers and binaries as children.
- The tree structure allows for inheritance of things like security policies.

Properties

- Resources have a number of properties, which are expressed as RDF triples.
    - The resource itself is the implicit subject of each triple.
- Properties can be RDF literals (e.g. dc:title) or they can express relationships both internal and external to the repository.
- Any number of RDF namespaces can be defined and used.

Content Models

- Content can be modelled using RDF properties
- Properties can have either literals or URIs as their objects
- Resources can have any number of properties using any number of namespaces

Linked Data

- Fedora 4 is compliant with the LDP 1.0 spec.
- Metadata can be represented as RDF triples that point to resources outside the repository.
- Many possibilities for exposing, importing, sharing resources with other web applications.

User Interface

Administrative Console

Tour of the HTML administrative interface.

External Components

Camel

Indexing

- Indexing repository content for external applications can be accomplished by using the Camel component.
- The Camel component receives JMS messages on repository updates and relays these messages to one or more external applications.
- Repository content can be optionally assigned the rdf:type property "Indexable" in order to be filtered from non-indexable content (any predicate will do)

Triplestore

- An external triplestore can be used to index the RDF triples of content managed by Fedora.
- Any triplestore that supports SPARQL-update can be used; Fuseki and Sesame have been tested.

External Search

- An external search application can be used to perform more complex search queries on repository content.
- Any search application that supports JSON-based updates can be used; Solr has been tested.

Authorization

- Authentication (not to be confused with authorization) is assumed to take place in a layer above the application.
- The authorization framework provides a plug-in point within the repository that calls out to an optional authorization enforcement module.
- Currently, two authorization implementations exist.

Role-based Authorization

- Basic authorization compares the user's role(s) with an Access Control List (ACL) defined on a Fedora resource.
- ACLs can be inherited; if a given resource does not have an associated ACL, Fedora will examine parent resources until it finds one.

XACML Authorization

- XACML policies can provide much more complex and granular authorization.
- A default policy must be defined for the repository, and each resource can override the default with another policy.
- A XACML policy referenced by a resource will also apply to all the resource's children, unless they define their own XACML policies that override the parent policy.

Audit Service

- Maintains a history of events for each repository resource
- Both internal repository events and events from exterior sources can be recorded
- Uses the existing event system, Camel component, and external triplestore

OAI Provider

- Implements Open Archives Protocol Version 2.0 using Fedora 4 as the backend.
- Exposes an endpoint which accepts OAI conforming HTTP requests.
- Supports oai_dc out if the box, but users are able to add their own metadata format definitions to oai.xml.

Performance

Transactions

- Multiple actions can be bundled together into a single repository event (transaction).
- Transactions offer performance benefits by cutting down on the number of times data is written to the repository filesystem (which tends to be the slowest action).

Clustering

- Two or more Fedora instances can be configured to work together in a cluster.
- Fedora 4 currently supports clustering for high-availability use cases.

- A load balancer can be setup in front of two or more Fedora instances to evenly distribute read requests across each instance.
- If one Fedora instance in the cluster goes down, read requests can be directed to the other instance.
- Ingests are replicated across all instances in the cluster.