

Introduction to Fedora 4 Features

Learning Outcomes

Understand the purpose of a Fedora repository

Understand the core features of the software

What is a Fedora Repository?

Secure software that stores, preserves, and provides access to digital materials

Supports complex semantic relationships between objects inside and outside the repository

Supports millions of objects, both large and small

Capable of interoperating with other applications and services

Exposing and Connecting Content

Flexible, extensible content modeling

Atomic resources with semantic connections
using standard ontologies

RDF-based metadata using Linked Data

RESTful API with native RDF response format

Fedora 4 Project Goals

Improved performance

Flexible storage options

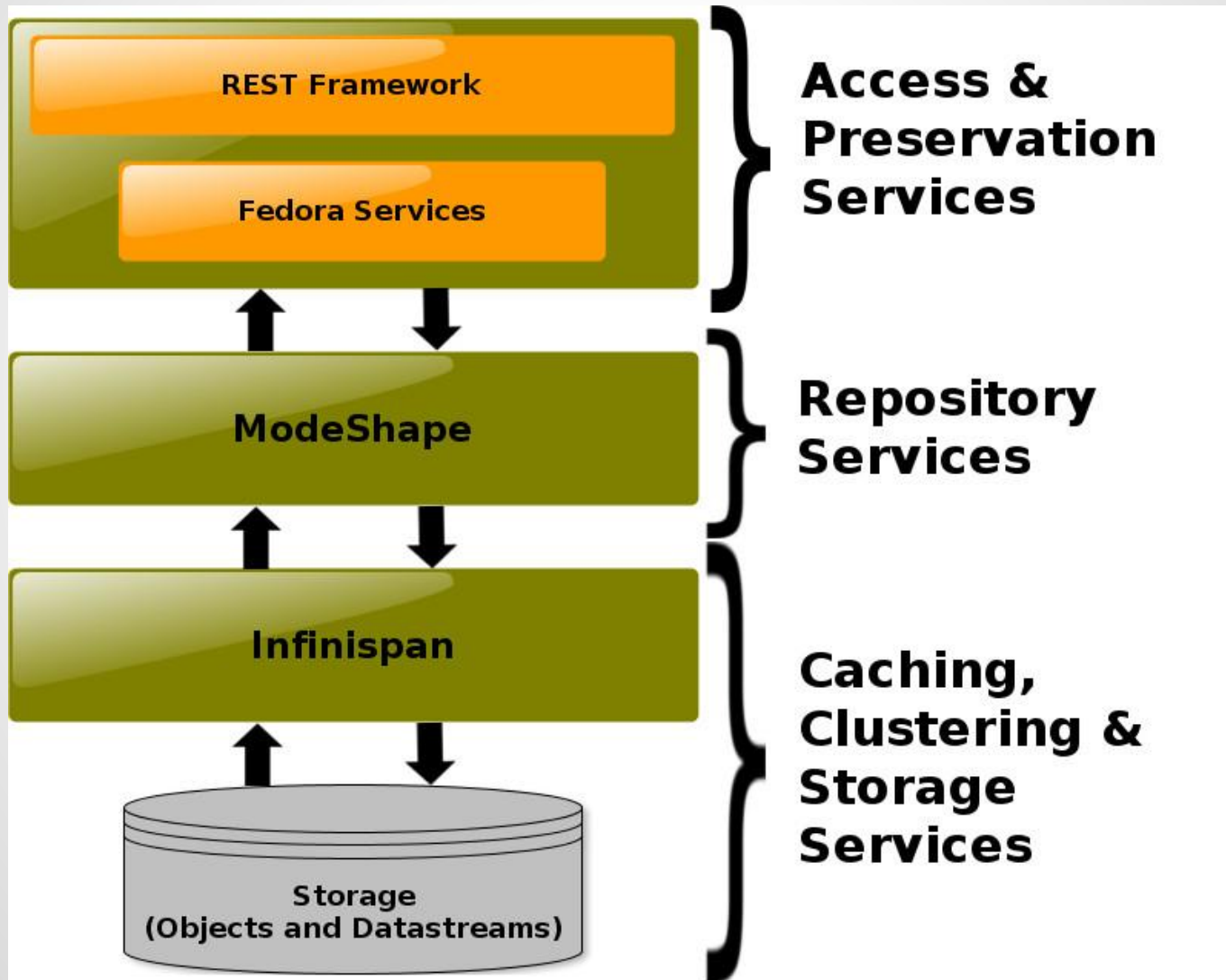
Research data management

Linked open data support

Improved platform for developers

Core Features

Component Stack



Standards

Focus on existing standards

Fewer customizations to maintain

Opportunities to participate in related communities

Core Features and Standards

CRUD - *Linked Data Platform (LDP)*

Versioning - *Memento?*

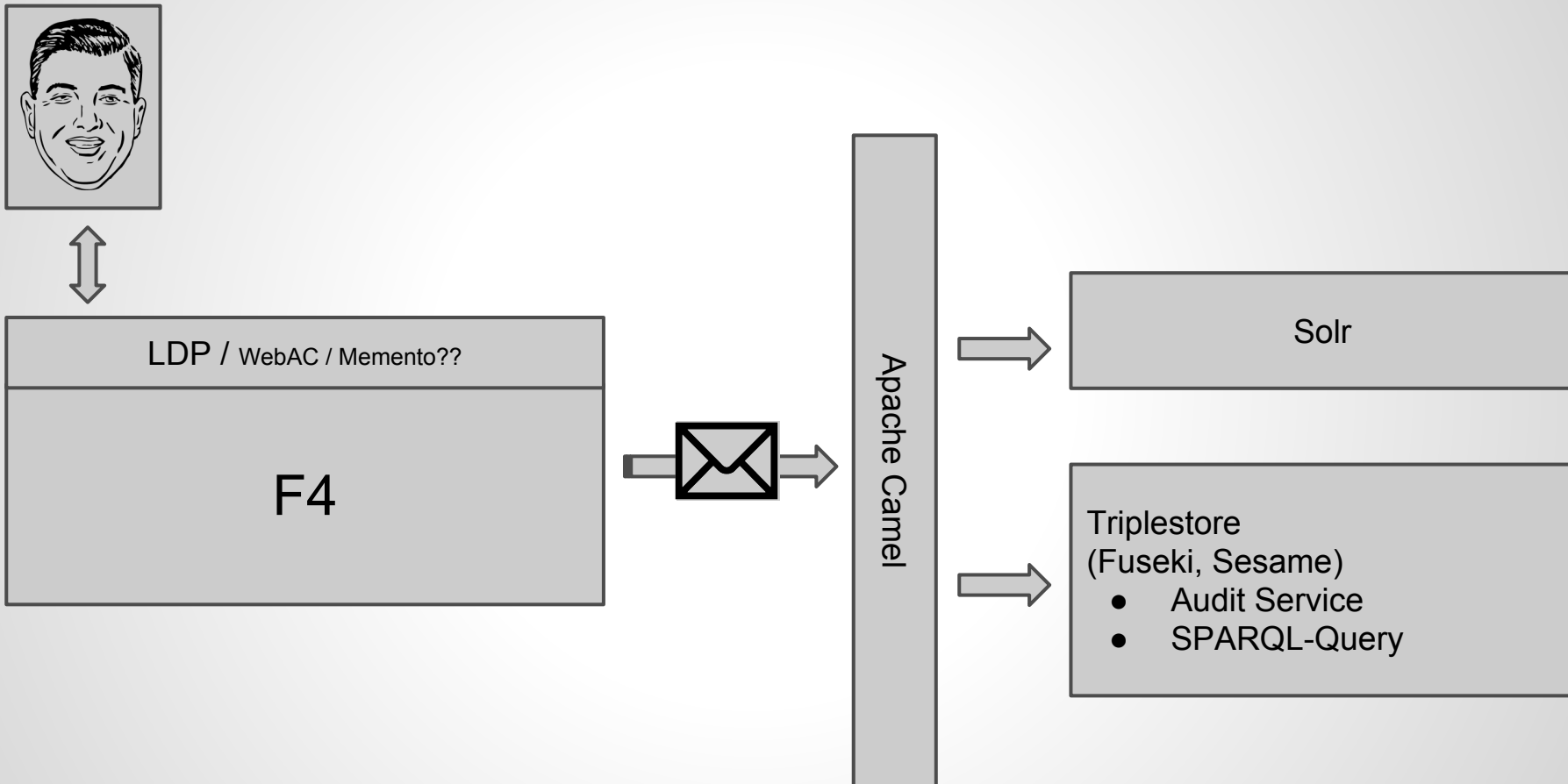
Authorization - *WebAC*

Transactions - ??

Fixity - *<http://tools.ietf.org/html/rfc3230#section-4.3.2>*

Hands-on: CRUD

Fedora Vagrant Components



Create a “cover” Container

PUT vs. POST

...Note: names in demo are only for readability

Make “cover” a pcdm:Object

```
PREFIX pcdm: <http://pcdm.org/models#>
```

```
INSERT {
```

```
  <http://localhost:8080/fcrepo/rest/cover>
```

```
  rdf:type
```

```
  pcdm:Object
```

```
}
```

```
WHERE { }
```

REDUX

Make “cover” a pcdm:Object

PREFIX pcdm: <<http://pcdm.org/models#>>

INSERT { <> a pcdm:Object }

WHERE { }

Transactions

Multiple actions can be bundled together into a single repository event (transaction)

Transactions can be rolled back or committed

Can be used to maintain consistency

Hands-on: TXNs



Authorization

The authorization framework provides a plug-in point within the repository that calls out to an optional authorization enforcement module

Currently, four authorization implementations exist:

- No-op
- Role-based
- XACML and
- **WebAC**

Hands-on: AuthZ

Create following Containers

- “files”

...contained inside “cover”

- “my-acls”

...at top-level is fine

- “acl”

...contained inside “my-acls”

- “authorization”

...contained inside “acl”


Final result (structure)

- cover/
 - files/

- my-acls/
 - acl/
 - authorization/

Final result (structure)

“cover” must point to its ACL

- cover/
 - files/
 - my-acls/
 - acl/
 - authorization/
- acl:accessControl
- 

- *An ACL must have one or more authorizations*
- *“authorizations” define:*
 - *agent(s)*
 - *mode(s)*
 - *resource(s) or class*

Define the “authorization”

PREFIX acl: <http://www.w3.org/ns/auth/acl#>

PREFIX pcdm: <http://pcdm.org/models#>

INSERT {

<> a acl:Authorization ;

acl:accessToClass pcdm:Object ;

acl:mode acl:Read, acl:Write;

acl:agent "adminuser" .

} WHERE { }

Link “acl” to “cover”

-- Update “cover” resource --

PREFIX acl: <http://www.w3.org/ns/auth/acl#>

INSERT {

 <> acl:accessControl </fcrepo/rest/my-acls/acl>

} WHERE { }

Versioning

Versions can be created on resources with an API call

A previous version can be restored via the REST-API

Hands-on: Versioning

Create version “v0” of “cover”

*** Warning cURL sighting ***

```
curl -ufedoraAdmin:secret3 -i -XPOST -H"slug: v0"  
localhost:8080/fcrepo/rest/cover/fcr:versions
```

Add dc:publisher to “cover”

```
INSERT {
```

```
  <> dc:publisher "The Press"
```

```
}
```

```
WHERE { }
```

Create version “v1” of “cover”

```
curl -ufedoraAdmin:secret3 -i -XPOST -H"slug: v1"  
localhost:8080/fcrepo/rest/cover/fcr:versions
```

* Inspect and Revert

Hands-on: Fixity

Fixity

Over time, digital objects can become corrupt

Fixity checks help preserve digital objects by verifying their integrity

On 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

Create some cover binaries

...contained inside “files”

cover.jpg

cover.tif

* Fixity

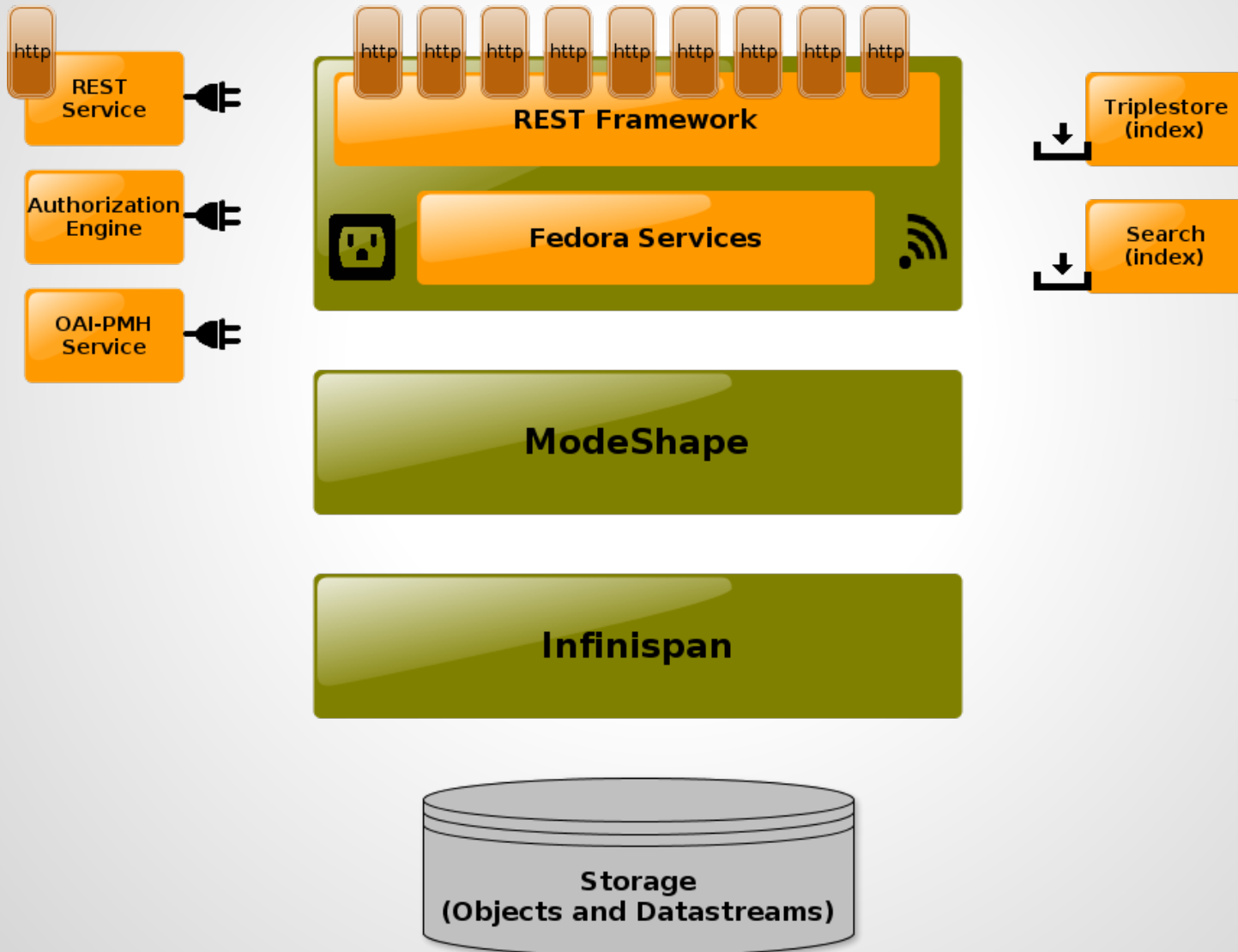
* Corrupt and test?

Non-core Features

Two Non-Core Feature Types

1. External components
 - Consume and act off repository messages
2. Optional, pluggable components
 - Separate projects that can interact with Fedora 4 using a common pattern

Component Architecture



External Component Integrations

Leverages the well-supported Apache Camel project

- Camel is middleware for integration with external systems
- Can handle any asynchronous, event-driven workflow

External - Indexing

Index repository content for search

Content can be assigned the `rdf:type` property "Indexable" to filter from non-indexable content

Solr has been tested

External - Triplestore

An external triplestore can be used to index the RDF triples of Fedora resources

Any triplestore that supports SPARQL-update can be plugged in

Fuseki, Sesame, BlazeGraph have been tested

External & Pluggable - Audit Service

Maintains a history of events for each repository resource

Both internal repository events and events from external sources can be recorded

Uses the existing event system and an external triplestore

Pluggable - OAI Provider

fcrepo4-oaiprovider 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 of the box, but users are able to add their own metadata format definitions to oai.xml

Pluggable - SWORD Server

SWORD is a lightweight protocol for depositing content from one location to another

fcrepo4-swordserver implements 2.0 AtomPub Profile, using Fedora 4 as the backend

SWORD v2 includes AtomPub CRUD operations

Success!