Combining Content, Semantic Relationships, and Web Services – Fedora
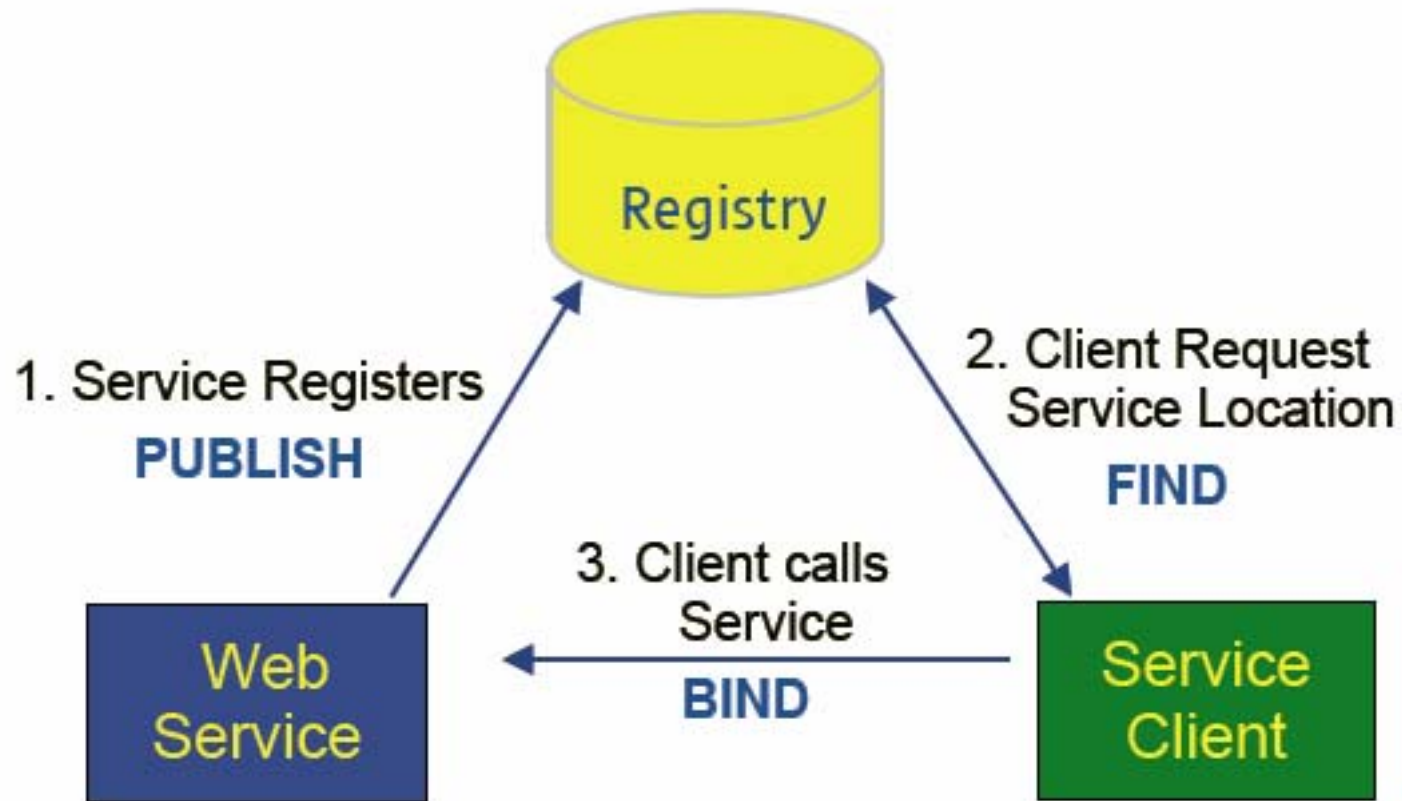
CS 431 – April 12, 2006
Carl Lagoze – Cornell University

What is a Web Service

- A software application
- Identified by a URI
- Interfaces and bindings described by XML
- Supports direct interactions with other software applications
- Messaging (request and response) based on XML
- Uses Internet protocols (e.g., HTTP)

Basic Web Service Architecture

Web Services Compents (SOAP)

- Simple Object Access Protocol
- XML-based RPC
- Uses XML for data encoding
- Defines
  - Message envelope
  - Encoding Rules
  - RPC Convention
  - Binding with underlying protocol

SOAP RPC Request Example

```
<SOAP-ENV:Envelope
    xmlns:SOAP-ENV="..."
    SOAP-ENV:encodingStyle="...">
  <SOAP-ENV:Header>
    <!-- Optional context information -->
  </SOAP-ENV:Header>

  <SOAP-ENV:Body>
    <m:GetLastTradePrice xmlns:m="some_URI">
      < tickerSymbol> SUNW</tickerSymbol>
    </m:GetLa stTradePric e>
  </SOAP-ENV:Body>

</SOAP-ENV:Envelope>
```

SOAP RPC Response Example

```
<SOAP-ENV:Envelope
    xmlns:SOAP-ENV="..."
    SOAP-ENV:encodingStyle="...">
  <SOAP-ENV:Header>
      <!-- Optional context information -->
  </SOAP-ENV:Header>

  <SOAP-ENV:Body>
    <m:GetLastTradePriceResponse xmlns:m="some_URI">
        <price>30 .5</price>
    </m:GetLa stTradePric eResponse>
  </SOA P-ENV:Body>

</SOAP-ENV:Envelope>
```

Web Service Components - WSDL

- Web Services Description Language
- Components
  - Abstract Definition of operations and messages
  - Concrete binding to network protocol and endpoint address
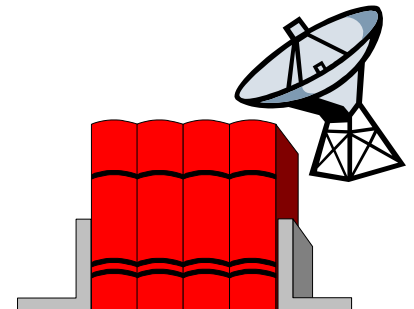
The Fedora Project

- Fedora
  - Flexible
  - Extensible
  - Digital
  - Object
  - Repository
  - Architecture

- Open source software
  - Not Red Hat !
  - Mozilla Public License

- http://www.fedora.info

Heterogeneous Digital Content

- Conventional objects
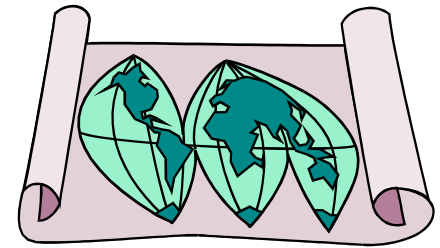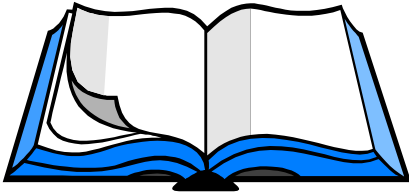
- Complex, compound, dynamic objects

Fedora History

- **Cornell Research (1997-present)**
  - DARPA and NSF-funded research
  - First reference implementation developed
  - Distributed, Interoperable Repositories (experiments with CNRI)
  - Policy Enforcement

- **First Application (1999-2001)**
  - University of Virginia digital library prototype
  - Technical implementation: adapted to web; RDBMS storage
  - Scale/stress testing for 10,000,000 objects

- **Open Source Software (2002-present)**
  - Andrew W. Mellon Foundation grants
  - Technical implementation: XML and web services
  - Fedora 1.0 (May 2003)
  - Fedora 2.0 (Jan 2005)

Fedora Use Cases

- Digital Library Collections
- Institutional Repository
- Educational Software

- Information Network Overlay
- Digital archives and preservation
- Digital Asset Management
- Content Management System
- Scholarly publishing

Selected Fedora Users

- **University of Virginia: digital library (**image collector**,** EAD**, e-texts)**
- **VTLS (software company): commercial product (**VITAL**)**
- **Tufts University: education (**VUE**/concept maps); digital library**
- **Northwestern: academic technologies (**images**,** art**, video, e-texts)**
- **National Science Digital Library (NSDL): Cornell Core Integration**
- **ARROW: National Library of Australia and Monash University**
- **Royal Library of Denmark and DTU**
- **Rutgers University:** digital library **(e-journals, numeric data)**
- **Indiana University:** EVIA Digital Archive **(video)**
- **American Geophysical Union: scholarly publications**
- **Max Planck Institue: Scholarly Communication**
- **Cornell University: Bear Access**
- **Yale University – electronic records**
- **New York University: humanities computing; digital library**
- **OhioLink**
- **DISA – South Africa, History of Apartheid resistance**

Why Fedora? (1)

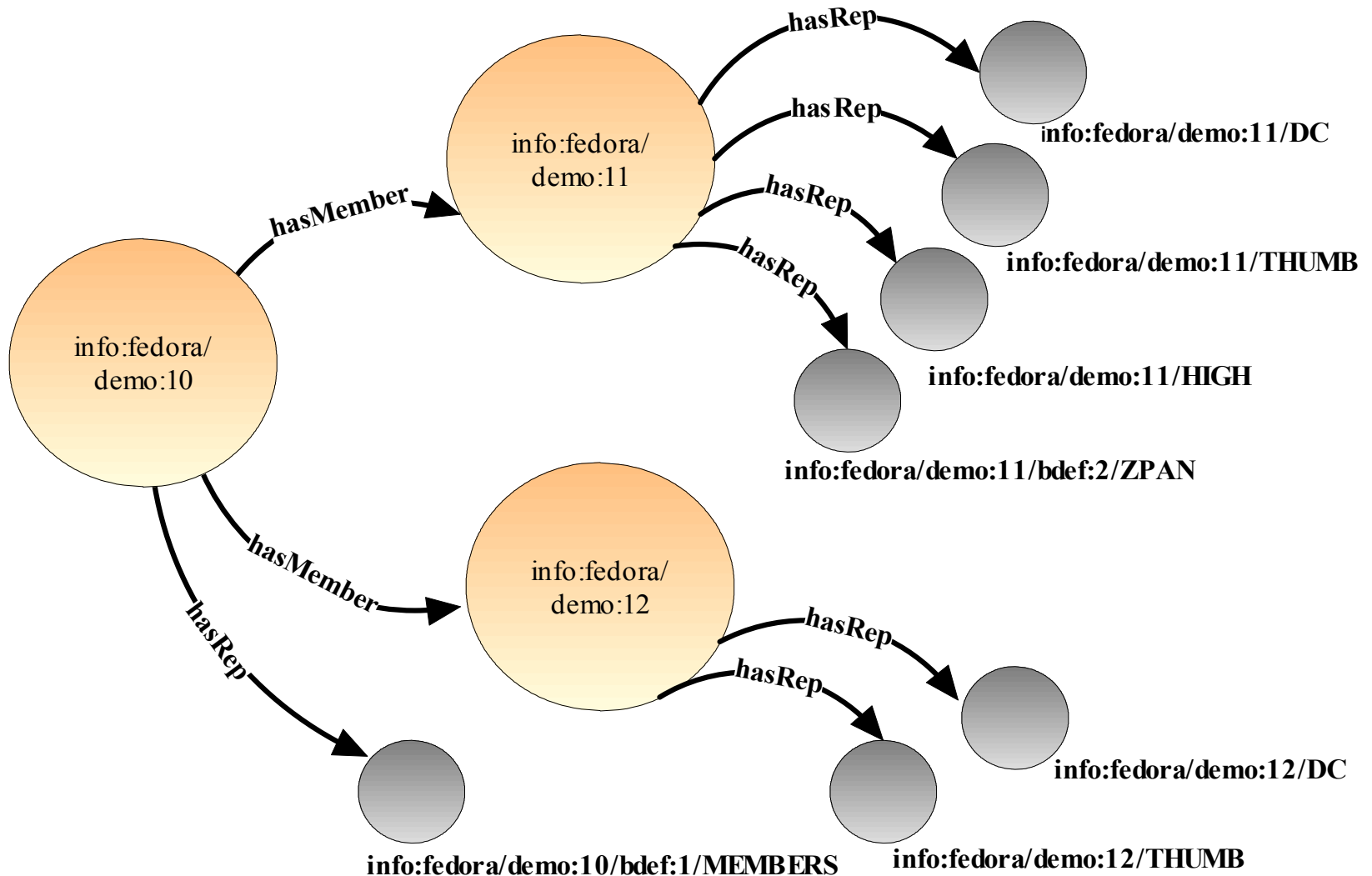- ## Digital Object Model
  - Abstraction for heterogeneous digital resources
  - Container for content and metadata
  - Aggregate local and remote content
  - Associate behaviors with objects (extensible service interfaces)
- ## Repository web service
  - Digital object storage
  - Web service APIs (SOAP and REST) to manage, access, search
  - Relationships
  - Define and query object-to-object relationships
- ## Feature-worthy for archiving and preservation
  - XML object serialization for ingest, storage, and export
  - Content versioning
  - Event history

Why Fedora? (2)

- **Content repurposing**
  - Reuse digital content in different contexts
  - Re-purpose content via mechanisms for dynamically transforming content to fit new requirements
- **Web Services**
  - SOAP and REST bindings
  - WSDL to define interfaces
  - XML transmission
- **Easy integration with other apps and systems**
  - Does not assume any particular workflow or end-user application
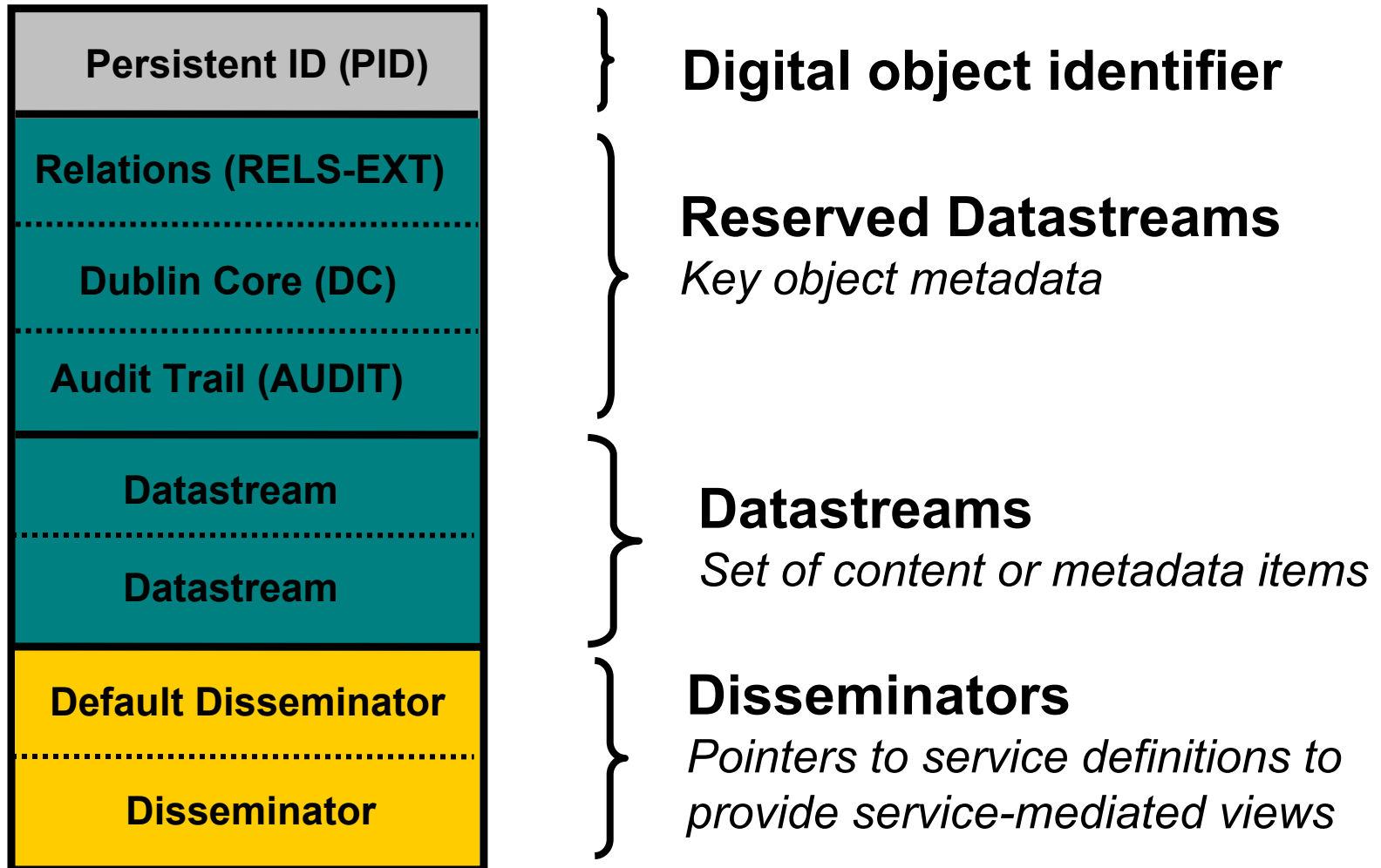 - Generic repository service as substrate

Digital Object Model

"Graph" View of Fedora Objects

Fedora Digital Object Model
Component View

The Datastream Component
4 Classifications for Datastreams

Inline XML

Fedora stores a name-spaced block of XML content within the Fedora digital object XML file.

Managed Content

Fedora stores and manages the content bytestream (non-XML content)

External Referenced

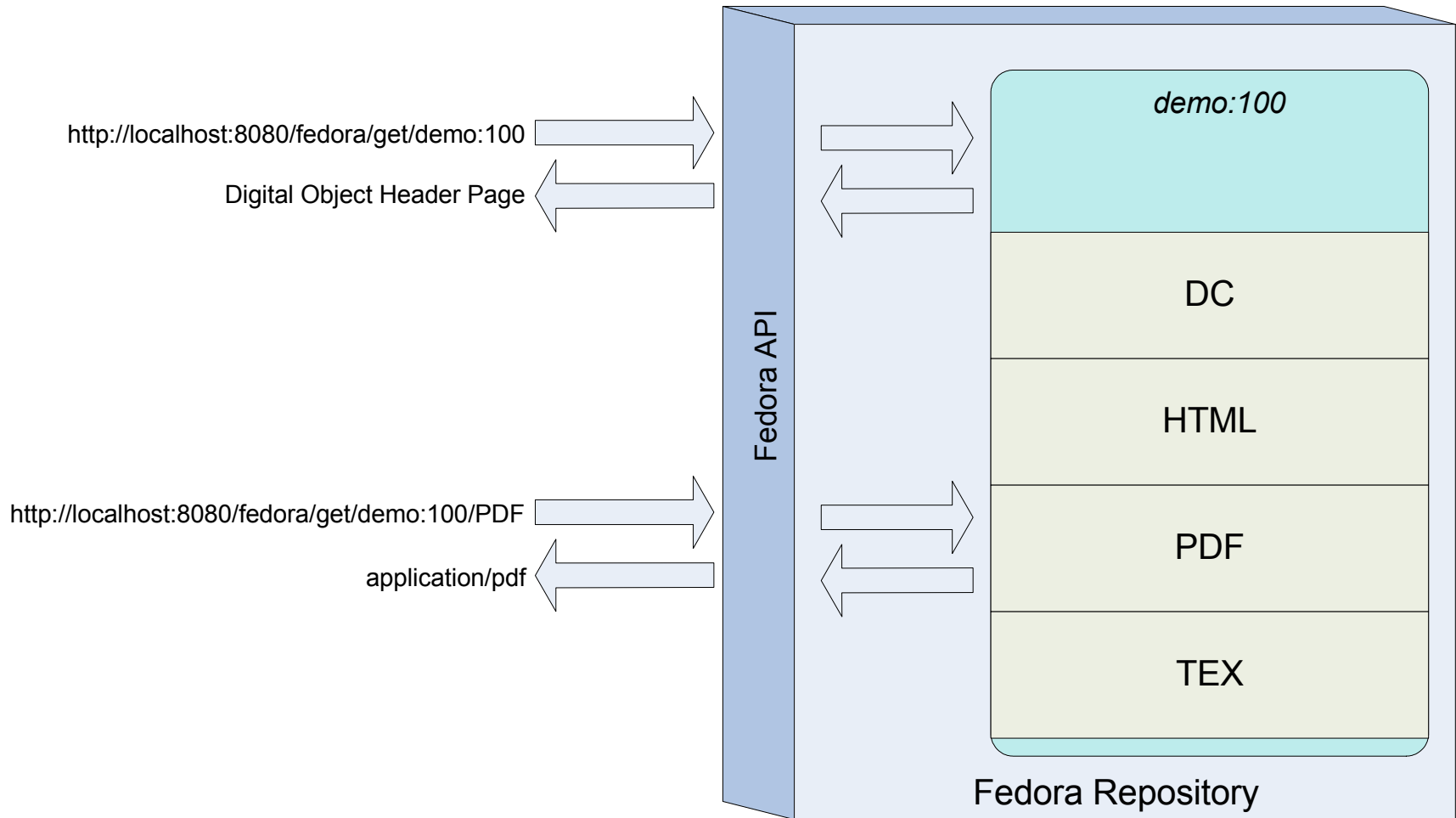*Fedora stores a reference (URL) to the content*

External Redirected

Fedora stores a reference (URL) to the content, but will not mediate access to content. (Optimized for streaming)

Simple Fedora model for aggregating static content

- Representations map to datastreams
- Datastreams may be local or surrogates (redirect) to remote data
- REST URL's give client access to representations

Digital Object Aggregating Local Content

http://localhost:8080/fedora/get/demo:100

Digital Object Header Page

Fedora API

demo:100

DC

HTML

http://localhost:8080/fedora/get/demo:100/PDF

application/pdf

PDF

TEX

Fedora Repository

Digital Object for Local and Remote Content
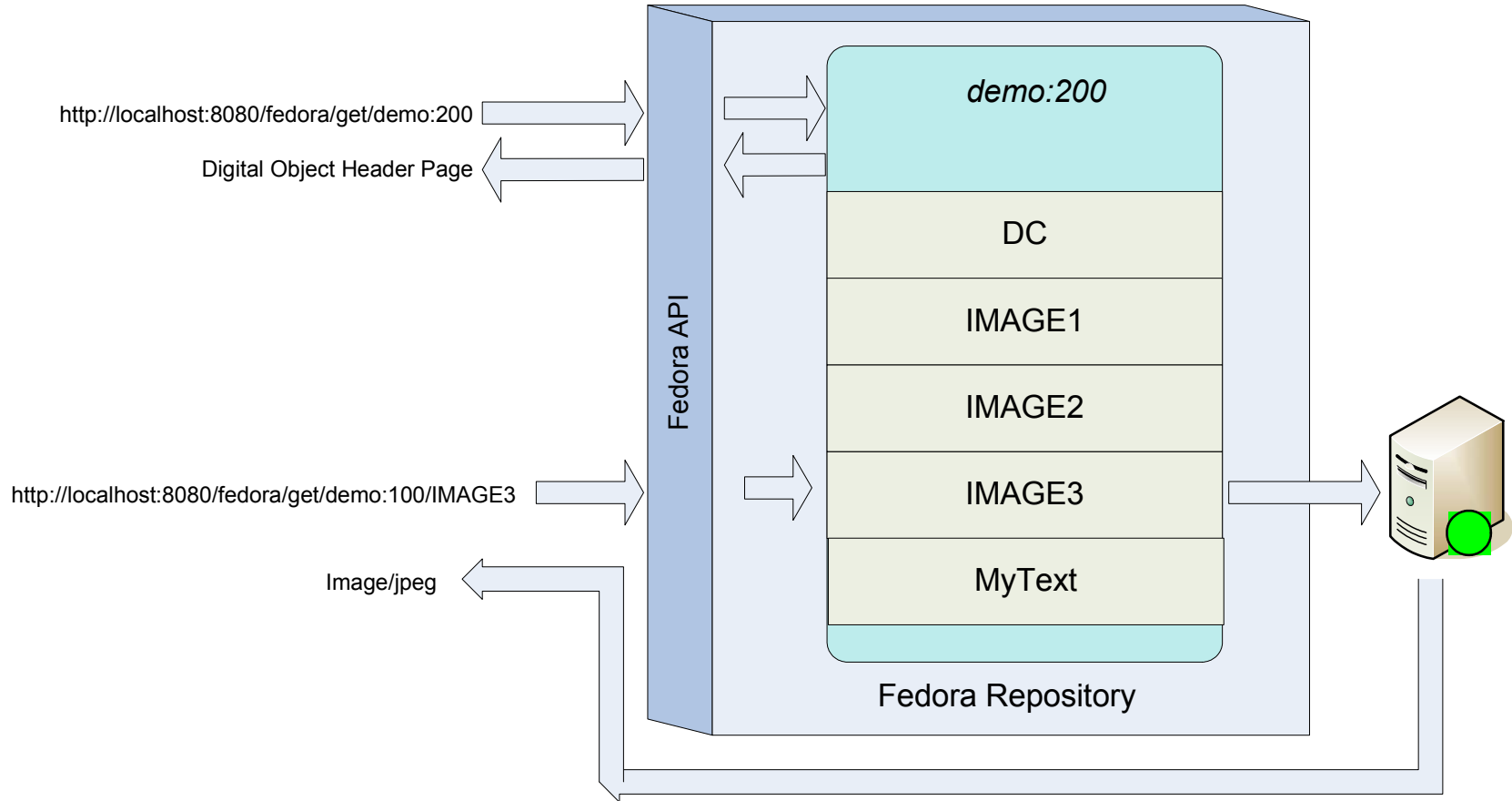
Fedora for dynamic content

- Representations map to service-based transforms of data (in addition to static datastreams)
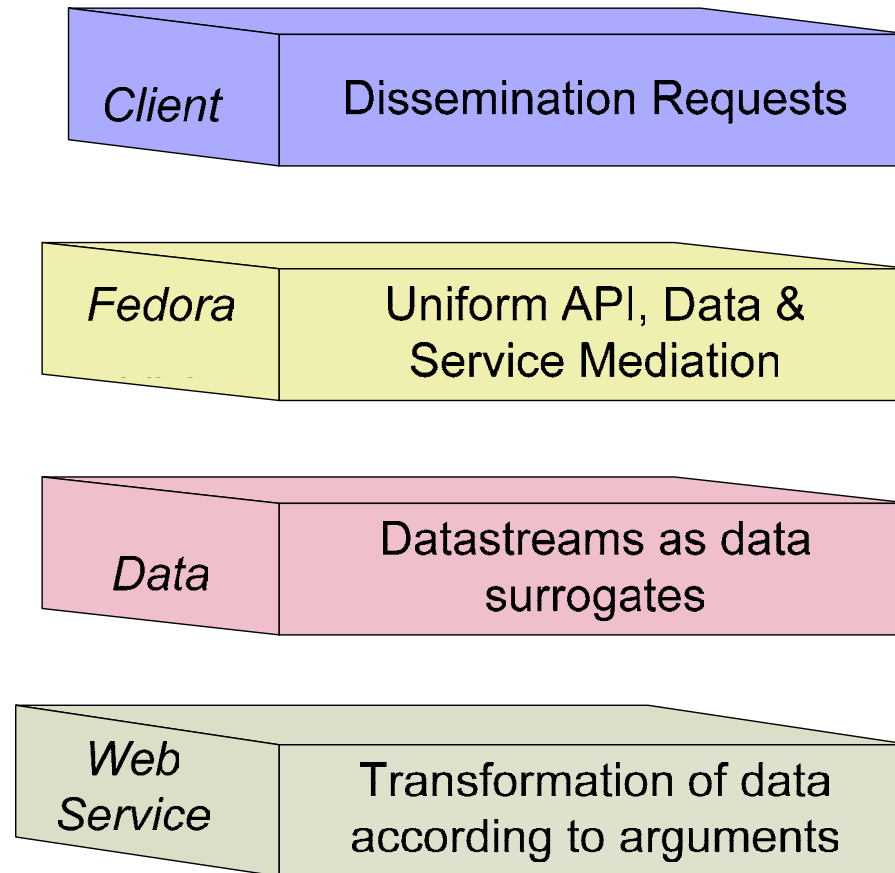- Opaque to REST based access (client see only representations, not how they are produced)
- Motivating examples
  - Canonical XML metadata format – XSLT to Dublin Core
  - Document source in TeX, programmatic transform to PDF, PS, HTML, etc.
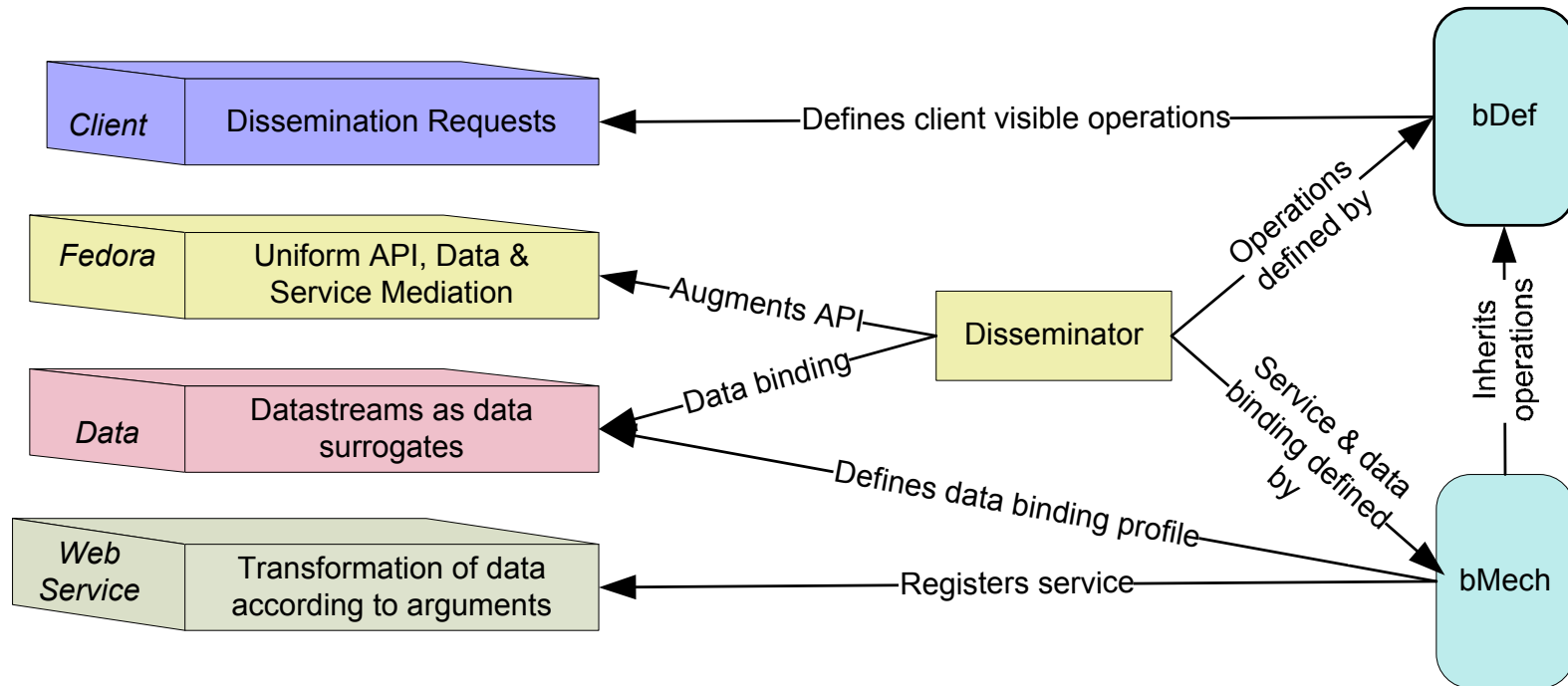
Understanding Dynamic Disseminations (1)

Client — Dissemination Requests

Fedora — Uniform API, Data & Service Mediation

Data — Datastreams as data surrogates

Web Service — Transformation of data according to arguments

Understanding Dynamic Disseminations (2)

- ## Behavior Definitions (bDef)
  - Special digital object defining client side functionality (method template)
- ## Behavior Mechanism (bMech)
  - Special digital object that refines a bDef by defining:
    - Data profile: set of datastreams required for execution
    - Service binding: where the work is performed
  - May be many bMechs for a bDef
- ## Disseminator
  - Association of a bMech/bDef with a digital object endowing it with bDef-defined functionality (methods)
  - A digital object may have multiple disseminators (polymorphic typing)

Understanding Dynamic Disseminations (3)

Dynamic Dissemination Access
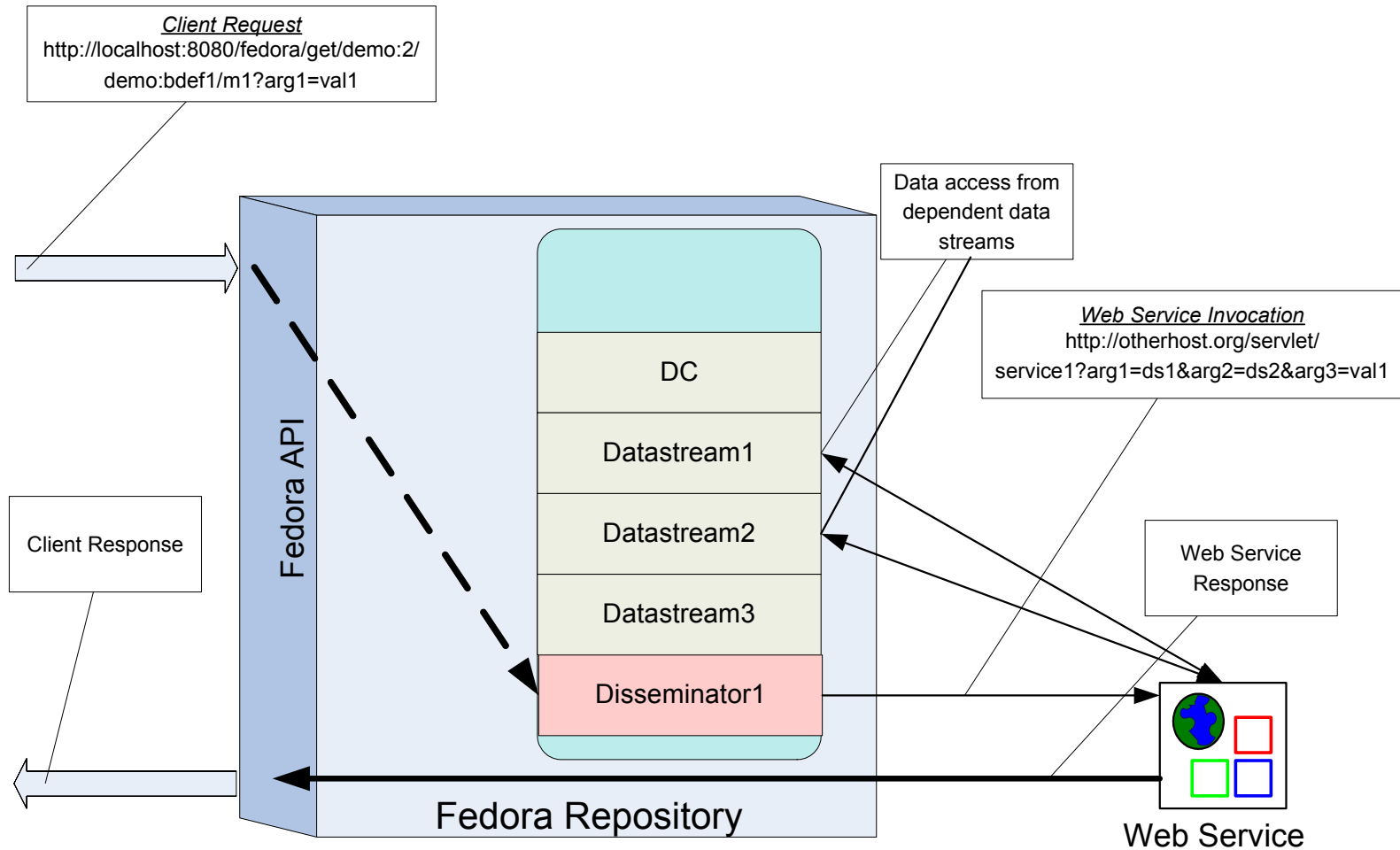
Dynamic Dissemination Example

Client Request
http://localhost:8080/fedora/get/
demo:300/demo:ex3bDef/getContent

Data access from
dependent data
streams

Web Service Invocation
http://localhost:8080/service/
saxon?xsl=XSL&source=Source

Fedora API

DC

XSL
(xml to HTML)

Source
(poem data)

HTML Output

Saxon Response

Disseminator1

Fedora Repository

Saxon Service
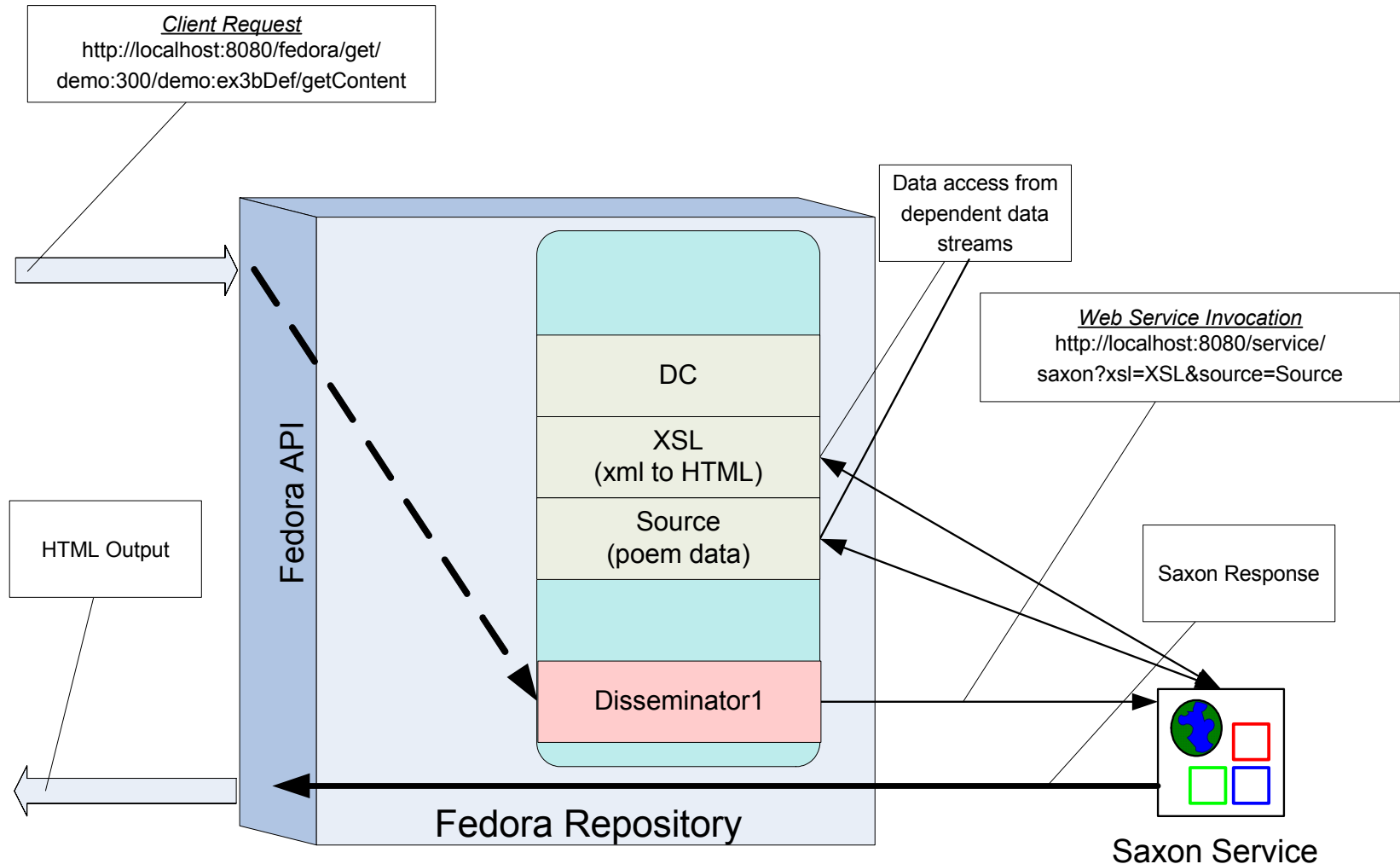
Fedora – XML for digital objects

- **FOXML (Fedora Object XML)**
    - Simple XML format directly expresses Fedora object model
    - Easily adapts to Fedora new and planned features
    - Easily translated to other well-known formats
    - Internal storage format for objects in repository
- **XML-based Ingest/Export of objects**
    - FOXML, METS (Fedora extension)
    - Extensible to accommodate new XML formats
    - Planned: METS 1.4, MPEG21 DIDL

FOXML – Object Properties

```
<foxml:objectProperties>
    <foxml:property  NAME="http://www.w3.org/1999/02/22-rdf-syntax-ns#type" VALUE="FedoraObject"/>
     <foxml:property NAME="info:fedora/fedora-system:def/model#state" VALUE="A" />
     <foxml:property NAME="info:fedora/fedora-system:def/model#label" VALUE="Sandy's Test Object"/>
     <foxml:property NAME="info:fedora/fedora-system:def/model#contentModel" VALUE="TEST"/>
  </foxml:objectProperties>
```

FOXML – Datastream (type 'E')

```
<foxml:datastream CONTROL_GROUP="E" ID="DS5" STATE="A" VERSIONABLE="true">
    <foxml:datastreamVersion ID="DS5.0" MIMETYPE="image/x-mrsid-image" LABEL="Pavilion III">
        <foxml:contentLocation REF="http://iris.lib.virginia.edu/mrsid//archerp01.sid" TYPE="URL"/>
    </foxml:datastreamVersion>
</foxml:datastream>
```

FOXML –
Relationships Datastream

```
<foxml:datastream ID="RELS-EXT" CONTROL_GROUP="X">
    <foxml:datastreamVersion ID="RELS-EXT.0" MIMETYPE="text/xml" LABEL="Relationship Metadata">
        <foxml:xmlContent>
            <rdf:RDF xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#" ….>

                <rdf:Description rdf:about="info:fedora/image:100">
                        <fedora:isMemberOfCollection rdf:resource="info:fedora/history:49"/>
                        <fedora:isMemberOfCollection rdf:resource="info:fedora/architecture:48"/>
                </rdf:Description>

            </rdf:RDF>
        </foxml:xmlContent>
    </foxml:datastreamVersion>
</foxml:datastream>
```
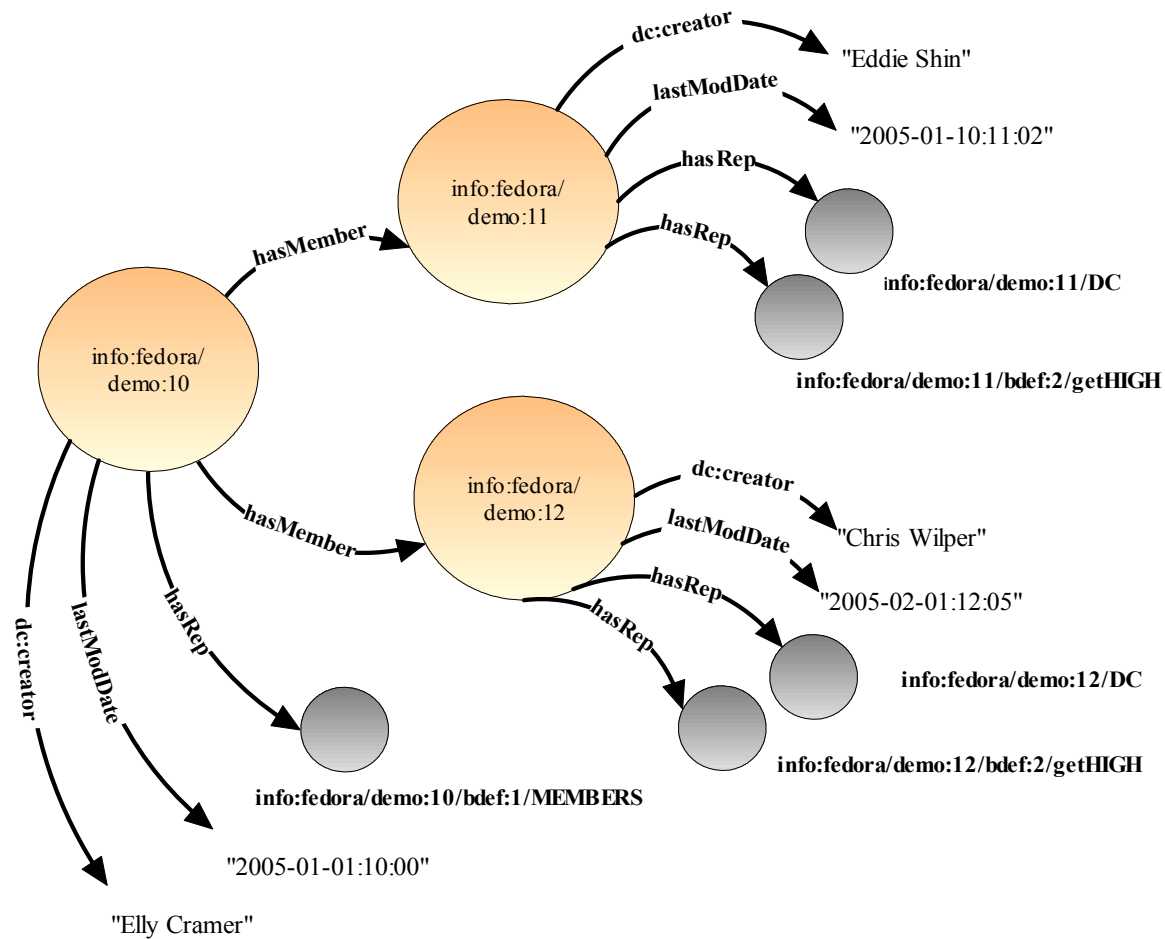
FOXML – Disseminator

```
<foxml:disseminator ID="DISS2" BDEF_CONTRACT_PID="demo:8" STATE="A" VERSIONABLE="true">
    <foxml:disseminatorVersion ID="DISS2.0" BMECH_SERVICE_PID="demo:9" LABEL="MrSID Service">
        <foxml:serviceInputMap>
            <foxml:datastreamBinding DATASTREAM_ID="DS5" KEY="MRSID" LABEL="Image binding"/>
        </foxml:serviceInputMap>
    </foxml:disseminatorVersion>
</foxml:disseminator>
```

Fedora Resource Index:
Using RDF and ontologies

Fedora Digital Objects
Resource Index View

Fedora 2.0 and RDF

- **Object-to-object Relationships**
  - Ontology of common relationships (RDF schema)
  - Relationships stored in special datastream (RELS-EXT)

- **Resource Index (RI)**
  - RDF-based index of repository (Kowari triple-store)
  - Graph-based index includes:
    - **Object properties and Dublin Core**
    - **Object Relationships**
    - **Object Disseminations**

- **RI Search**
  - Powerful querying of graph of inter-related objects
  - REST-based query interface (using RDQL or ITQL)
  - Results in different formats (triples, tuples, sparql)

Uses of Object Relationships

- Define collections (e.g., collection objects)
- Assert critical relationships among object for management purposes
- Enable network overlay
  - Surrogate objects referring to external entities
  - Assert relationships among them
  - Assert other relationships (e.g., annotations)
- Enable navigation of repository (as tree or graph)

Fedora Relationship Ontology (RDFS)

- isPartOf / hasPart
- isMemberOf / hasMember
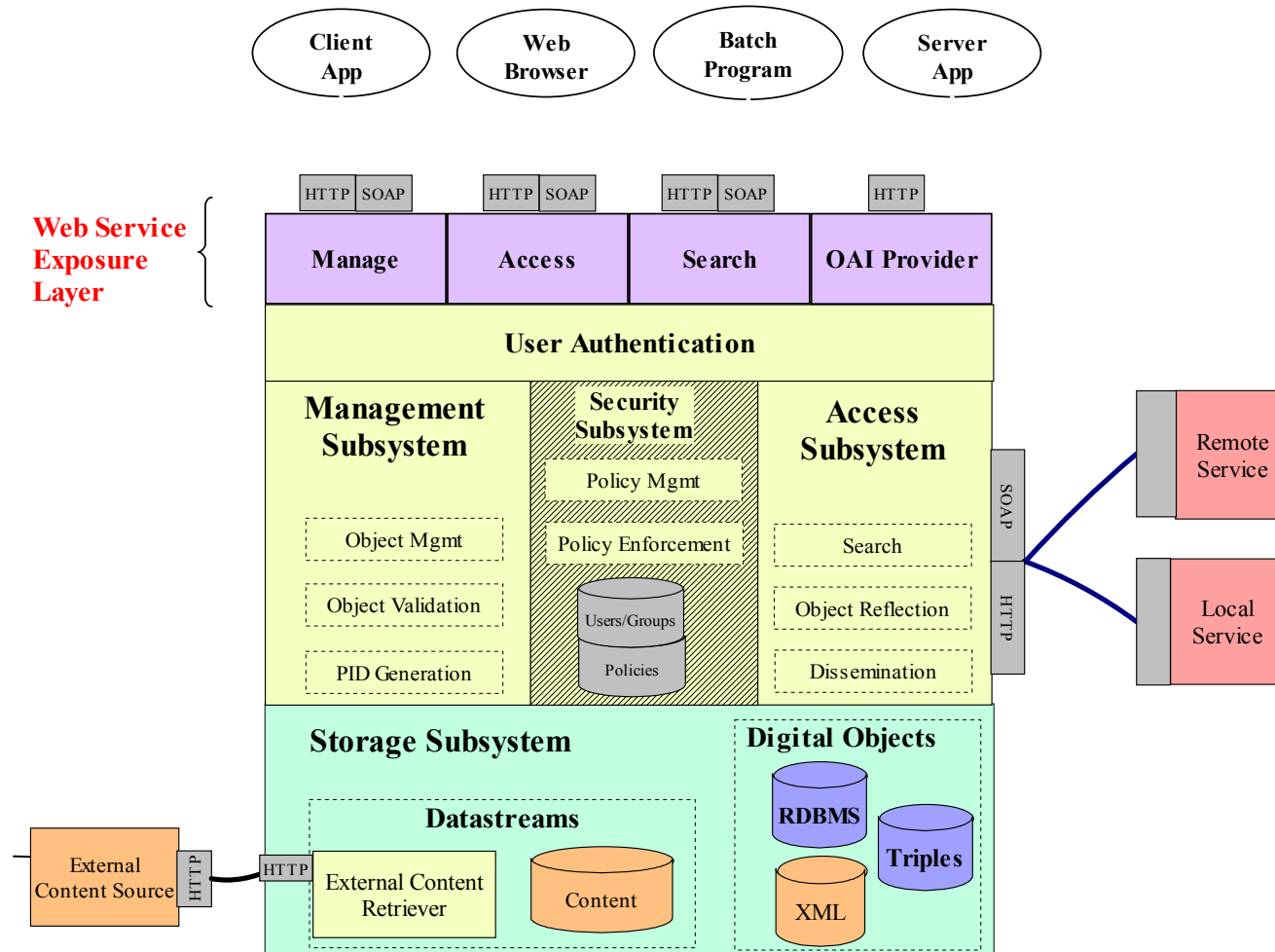- isDescriptionOf / hasDescription
- hasEquivalent
- … others

Demo:
Collection – Member Relationships

- Collection Object [smiley]
  - Datastream containing a query to Resource Index for all members of collection

- Image Objects [brush]
  - Use RELS-EXT datastream to assert relationship to collection object

Fedora Repository Service

Fedora Repository Service

Fedora Repository: 3 Layers

1. Interfaces	• Access/Search Service • Management Service • OAI Provider Service • Resource Index Service
2. Modules	Configurable modules that implement all repository functionality in terms of the Fedora digital object model.

3. Persistent Store	• RDBMS
	    – Digital object registry
	    – Object "cache" for performance
	• File System
	    – XML object serializations
	    – Managed Content (Datastreams)

Fedora 2.0 Server Feature Set

- **Management module**
  - **Ingest and Export (NEW! METS or FOXML)**
  - **Validation (XML and Schematron Rules)**
  - **PID assignment**
  - **Replication to object cache**
  - **Incremental indexing of metadata**
  - **Object create/modify/delete/purge**
- **XML Translation module**
  - **METS or FOXML ingest and export**
  - **Convert between formats**
- **Storage module:**
  - **File system for XML object wrappers**
  - **relational db object registry and object cache**
- **Content Versioning**
  - **Automatic version control for datastreams and disseminators**
  - **Enables *date-time stamped API requests* (see object as it looked then)**

Fedora 2.0 Server Feature Set

- **Access and Dissemination modules**
  - Mediation - auto-dispatching to distributed web services for content transformation
  - Built-in services: XSLT, image manipulation, xml-to-PDF
- **Search Module**
  - Searching of object properties and DC record of each object
- **Security module**
  - HTTP Basic Authentication and simple access control
  - NEW! LDAP tie-in for user attributes
  - NEW! XACML policies and policy enforcement
  - Future: Shibboleth
- **OAI-PMH**
- **Resource Index**
  - RDF-based index of repository (Kowari triple-store)
  - Contains key object attributes, DC, relationships
  - REST-based query interface (using RDQL or ITQL)

Fedora Web Service APIs in a Nutshell

- **Management Service (API-M)**

    – Ingest Object
    – Export Object
    – Get Object XML
    – Purge Object
    – Modify Object
    – Get Next PID

    – Get Datastream(s)
    – Get DatastreamHistory
    – **Get DisseminatorHistory**
    – Get Disseminator(s)

    – Add/modify/purge Datastream
    – Add/modify/purge Disseminator
    – Set State

Fedora Web Service APIs in a Nutshell

- **Access Service (API-A and API-A-LITE)**

  - Describe Repository
  - Get Object Profile
  - Get Object History
  - Get Datastream
  - Get Dissemination

  - Find Objects
  - Resume Find Objects

Fedora Web Service APIs in a Nutshell

- **API-A-Lite**

  - **Repository-level operations:**
    - **fedora/describe -** Describe Repository
    - **fedora/search** – methods to locate objects via the default repository index

  - **Object-level operations:**
    - **fedora/get -** method to get object profile
    - **fedora/get/..** – method to "disseminate" a view of an object's content
    - **Fedora/getMethods** – methods get information about all disseminations available on object

- **OAI-PMH Provider Service**
  - **All OAI-PMH methods to harvest OAI-DC from each object**

Fedora Administrator (via Fedora SOAP interfaces)
- **Java Swing client**
- **Ingest/Export objects**
- **Batch creation and modification of objects**
- **One-up creation and modification of objects**
- **Search repository**
- **Wizards for creating BDEF/BMECH objects**

- **Web Browser (via Fedora REST interfaces)**
  - **Access, Search,**
  - **OAI**
  - **Resource Index**
  - **Selected management operations**

- **Command Line Utilities**
  - **Ingest, export, purge**
  - **Migration**

Fedora Software Distribution

- **Open Source (Mozilla Public License)**
- **100% Java (Sun Java J2SDK1.4)**
- **Supporting Technologies**
  - Apache Tomcat and Apache Axis (SOAP)
  - Xerces for XML parsing and validation
  - Saxon for XSLT transformation
  - Schematron for validation
  - MySQL and Mckoi relational database
  - Oracle 9i support
  - Kowari for triple-store
- **Deployment Platforms**
  - Windows 2000, NT, XP
  - Solaris
  - Linux
  - Mac OSX

Fedora 2.1.1 (Apeil 2006)

- Authentication plug-ins
  - HTTP basic authentication and SSL
  - Plug-in #1 : user/password file
  - Plug-in #2 : LDAP tie-in
  - Plug-in #3 : Radius Authentication
- Authorization module
  - XACML policy enforcement for API operations
- New OAI Provider (stand-alone service)
- Support for MPEG21-DIDL (ingest/export/oai)
- Performance testing and improvements
- Integration with Storage Resource Borker