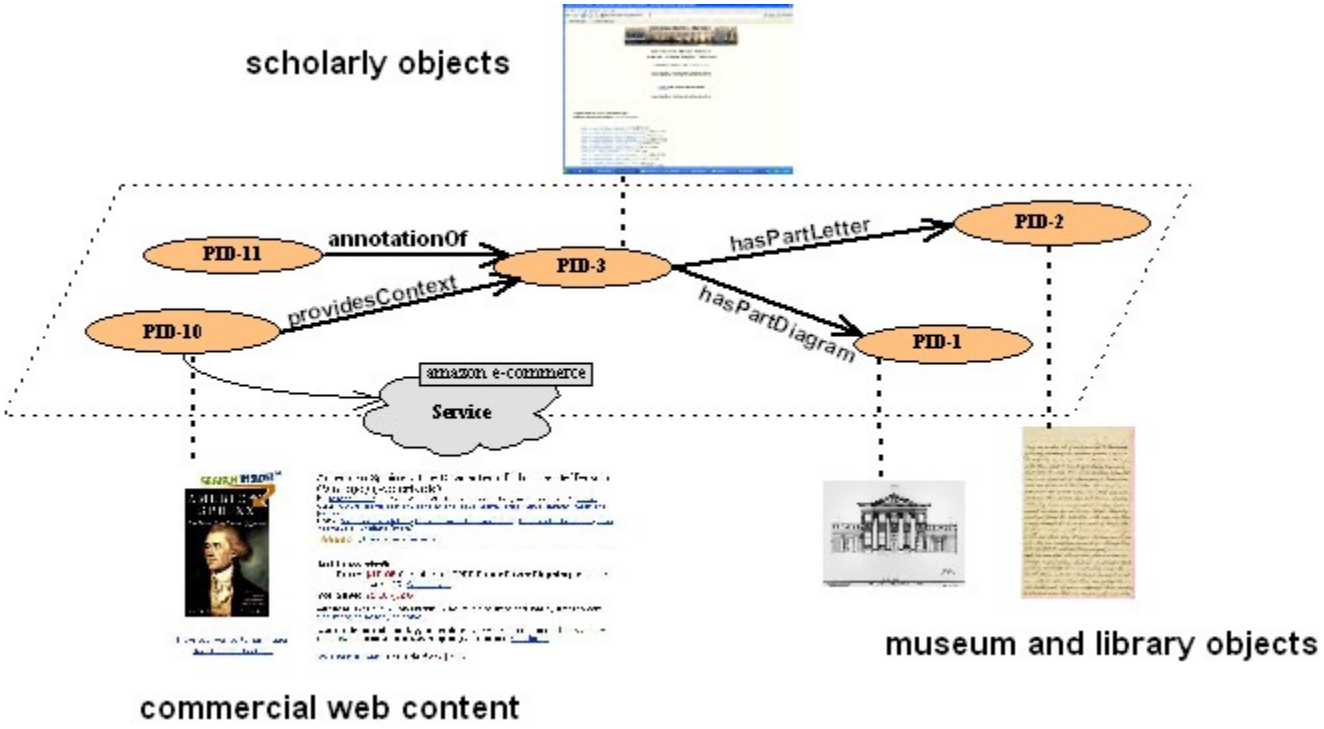
Digital Object Relationships

On page:

- What are Fedora Digital Object Relationships?
- Why are Fedora Digital Object Relationships Important?
- Where is Digital Object Relationship Metadata Stored?
- How is Digital Object Relationship Metadata Encoded?
  - RELS-EXT Validation
  - RELS-INT Validation
- Resource Index - RDF-based Indexing and Searching for Digital Objects

What are Fedora Digital Object Relationships?

Fedora digital objects can be related to other Fedora objects in many ways. For example there may be a Fedora object that represents a collection and other objects that are members of that collection. Also, it may be the case that one object is considered a part of another object, a derivation of another object, a description of another object, or even equivalent to another object. For example, consider a network of digital objects pertaining to Thomas Jefferson, in which scholarly works are stored as digital objects, which are related to other digital objects representing primary source materials in libraries or museums. The composite scholarly objects can be considered a graph of related digital objects. Other types of objects can also be related to the scholarly object over time, for instance annotations about the scholarly object can be created by others and related to the original object. Also, digital objects can be created to act as "surrogates" or "proxies" for dynamically produced web content such as an Amazon page for a book relevant to the scholarly object. Such a network of digital objects can be created using Fedora, which in the abstract, would look like this:



Digital object relationship metadata is a way of asserting these various kinds of relationships for Fedora objects. A default set of common relationships is defined in the Fedora relationship ontology (actually, a simple RDF schema) which defines a set of common generic relationships useful in creating digital object networks. These relationships can be refined or extended. Also, communities can define their own ontologies to encode relationships among Fedora digital objects. Relationships are asserted from the perspective of one object to another object as in the following general pattern:

```
<subjectFedoraObject>    <relationshipProperty>    <targetFedoraObject>
```

The first Fedora object is considered the "subject" of the relationship assertion. The relationship, itself, is considered a property of the subject. The target Fedora object is the related object. Thus, a valid relationship assertion as an English-language sentence might be:

```
<MyCatVideo>    <is a member of the collection>    <GreatCatVideos>
```

Using RELS-INT, relationships can also be asserted from the datastream of one object to another object or datastream.

```
<MyCatPicture/Thumbnail>    <is thumbnail of>    <MyCatPicture/FullSizeImage>
```

Why are Fedora Digital Object Relationships Important?

The creation of Fedora digital object relationship metadata is the basis for enabling advanced access and management functionality driven from metadata that is managed within the repository. Examples of the uses of relationship metadata include:

- Organize objects into collections to support management, OAI harvesting, and user search/browse
- Define bibliographic relationships among objects such as those defined in Functional Requirements for Bibliographic Records
- Define semantic relationships among resources to record how objects relate to some external taxonomy or set of standards
- Model a network overlay where resources are linked together based on contextual information (for example citation links or collaborative annotations)
- Encode natural hierarchies of objects
- Make cross-collection linkages among objects (for example show that a particular document in one collection can also be considered part another collection)

Where is Digital Object Relationship Metadata Stored?

Object-to-Object relationships are stored as metadata in digital objects within special Datastreams. These Datastreams are known by the reserved Datastream identifiers of "RELS-EXT" (which stands for "Relationships-External") and "RELS-INT" (which stands for "Relationships-Internal"). Each digital object can have one RELS-EXT datastream which is used exclusively for asserting digital object relationships and one RELS-INT datastream which is used exclusively for asserting relationships from datastreams of the digital object.

RELS-EXT and RELS-INT Datastreams can be provided as part of a Fedora ingest file. Alternatively, they can be added to an existing digital object via component operations of the Fedora management service interface (i.e., addDatastream, addRelationship). Refer to the FOXML reference example to see an example of the RELS-EXT Datastream in context. Modifications to the RELS-EXT and RELS-INT Datastreams are made via the Fedora management interface (i.e., modifyDatastream).

The RELS-EXT and RELS-INT Datastreams should be encoded as either Inline XML Datastreams, meaning that the relationships metadata is expressed directly as XML within the digital object XML file, or as Managed Content datastreams. The datastream control group used when Fedora creates new relationships datastreams (eg as part of an addRelationship API method) is specified in the `DOManager` section of `fedora.fcfg` using the property `defaultRELSControlGroup`.

Ensure DC, RELS-EXT and RELS-INT are versionable if using Managed Content

Due to an outstanding bug FCREPO-849, if you use Managed Content for DC, RELS-EXT or RELS-INT then please make sure these datastreams are versionable (the default setting for versionable is "true", so if you haven't specified this datastream property then you are safe).

How is Digital Object Relationship Metadata Encoded?

Fedora object-to-object metadata is encoded in XML using the Resource Description Framework (RDF). The relationship metadata must follow a prescribed RDF/XML authoring style where the subject is encoded using `<rdf:Description>`, the relationship is a property of the subject, and the target object is bound to the relationship property using the `rdf:resource` attribute. The subject of a relationship assertion must be a URI that identifies either a Fedora digital object for RELS-EXT, or a Fedora digital object datastream for RELS-INT. These URIs are based on Fedora object PIDs and conform to the syntax described for the Fedora "info" URI scheme.

The syntax for asserting RELS-EXT relationships in RDF is as follows:

```
<rdf:RDF
  xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
  xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#"
  xmlns:fedora="info:fedora/fedora-system:def/relations-external#"
  xmlns:myns="http://www.nsdl.org/ontologies/relationships#">
  <rdf:Description rdf:about="info:fedora/demo:99">
        <fedora:isMemberOfCollection rdf:resource="info:fedora/demo:c1"/>
        <myns:isPartOf rdf:resource="info:fedora/mystuff:100"/>
        <myns:owner>Jane Doe</myns:owner>
  </rdf:Description>
</rdf:RDF>
```

The above RDF fragment indicates that the Fedora digital object identified as "info:fedora/demo:99" is the subject that is being described (as specified by the rdf:about attribute). This object has two relationships to other digital objects. It has an "isMemberOfCollection" relationship to the object identified as "info:fedora/demo:c1" which is a Fedora object that represents a collection. This collection object is considered the target of the relationship assertion. The second relationship assertion that is just like the first one, except that it asserts that the subject is a part of another Fedora digital object, "info:fedora /mystuff:100". The third relationship asserts that the digital object has owner "Jane Doe". Note that the object of this relationship is a text literal, not a URI.

The syntax for asserting RELS-INT relationships in RDF is as follows:

```
<rdf:RDF
  xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
  xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#"
  xmlns:fedora="info:fedora/fedora-system:def/relations-external#"
  xmlns:myns="http://www.nsdl.org/ontologies/relationships#">
  <rdf:Description rdf:about="info:fedora/demo:99/Thumbnail">
        <myns:isThumbnailOf rdf:resource="info:fedora/demo:99/FullSizeImage"/>
  </rdf:Description>
  <rdf:Description rdf:about="info:fedora/demo:99/FullSizeImage">
        <myns:hasImageSize>1600 x 900</myns:hasImageSize>
  </rdf:Description>
</rdf:RDF>
```

The above RDF fragment indicates that the Fedora digital object datastream identified as "info:fedora/demo:99/Thumbnail" is the subject that is being described (as specified by the rdf:about attribute) in the first rdf:Description element. This datastream has a relationship to another datastream, in this case within the same digital object. It has an "isThumbnailOf" relationship to the datastream identified as "info:fedora/demo:99/FullSizeImage", asserting that it is a thumbnail of the full-sized image.

The second rdf:Description element has the datastream "info:fedora/demo:99/FullSizeImage" as the subject, and it asserts that the image size is 1600 x 900 pixels. Note that the object of this relationship is a text literal, not a URI.

Unlike RELS-EXT, the RDF for RELS-INT can have multiple rdf:Description elements, with each having an rdf:about attribute identifying a datastream within this Fedora digital object, and each rdf:Description element can contain one or more relationships with specified datastream as the subject.

To ensure the integrity of relationship metadata so that it can be properly indexed by Fedora, the Fedora repository service validates all RELS-EXT and RELS-INT Datatreams and enforces the following constraints on the RDF.

RELS-EXT Validation

1. The subject must be encoded as an `<rdf:Description>` element, with an "rdf:about" attribute containing the URI of the digital object in which the RELS-EXT Datastream resides. Thus, relationships are asserted about this object only. Relationship directionality is from *this* object to other objects.

2. The relationship assertions must be RDF properties associated with the `<rdf:Description>`. Relationship assertions can be properties defined in the default Fedora relationship ontology, or properties from other namespaces.

3. Prior to 2.1, the objects of relationships were restricted to other Fedora digital object URIs. This has since been relaxed so that a relationship property may reference any URI or literal, with the following exception: a relationship may not be self-referential, `rdf:resource` attribute must not point to the URI of the digital object that is the subject of the relationship.

4. There must be only one `<rdf:Description>` in the RELS-EXT datastream. One description can have as many relationship property assertions as necessary.

5. There must be no nesting of assertions. Specifically, there cannot be an `<rdf:Description>` within an `<rdf:Description>`. In terms of XML "depth," the RDF root is considered at the depth of zero. The must be one `<rdf:Description>` element that must exist at the depth of one. The relationship assertions are RDF properties of the `<rdf:Description>` that exist at a depth of two.

6. Assertions of properties from certain namespaces for forbidden in RELS-EXT. There must NOT be any assertion of properties from the Dublin Core namespace or from the FOXML namespace. This is because these assertions exist elsewhere in Fedora objects and may conflict if asserted in two places. The RELS-EXT Datastream is intended to be dedicated to solely object-to-object relationships and not used to make general descriptive assertions about objects.

RELS-INT Validation

1. The subject must be encoded as an `<rdf:Description>` element, with an "rdf:about" attribute containing the URI of a digital object datastream in which the RELS-INT Datastream resides. Thus, relationships are asserted about datastreams in this object only. Relationship directionality is from *this* object's datastreams to other objects and datastreams. (Note that the existence of the datastream is not checked, the URI must be syntactically valid for a datastream in this digital object, but the datastream does not actually have to exist.)

2. The relationship assertions must be RDF properties associated with the `<rdf:Description>`. Relationship assertions can be properties defined in the default Fedora relationship ontology, or properties from other namespaces.

3. A relationship property may reference any URI or literal.

4. There may be more than one `<rdf:Description>` elements. One `<rdf:Description>` should be included for each datastream to be described, and each must have an "rdf:about" attribute identifying the datastream being described. One description can have as many relationship property assertions as necessary.

5. There must be no nesting of assertions. Specifically, there cannot be an `<rdf:Description>` within an `<rdf:Description>`. In terms of XML "depth," the RDF root is considered at the depth of zero. The must be one `<rdf:Description>` element that must exist at the depth of one. The relationship assertions are RDF properties of the `<rdf:Description>` that exist at a depth of two.

6. Assertions of properties from certain namespaces is forbidden in RELS-INT. There must NOT be any assertion of properties from the Fedora object properties namespaces (model and view). This is because these assertions exist elsewhere in Fedora objects and may conflict if asserted in two places.

Resource Index - RDF-based Indexing and Searching for Digital Objects

The Fedora repository service automatically indexes the RELS-EXT and RELS-INT Datastreams for all objects as part of the RDF-based Resource Index.

This provides a unified "graph" of all the objects in the repository and their relationships to each other. The Resource Index graph can be queried using SPARQL or iTQL which are SQL-like query languages for RDF. The Fedora repository service exposes an web service interface to search the Resource Index. Please refer to the Resource Index documentation for details.