LDP-PCDM-F4 In Action - Ordering

Ordering In Action

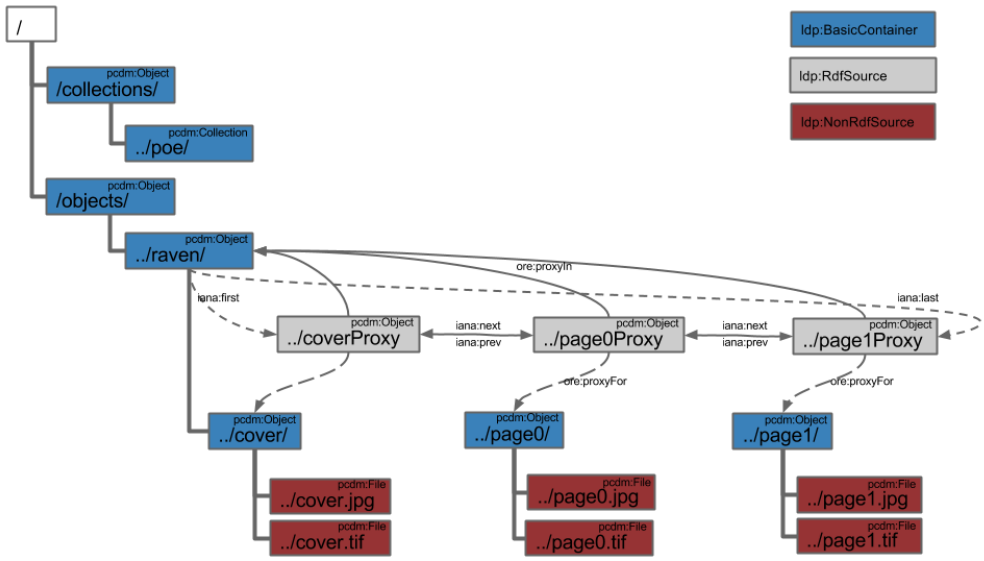
3: Final State - Ordered Pages

This final example will both illustrate a second use of ldp:DirectContainers as well as detail the PCDM recommendation for how to handle ordering of resources.

The addtional predicates/relationships that will be used in this example are:

- ore:proxyIn
- ore:proxyFor
- iana:first
- iana:next
- iana:prev
- iana:last

...all of which are further described in the Portland Common Data Model.



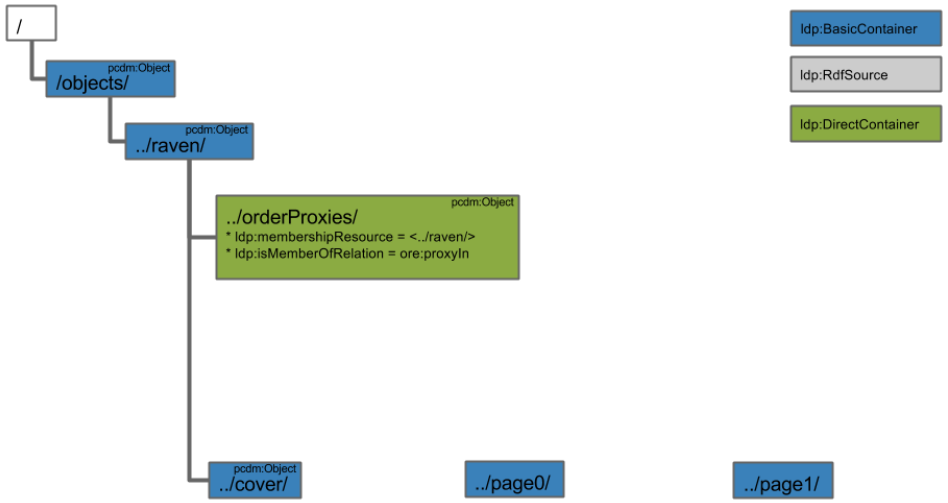
- Ordering - Create DirectContainer
- Ordering - Create Cover Proxy
- Ordering - Create Page0 Proxy
- Ordering - Create Page1 Proxy
- Ordering - Create Next and Prev
- Ordering - Create First and Last
- Ordering - Conclusion

Ordering - Create DirectContainer

Ordering - Create DirectContainer

As in the book example, begin with creating an ldp:DirectContainer, "orderProxies/", as a child of the book, "raven/", resource. This new DirectContainer will facilitate the auto-creation of triples that will define the membership relationship between the book, "raven/", and the proxies. Then, the new proxy resources within this DirectContainer will be used to establish an ordering of the books pages.

Note: This example assumes the previous creation of "/objects/raven/" and the cover and page resources from the Book example in this series.




```
curl -i -XPUT -H"Content-Type: text/turtle" --data-binary @ldp-ordering-direct.ttl localhost:8080/fcrepo/rest
/objects/raven/orderProxies/
```

Where "ldp-ordering-direct.ttl" follows:

ldp-ordering-direct.ttl

```
@prefix ldp: <http://www.w3.org/ns/ldp#>
@prefix pcdm: <http://pcdm.org/models#>
@prefix ore: <http://www.openarchives.org/ore/terms/>

<> a ldp:DirectContainer, pcdm:Object ;
  ldp:membershipResource </fcrepo/rest/objects/raven/> ;
  ldp:isMemberOfRelation ore:proxyIn .
```

An ldp:DirectContaner is an LDP construct that activates the creation of certain RDF triples when a new resource is added as a child of this container.

Like the "pages/" DirectContainer in an earlier example, the "orderProxies/" includes the ldp:membershipResource property ("raven/"). However, it is important to point out that the "orderProxies/" DirectContainer *does not* have the ldp:hasMemberRelation property defined, but instead uses ldp:isMemberOfRelation of "ore:proxyIn".
By using ldp:isMemberOfRelation, the auto-created triple resulting from the addition of a new child resource within "orderProxies/" will take the form:

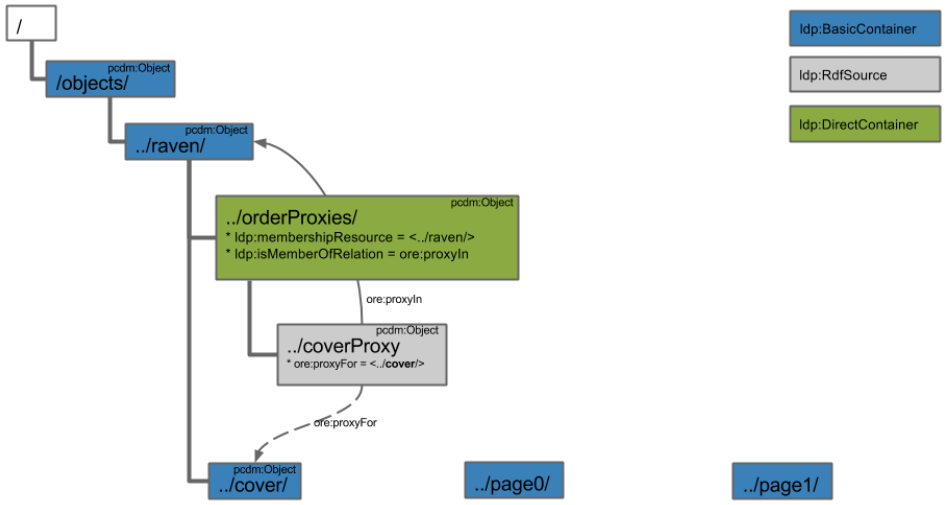
```
<new-resource> <ore:proxyIn> <http://localhost:8080/fcrepo/rest/objects/raven/>
```

We will see this in action next!

Ordering - Create Cover Proxy

Ordering - Create Cover Proxy

Create a new pcdm:Object, "coverProxy/", that is also an ldp:RdfSource within the "orderProxies/" DirectContainer.




```
curl -i -XPUT -H"Content-Type: text/turtle" --data-binary @ldp-cover-proxy.ttl localhost:8080/fcrepo/rest
/objects/raven/orderProxies/coverProxy
```

Where "ldp-cover-proxy.ttl" follows:

ldp-cover-proxy.ttl

```
@prefix pcdm: <http://pcdm.org/models#>
@prefix ore: <http://www.openarchives.org/ore/terms/>

<> a pcdm:Object ;
  ore:proxyFor </fcrepo/rest/objects/raven/pages/cover> .
```

As described in the previous step, the addition of "coverProxy" automatically creates the following new triple on "coverProxy"

```
<http://localhost:8080/fcrepo/rest/objects/raven/orderProxies/coverProxy> ore:proxyIn <http://localhost:8080
/fcrepo/rest/objects/raven>
```

Restating from the previous step,

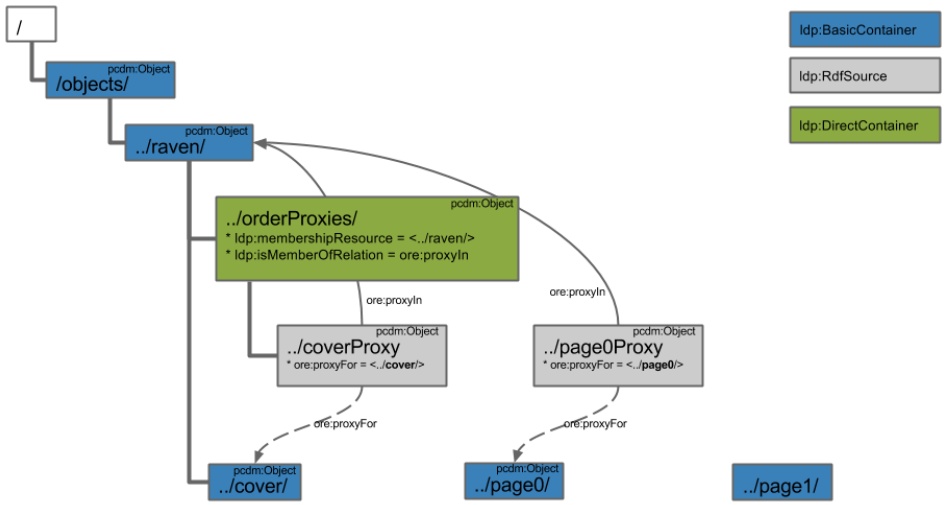
- the subject of the triple is the new resource ("coverProxy") that was added to the ldp:DirectContainer ("orderProxies/")
- the predicate of the triple comes from the "ldp:isMemberOfRelation" defined on "orderProxies/", and
- the object of the triple comes from the "ldp:membershipResource" defined on "orderProxies/"

Ordering - Create Page0 Proxy

Ordering - Create Page0 Proxy

In the same fashion as the previous step, adding "page0Proxy" to the DirectContainer, "orderProxies/" results in a new auto-generated triple on "page0Proxy" of the form:

```
<http://localhost:8080/fcrepo/rest/objects/raven/orderProxies/page0Proxy> ore:proxyIn <http://localhost:8080
/fcrepo/rest/objects/raven>
```




```
curl -i -XPUT -H"Content-Type: text/turtle" --data-binary @ldp-page0-proxy.ttl localhost:8080/fcrepo/rest
/objects/raven/orderProxies/page0Proxy
```

Where "ldp-page0-proxy.ttl" follows:

ldp-page0-proxy.ttl

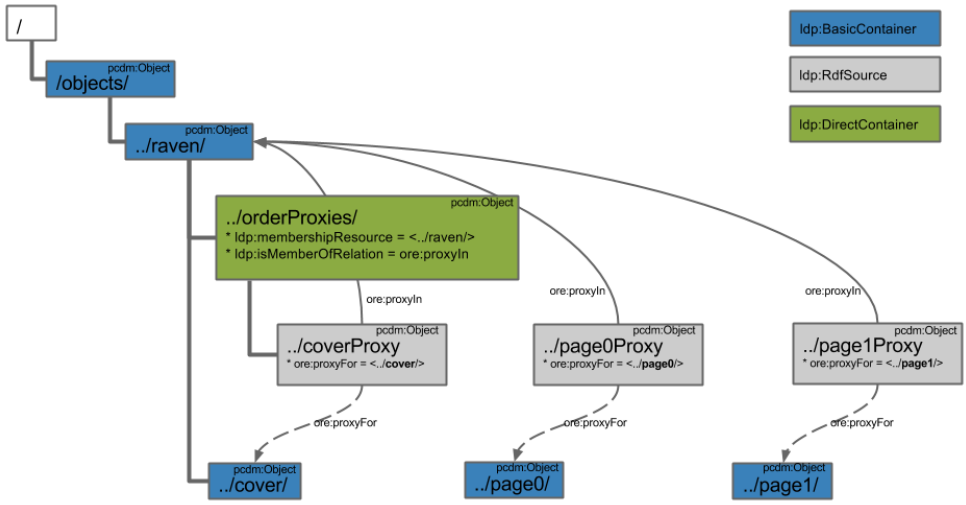
```
@prefix pcdm: <http://pcdm.org/models#>
@prefix ore: <http://www.openarchives.org/ore/terms/>

<> a pcdm:Object ;
  ore:proxyFor </fcrepo/rest/objects/raven/pages/page0/> .
```

Ordering - Create Page1 Proxy

Ordering - Create Page1 Proxy

This step in creating the final page, "page1Proxy", follows the same pattern shown in the previous two steps.




```
curl -i -XPUT -H"Content-Type: text/turtle" --data-binary @ldp-page1-proxy.ttl localhost:8080/fcrepo/rest
/objects/raven/orderProxies/page1Proxy
```

Where "ldp-page1-proxy.ttl" follows:

ldp-page1-proxy.ttl

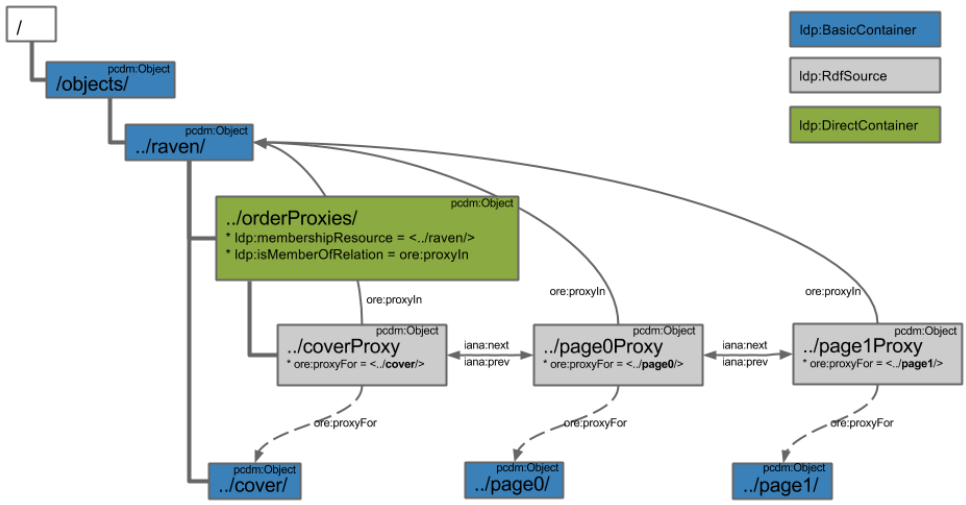
```
@prefix pcdm: <http://pcdm.org/models#>
@prefix ore: <http://www.openarchives.org/ore/terms/>

<> a pcdm:Object ;
  ore:proxyFor </fcrepo/rest/objects/raven/pages/page1/> .
```

Ordering - Create Next and Prev

Ordering - Create Next and Prev

Now that the proxies have been created, and associated with the book ("raven/") and the proxied resources, we can actually use the proxies to establish ordering, per the PCDM recommendations.
First, establish the order among the proxies with iana:next and iana:prev.



1) Establish "coverProxy" has iana:next of "page0Proxy":

```
curl -i -XPATCH -H"Content-Type: application/sparql-update" --data-binary @iana-cover-proxy.ru localhost:8080
/fcrepo/rest/objects/raven/orderProxies/coverProxy
```

Where "iana-cover-proxy.ru" follows:

iana-cover-proxy.ru

```
PREFIX iana: <http://www.iana.org/assignments/relation/>

INSERT {
  <> iana:next </fcrepo/rest/objects/raven/orderProxies/page0Proxy>
} WHERE {
}
```

2) Establish both:

- "page0Proxy" has iana:prev of "coverProxy", and
- "page0Proxy" has iana:next of "page1Proxy":

```
curl -i -XPATCH -H"Content-Type: application/sparql-update" --data-binary @iana-page0-proxy.ru localhost:8080
/fcrepo/rest/objects/raven/orderProxies/page0Proxy
```

Where "iana-page0-proxy.ru" follows:

iana-page0-proxy.ru

```
PREFIX iana: <http://www.iana.org/assignments/relation/>

INSERT {
  <> iana:next </fcrepo/rest/objects/raven/orderProxies/page1Proxy> .
  <> iana:prev </fcrepo/rest/objects/raven/orderProxies/coverProxy>
} WHERE {
}
```

3) Establish "page1Proxy" has iana:prev of "page0Proxy":

```
curl -i -XPATCH -H"Content-Type: application/sparql-update" --data-binary @iana-page1-proxy.ru localhost:8080
/fcrepo/rest/objects/raven/orderProxies/page1Proxy
```

Where "iana-page1-proxy.ru" follows:

iana-page1-proxy.ru

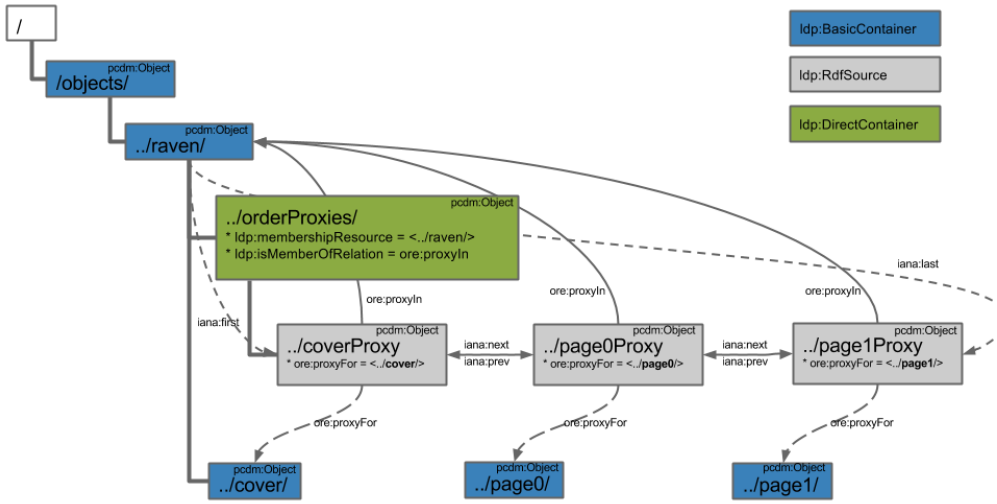
```
PREFIX iana: <http://www.iana.org/assignments/relation/>

INSERT {
  <> iana:prev </fcrepo/rest/objects/raven/orderProxies/page0Proxy>
} WHERE {
}
```

Ordering - Create First and Last

Ordering - Create First and Last

Finally, the very last step is to define from the book's perspective, the iana:first and iana:last pages of "raven/".



Establish both:

- "raven/" has iana:first of "coverProxy", and
- "raven/" has iana:last of "page1Proxy":

```
curl -i -XPATCH -H"Content-Type: application/sparql-update" --data-binary @iana-raven.ru localhost:8080/fcrepo
/rest/objects/raven/
```

Where "iana-raven.ru" follows:

iana-raven.ru

```
PREFIX iana: <http://www.iana.org/assignments/relation/>

INSERT {
  <> iana:first </fcrepo/rest/objects/raven/orderProxies/coverProxy> .
  <> iana:last </fcrepo/rest/objects/raven/orderProxies/page1Proxy>
} WHERE {
}
```

Ordering - Conclusion

Using LDP in conjunction with PCDM terms, we have established the ordering of pages within the book, "raven/".