# Reversing Collection Membership

## Background

In Hydra::PCDM, Hydra::Works, and CurationConcerns, Collections link to their member Objects using pcdm:hasMember. In Fedora 4, this is implemented as an IndirectContainer which contains a proxy for each member Object. When the RDF for the Collection is retrieved, those proxies are followed to the member Objects, which are used to generate the membership triples \(<\text{col1}> \text{pcdm:hasMember} \text{<obj1>}\).

Processing each individual proxy takes only a few milliseconds, but since Collections can have thousands or tens of thousands of members, that can add up to 30+ seconds to retrieve a Collection with 10,000 members.

See Real World Performance for more information on the underlying issue and discussion about it in the Fedora context. It's also worth noting that there is ongoing performance work in Fedora, which may improve the performance of listing members. There may also be other ways of working around the performance issues, such as background indexing.

## Using pcdm:memberOf

PCDM provides reciprocal predicates for linking from an Object to the Collections it is a member of, \(\langle\text{obj1} \text{pcdm:memberOf} \text{<col1>}\). Reversing Collection membership to link from Objects to Collections avoids the issue of having a large number of proxies to follow, since an Object is typically only a member of a small number of Collections. Using memberOf Collection membership for the 10,000 members discussed above resulted in a Collection and individual Objects that could all be retrieved in less than a second.

One downside to using pcdm:memberOf is that it is harder to get a list of all the Objects in a Collection. This will typically be handled by indexing all of the Objects in Solr, including their Collection membership. This provides a fast way to list the Collection's members. If the Objects aren't indexed in Solr or the index is inconsistent for some reason, the Collection can be retrieved from Fedora with the InboundReferences Prefer header. This is slow, but would typically only be needed to rebuild a Collection index.

## Implementation

A proof-of-concept implementation is available demonstrating using pcdm:memberOf to link from Objects to Collections:

- Hydra::PCDM: https://github.com/projecthydra/hydra-pcdm/pull/227
  - Adds a memberOf link from Objects to Collections
- CurationConcerns: https://github.com/projecthydra/curation_concerns/pull/888
  - Adds forms for editing memberOf links

This code is available as a starting point for adoption in your applications.

## Outstanding Questions

- Does this replace hasMember Collection membership, or should we have both kinds of Collection membership?
  - For example, if there are some scenarios where one Object might be present in a large number of Collections (such as a popular item in many user Collections).
  - Ordered collections might need to use hasMember in order to have a place to encode the order.
- If we keep both kinds of Collection membership, should the two kinds of members be grouped together, or kept separate?
  - If separate, how do we clearly label the different types of Collection and member lists
    - HasMemberCollection vs. MemberOfCollection?
    - UserCollection vs. DisplayCollection?
    - Something else?
- If Fedora performance improves, do we keep using hasMember Collections? Or are there other reasons to have memberOf Collections?