

Reference

Fedora

[Fedora 4 - Asynchronous REST API](#)
[Fedora 4 Federation - File System Connector](#)

PCDM/Sufia

[2015-03-09 Sufia Roadmap Call Notes](#)
[PCDM 4/17/2015 Call Info](#)

Sufia Sprint

[PCDM Sufia Sprint Planning Meeting](#)
[PCDM Sufia Sprint Documentation](#)
[Hydra::Works PCDM Diagram](#)
"Code Map" of a Work in Sufia (needs review for HydraDAM2 usefulness)
[Google doc describing phases for implementing PCDM](#) (with Sufia in mind)
[Github code changes from sprint](#)

- [Sufia](#)
- [Hydra-pcdm](#)
- [Hydra-works](#)

PCDM General Info

[LDP-PCDM-F4 In Action](#)
[Portland Common Data Model](#)
[PCDM Examples](#)
[Hydra and the Portland Common Data Model \(PCDM\)](#)
[Hydra Metadata Working Group](#)

PCDM with HydraDAM2

[IU Diagram of example MDPI object - simplified](#)
[IU Diagram of example MDPI object - complex](#)
[HydraDAM2 model with FileSet - common between WGBH and IU](#)
[HydraDAM2 PCDM WGBH Map \(Basic\)](#)
[HydraDAM2 PCDM WGBH Map \(Basic\)](#)

[2015-08-31 HydraDAM2 at IU Meeting Notes](#)

Metadata, RDF, etc.

Dynamic ad-hoc vocabularies

This can be done from the higher level RDF gem, not the rdf-vocab gem. See <https://github.com/ruby-rdf/rdf>.
In metadata code associated with models you would be able to do something like:

```
vocab = RDF::Vocabulary.new("http://iu.edu/vocab/0.1/")
```

Then you generate ad-hoc uri with method names, symbols, or strings:

```
vocab.name.to_s  
=> "http://iu.edu/vocab/0.1/name"  
  
vocab[:subject].to_s  
=> "http://iu.edu/vocab/0.1/subject"
```