Stores

- Interfaces
  - Interface WritableStore
    - Key delete(DigitalObject obj);
    - Key create(DigitalObject obj);
    - Key update(DigitalObject old, DigitalObject new)
    - Key rollback(DigitalObject new, DigitalObect old)
    - Status queryKey(Key key)
  - Interface ReadableStore
    - DigitalObject read(String pid);
  - Interface ReadableVersionableStore
    - Extends ReadableStore
    - DigitalObject read(String pid, String version)
    - List<String> getVersionNumbers(String pid)

Config

- One “CentralWrite” WritableStore
- One “CentralRead” ReadableStore
- All writes goes into CentralWrite
- All reads comes from CentralRead
- Versioning is a StorageLayer Concern
- Datastream storage is a StorageLayer concern
- Serialization is a StorageLayer concern

Object Model

- Pid
- Properties
- Datastreams
  - ID
  - Properties
  - Content (DataHandler?)

So, no datastream versioning.

Datastreams are Internal or External (Managed and Inline are storage layer concerns)

Plumbing Stores

- OneToN_WritableStore
  - Knows a number of WritableStores
  - Any writes are propagated to the entire list
- Async_WritableStore
  - Knows one WritableStore
  - Forwards any writes to this, async and returns
- Cachable_ReadableStore
  - Knows one ReadableStore
  - Keeps caches of read objects

Preservation Stores

- FoxmlFileSystemStore
  - Implements ReadableStore, WritableStore
  - The current one
- DbStore
  - Stores the objects in a database
- AdoreStore
- AkubraStore

Indexes

- TripleStore_WritableStore
- DB_WritableStore
- Messaging_WritableStore
- Gsearch_WritableStore
- ...

IntelligentStores

- AronsStore
  - Implements ReadableVersionableStore, WritableStore
  - Knows a number of stores
  - Perform intelligent decisions on where to read and write