Design - Transparent Persistence

Transparent persistence, or human-readable persistence, is the practice of keeping a copy of repository contents as files on disk.

Rationale

Different users have different rationales for wanting to access repository content as files on disk, such as:

- Making it easier to use disk-based tools and workflows
- Reducing the technology stack and skills required to recover repository content

Scenarios

There are a few different scenarios for keeping a copy of repository content on disk and keeping it in sync with the repository:

- The copy on disk is the only copy of the data, used by the repository as the primary storage
- The copy on disk is an additional copy of the data, updated synchronously during request processing
- The copy on disk is an additional copy of the data, updated asynchronously, e.g., by receiving JMS events and retrieving repository content
- A disk-like API is provided using FUSE or a similar tool that allows disk-based tools to work with the repository directly

Role in preservation

Having a copy of repository content on disk may enable a preservation workflow, but it is not a preservation strategy by itself. So transparent persistence is "preservation-enabling", allowing a disk-based preservation workflow to easily access the repository content.

Existing functionality

- fcrepo-serialization can be configured to serialize metadata as RDF files on disk.
- Using the default configuration, files larger than 4KB are stored on disk named after their SHA1 digest. So these files are already on disk, and can be matched with their associated metadata records using the SHA1.

Requirements

1. Fedora 4 resources shall be persisted as exploded BagIt bags, in a directory tree separate from the repository's primary storage
2. The directory structure of the Bags shall have a discoverable and predictable relationship with the resource's repository URL
3. RDF resources shall be persisted to disk in a client-defined RDF serialization, from the following options: application/ld+json, text/rdf+n3, application/rdf+xml, or text/turtle
4. NonRDF resources shall be associated with their respective RDF resources by the following **optional** modes:
    a. copying the NonRDF resource to the Bag's data directory
    b. hard-linking from the Bag's data directory to the NonRDF resource in the repository's primary storage (requires the Bag and repository storage to be on the same filesystem)
    c. sym-linking from the Bag's data directory to the NonRDF resource in the repository's primary storage
    d. creating a manifest with NonRDF repository URLs (holey bags)