2016-02-04 Fedora API Meeting

Date: Thursday February 4, 9am PST

Attendees

- Mark Bussey
- Adam Wead
- Longshou Situ
- Vivian Chu
- Tom Johnson
- Rob Sanderson
- Esme Cowles

Agenda

1. Goals of API (and SPI) work
    a. Defining an API with a clear spec, versioned independent of the implementation, etc.
        i. Including HTTP API, messaging
    b. Having a spec opens up the possibility of multiple implementations with different priorities
    c. We could use the existing (4.5.0) API as the baseline of the spec, and be thoughtful about changes going forward?
        i. Don't expect dramatic API change, but do expect some changes
        ii. Maybe we should codify existing API, and also plan a new version that improves parts of the API, and have a predictable process for moving from one to the other
    d. Would like more predictability of API changes
        i. There are release candidates available for 2-4 weeks, and testing against them would help identify breaking changes earlier
        ii. DCE supports multiple projects on multiple Fedora releases, and needs to manage changes
    e. How much of the API is stable? How do people know about upcoming changes?
        i. Some changes (e.g. removing JCR types) known about long in advance, could improve communication and predictability
        ii. The weekly Fedora committers call is a good way to know about changes, but too high an overhead for many people to participate in
        iii. Roughly quarterly meetings (HydraConnect, LDCX, etc.) would be more convenient
        iv. In favor of frequent releases, but not breaking changes
            1. Would like breaking changes to be less frequent and better communicated, to make it easier to test and adapt to them
2. Discussion of proposed services, in the context of Hydra
    a. CRUD
        i. Aligned with LDP, so already specified
        ii. Fedora's HTTP API docs also cover the particular implementation choices (e.g., Prefer headers supported)
        iii. Fedora complies with the LDP spec and wants to keep compliant
    b. Fixity checking
        i. On upload, you can provide a checksum and it will be verified
            1. Hydra doesn't support this now, but it could
            2. May want to have a slightly different approach: upload and checksum at the same time, and then compare checksums
        ii. On demand, you can check that the resource on disk matches the recorded checksum
    c. Versioning
        i. Existing versioning API Fedora-specific
        ii. The implementation is efficient and full-featured
        iii. Implementing it might complicate other implementations
        iv. The API spec should specify how an implementation that didn't support versioning would behave
            1. Or the API spec could require versioning, since many storage backends support versioning
        v. Would like to use the Memento API for version retrieval
            1. But there is no Memento spec for how to create versions
            2. Marmotta's Memento implementation isn't LDP-aligned, it just auto-versions triples
            3. Fedora could auto-version metadata to avoid needing to create them explicitly
                a. Non-versioning backends could just report the current version following the Memento spec
            4. But Fedora would need to have explicit versioning for binaries because storage concerns
            5. Fedora also has an API to restore versions
                a. But that could be a COPY from the old version URI to the current URI
        vi. ActiveFedora has limited support for versioning (files only), so need to support metadata versioning, subtree versioning
            1. Now would be a good time to change the API, since Hydra isn't really using it now
        vii. Would be good to include the broader LDP community into the versioning API discussion to encourage a LDP-wide versioning approach
        viii. Wouldn't mind having auto-versioning, but would still like to be able to tag/label specific versions
        ix. Don't want lots of extra versions of files because I version the metadata that links to it
            1. ActiveFedora can control this and decide when to create versions and/or label versions
            2. **ACTION**: Esme: Check whether creating a version of a tree also creates distinct versions of unchanged files
    d. Transactions
        i. Would like to consider all the changes in a transaction as a version
            1. Can do this now by opening a transaction, making changes, creating a version, and then committing the transaction
        ii. Somewhat awkward for RESTful API, so there is probably not an existing standard
        iii. The current API is a good strawperson
            1. Haven't heard any complaints about the API, non-Hydra clients are using it
        iv. Current discussion about what aspects of ACID Fedora supports
            1. Definitely Atomicity and Durability

            a. Atomicity might require all items to happen at the same time – would be hard to support in a distributed environment
            b. Want to make it as easy as possible to support diverse backends and scalability requirements
        2. Consistency and Isolation might be limited
            a. Different implementations might have different levels (e.g. snapshot isolation vs. read-uncommitted), and implementations should advertise what they support
    v. ACID is a set of guarantees for all updates, not just transactions, so it's important to consider them more broadly
e. Authorization
    i. Fedora provides authorization, but Hydra (for historical reasons) doesn't take advantage of it
    ii. Hydra does use WebACLs, but the implementation is different from what Fedora expects, so they are not compatible
        1. We should align them so Fedora could enforce Hydra's WebACLs for other clients
        2. Hydra also currently cannot provide the user who is making a request, which would be needed to enforce the WebACLs
            a. ActiveFedora would need to be refactored to allow per-request identification of the end user making the request
            b. **ACTION:** Adam and Esme will compare Fedora and Hydra WebACLs to see where they differ
    iii. Fedora authorization assumes either the client or the servlet container is handling authentication and group membership information
    iv. If there is an IndirectContainer, I shouldn't be able to use it to add triples to resources I don't have permission to write to
        1. **ACTION**: Rob will create a ticket to investigate this
3. Other API concerns
    a. Would like to have some kind of packaged version of all of the resources that make up a Work
        i. There is a Camel component that can sync updates to a triplestore, disk, etc. which might meet this need.
        ii. An RDF import/export functionality (like the current JCR/XML import/export functionality) would also meet this need, and could be a useful bulk edit API to address other concerns about the performance of editing multiple related resources.
    b. Muti-resource CRUD
        i. PCDM and Hydra Works mean that many users who used to have a single Fedora 3 object now have many Fedora 4 resources.
        ii. It would be great to have LDP community agreement on how this should work
        iii. We can all join the LDP next mailing list and discuss our approach, and then implement it in Fedora

Reference

- 2015 - 2016 Technical Priorities

Notes

A project tracking the Fedora API in Ruby: https://github.com/fcrepo4-labs/derby