Use Case Evaluation - F4 Asynchronous Storage

This page describes a potential set of questions to help evaluate use cases, present them in a clear and uniform way, and distill into requirements. The goal is to explore possible ways in which a common pattern for asynchronous interactions can be used to address each use case as concretely as possible, but falling short of providing a true specification or extension design.

Web Resources and Interactions

Would this asynchronous interaction expose any new resources, or involve existing ones (e.g. URI of a fedora resource, as defined via the Fedora HTTP API)? How do asynchronous and non-asynchronous aware clients interact with the resource?

Preconditions

Under what conditions should an asynchronous interaction be invoked? The current understanding of 'invoke' means "direct an HTTP request for a resource." If this particular use case uses 'invoke' in a different way, please define.

Examples of preconditions include:

- Interaction is invoked when a request is made to any repository resource.
- Interaction is invoked when a request to any resource contains content hinting at slow storage (e.g. is a POST, or contains a particular HTTP header)
- Interaction is invoked when a request is made to a web resource (URI) exposed by external service (i.e. API-X)

Deployment or Implementation notes

Are there any deployment or implementation-related details that may be relevant?

Examples include:

- We anticipate implementing this use case using the API Extension Architecture
- This pattern uses features of modeshape, and inherently needs to be installed with Fedora
- Implemented as camel routes that can be deployed into any osgi container

Proposed Requirements

What requirements may this use case place on the repository container, the core Fedora API, or the API extension architecture?

Value Proposition

What do you see as a potential value proposition of this use case in a common pattern for asynchronous interactions?

Examples include:

- The pattern for asynchronous interaction allows for a single public URI for a resource which abstracts away the details of its real location on my backend infrastructure
- The implementation provides a convenient way to allow for different asynchronous interactions for different storage backends
- The implementation provides a convenient way to distribute and deploy so that others can easily use or evaluate it (i.e. API-X)