Art Institute of Chicago Use Case - Object Services

Object Services

Title (Goal)	Support Fedora 3-style object classes (content models) - Object Services
Primary Actor	Repository architect & implementer
Scope	Data architecture and access
Level	High

Story (A paragraph or two describing what happens)	1. As a repository manager,
	  a. I will be able to associate services with "content models"
	    i. Example services include the generation of image derivatives, and the serving of images
	2. As a repository user,
	  a. I will be able to access resources via services that are associated with defined "content models"

Examples

a. Given the myns:image node above, I want to run a method (sequencer?) that creates/updates a thumbnail datastream under it every time the master datastream is updated.
b. The myns:image type constraints assures that master is always present so if a new myns:image is created, a master child is expected to run the method on.
c. Some users should be able to attach a pre-defined mymix:published_web mixin to the same object; when that happens, a sequencer is there waiting to create aweb_thumbnail, a web_small and web_large datastreams as children of the object.
d. Conversely, if the user removes mymix:published_web, the web_thumbnail,web_small and web_largedatastreams are destroyed.

Challenges

1. How do we handle the creation process so that the object does not go through validation before all mandatory children are ingested?

Issues / limitations

1. A mixin can define a mandatory type, but the mixin itself is not mandatory. If some editor removes the mixin, that would break the thumbnail method. In order to prevent that, access policies can be defined for who can remove or add a certain mixin; but for basic functionality not meant to be ever removed, a primary type would solve the problem more easily and elegantly.

- Stefano Cossu: tests should be made with transactions for Challenge #1 when https://www.pivotaltracker.com/s/projects/684825/stories/64058980 is fixed. If I am able to create a node with a primary type or mix-in that defines a mandatory child , and that child as well within a transaction, and the validation is performed ONLY after the transaction is committed, this would solve the issue.