Amherst: embargo handling

Title (Goal)	Amherst - embargo handling
Primary Actor	Developer
Scope	Component
Level	
Author	Unknown User (acoburn)
Story (A paragraph or two describing what happens)	For objects in the repository with embargos, an external process should be able to periodically check on those resources (e.g. weekly, monthly), updating the access controls (e.g. using WebAC) when the access restrictions go out of effect.

Web Resource interaction

As captured in the discussion below, there are two possible paths for this component. One approach would be to capture (filter) incoming requests, and, if an embargo is different than the authorization mode in effect, it could change that authorization mode to align with the embargo. This has the advantage that any embargo that has gone into or out of effect will be "immediately" translated into an authorization policy (by "immediately" I mean upon the next (filtered) request for the resource). The disadvantage is that every request would have the additional overhead of handling embargo dates; furthermore, a user request would require privilege escalation in order to change that resource, which is a situation that gives me pause.

The alternative approach would be to have a background batch process that is initiated either by an HTTP request or by running a script. That process would check on embargo-ed resources to verify whether the (WebAC) authorization mode should be changed. If it has, that change would be made. This script should not have to do a full repository scan: it should be able to generate a list of candidate resources to check based on some index: triplestore, database, solr, etc. That index, presumably, would be generated by a listener on Fedora's event stream.

Deployment or Implementation notes

This service would be deployed separately from fedora. I envision that this would require access to Fedora's HTTP API and event stream. It would probably make sense to implement this in a scripting language: python, ruby or go.

API-X Value Proposition

The primary use of this service would be for supporting an asynchronous background worker process for updating repository resources.