Design - Audit Service

This page lays out the considerations and activities surrounding the Fedora 4 Audit Service.

- Guiding Principles
- Actions
- Proposed Requirements
    - Functional - Write/Import
- Role Commitments
    - Development
    - Stakeholder
- Supplementary Documentation
    - UCSD

Guiding Principles

1. Any Fedora4 feature should be available through an API which is an implementation of LDP or an optional extension (ideally an existing standard)
2. Fedora4 features should favor existing tools over custom code
3. Fedora4 features should establish integration patterns where an implementation is not a part of the core code

Actions

	Action	Owner
1	Define required Audit Service queries	Dr. Arif Shaon
2	Perform comparative analysis of PROV-O vs. PREMIS-RDF (Reference: http://dcpapers.dublincore.org/pubs/article/view/3709)	Nick Ruest
3	Define repository events and event agents that should be recorded and supported by the Audit Service	Matt Critchlow Unknown User (escowles@ucsd.edu)
4	Define capability of the Audit Service REST-API	Doron Shalvi John Doyle

Proposed Requirements

Legend: (?) - Needs refinement, consensus, or removal

Functional - Write/Import

1. Audit service MUST ensure that all events minimally include the following information
    a. Event Agent
        i. When connecting through a service account, the Agent should use the standard header principal provider to pass the actual agent information.
    b. Event Date/Time
    c. Event Activity
    d. Event Entity
2. Audit service MUST be able to include/import events that were performed external to the repository.
    a. External events should be clearly labeled so they can be easily filtered from internal events
    b. Ideally, we would use a system that at the point of digitization, an object is assigned a globally unique identifier. At this point we could then trigger the CREATE event in a Fedora 4 system, likely a separate one that is only intended to store event data and provide the API /REST services as well as SPARQL interface for queries. The GUID would then stay with the digital object through its lifecycle which could be a significant amount of time before it is prepare for digital preservation. This is why we would use a separate Fedora 4 to store this data and would continue to do so even after ingest into the digital repository.
    The problem at hand is simply that our workflow is not to create an immediately ingest digital content into our Hydra repository. We could configure other systems to store this data and in some cases this is taking place. But this puts us in a position of having some event data stored in plain text files, some is stored in microsoft excel/access and then some is stored in various SQL instances. So as others have indicated, we do need the ability to bring in event data that occurred prior to ingest of the object into Fedora.
    It could also be possible that we use this separate Fedora 4 as the generator for the GUID making, what seems, like a smooth integration point between the instance that only handles event data and the instance that handles digital preservation. It also avoids a potential infinite loop. An event to say that we updated the record effectively updates the record which triggers an event to say we updated a record, again. For very sensitive materials, the level of event logging we would perform may be just that granular.
    The reason behind using a separate system for logging events is a fundamental principle of not having a system audit itself. So using a separate instance helps to maintain this separation, in my eyes it is separating the prison guards from the inmates, we should not trust

the inmate to count themselves. But more importantly, we may want to track inmates that are on their way into the system, not just after their arrival.
    c. One of the primary external use cases at UCSD is the transfer of objects to preservation management systems such as Chronopolis and Merritt. This will be triggered and performed external to Fedora, but the resulting Event metadata should be captured and linked to each Object for future querying. The common workflow would be as follows:
       1. Query Fedora for all Objects that have been created or modified since the last preservation transfer date.
       2. Attach Event metadata to each transferred Object in Fedora that includes: event type (PREMIS Event Types), Date, Agent, and optional outcome notes.
    d. Examples of external events,
        i. During ingest, audit service should accept audit log of an external application scanning a file for viruses.
        ii. During ingest, audit service should accept audit log of an external application validating a file's content against an external schema, profile, or using domain-specific validation tools.
        iii. Periodically, audit service should accept audit log of an external application, or a internal service provided by the repository itself, verifying a file's checksum.
    e. Audit service should accept audit log of an external application that moves a resource file.
3. Audit service MUST be able to maintain events for purged resources
4. Audit service MUST be able to perform with a large number of audit events
5. Audit service MUST not be able to remove events
6. Audit service MUST allow events to be stored separately from the repository resources themselves
7. Audit service MUST import events with RDF triples drawn from the specified ontologies

Functional - Read/Export

1. Audit service MUST export and answer queries in RDF format
2. Audit service MUST be able to export all events in the repository
3. Audit service MUST service queries that vary by:
    a. Single or all resources
    b. Date range
    c. Event type
    d. Agent
4. Audit service MUST provide a single search endpoint for all repository resource-related events
5. Audit service MUST provide a SPARQL-Query search endpoint
6. Audit service MUST be able to limit the number of audit events returned by a query, e.g., the first and most recent fixity check events

Non-Functional

1. Scale?
2. Security?
3. Performance?

Role Commitments

Development

- Mohamed Mohideen Abdul Rasheed
- Unknown User (escowles@ucsd.edu)

Stakeholder

- Matt Critchlow
- Nick Ruest
- Mark Jordan
- Unknown User (westgardja)

Supplementary Documentation

UCSD

Document
DAMS-Events-Agents-Final Version--20120106.docx
Event type controlled value list.pdf
Event Class and Properties

user-stories.pdf