

Repository Federation Service Usage Model: *repository federation – federated metadata registry*

Intellectual Property Rights: This document may be used under the *Creative Commons Attribution-ShareAlike 2.5 Australia License* [<http://creativecommons.org/licenses/by-sa/2.5/au/>].

The appropriate attribution for a derivative of this work is: “This document is derived from work created as part of the Federated Repositories for Education (FRED) Project within the Australian ADL Partnership Laboratory. Copyright © University of Southern Queensland and University of Memphis. 2007.”

Introduction

The Introduction provides a brief, standalone overview of the Service Usage Model. It is for a non technical reader. It is not for inclusion in a submission to the e-Framework, and thus may duplicate other material in the Service Usage Model Definition.

Repository federation is the process by which a set of repositories or content collections are combined into a single (logical) entity to support the discovery of the content contained within the individual repositories that form the federation. For the end user, the federation (as a software system) provides the single access point to the content in the federation, eliminating the need for a user to identify and individually access (search) each of the repositories to find content. The end user goes to the federation service end point to discover and identify content and its location (in the constituent repositories in the repository federation). The user may then go to an identified source repository to obtain the identified content.

This repository federation service usage model is based on a set of core assumptions:

- The repository federation includes a central registry (a resource) that provides the service end points for all services exposed by the service usage model.
- The central registry federates the metadata (contains copies) for the content from the repositories that participate in the repository federation.
- The content remains in the constituent repositories; only the metadata is federated. Access to the content is controlled by the individual repositories.
- The central registry also contains information about the repositories and content collections that are part of the federation.
- The repository federation is for learning content, described by appropriate standards-based learning content metadata.
- Access to the services of the repository federation may require authorization, and authentication processes may be applied to determine access conditions and to limit results.
- Access to the services of the participating repositories may require authorization, and authentication processes may be applied to determine access conditions and to limit results.
- Participation in the repository federation is closed; a repository must agree to a service level agreement to participate in the repository federation.
- The repository federation may be further federated into one or more federations of federations.

The overall model of the repository federation is illustrated in Figure 1.

The repository federation service usage model describes the business functions and underlying services (as service genres) and associated resources that define the structure and behaviour of a repository federation. The service usage model covers the creation of a repository federation (building the metadata registry), discovery of content from the metadata registry, and delivery of content from the central registry (but not the constituent repositories). Thus only functionality relating to objects as opposed to the overall system is covered in this service usage model. The operations and management of the infrastructure and core components of the repository federation, i.e., the central registry and its supporting systems, are covered separately.

The services exposed by the repository federation service usage model may be used to build end-user applications, e.g., a content discovery portal; or the services may be integrated into other applications, e.g., an authoring environment may add new content to a repository and register it with the repository federation; or a learner in a learning delivery environment may discover content from the registry and transfer it into the learning delivery environment.

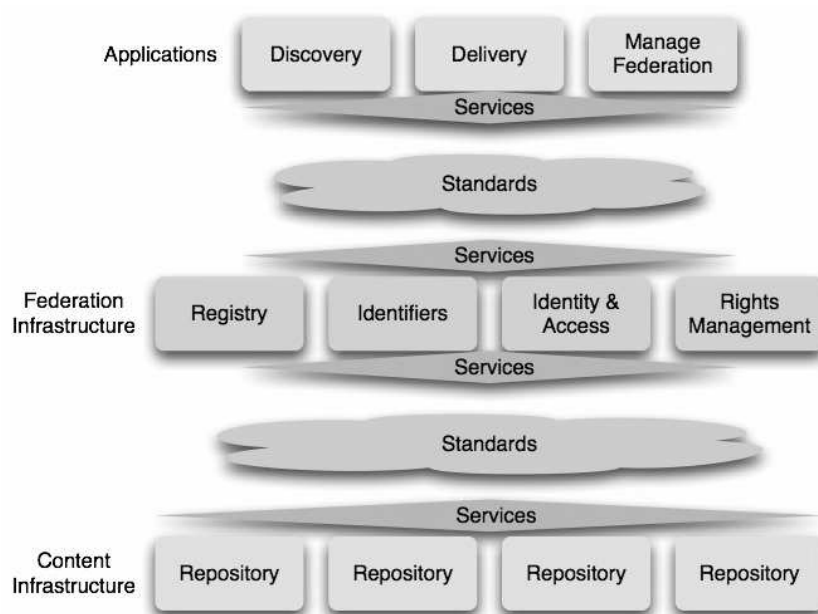


Figure 1: Repository Federation Model

Service Usage Model Definition

The Service Usage Model Definition is the complete formal documentation of the Service Usage Model for submission to the e-Framework. The elements included extend those from the current e-Framework template.

Rationale

The Rationale element presents the justification for the Service Usage Model, and explains its use intended by its authors.

The FRED Service Usage Model specifies the range of services expected to be used in repository federations in education in Australia for interacting with content objects (namely the functionality relating to objects, as distinct from the overall repository federation systems). The FRED project supports such repository federations by providing toolkits to develop the services those repositories require. So FRED uses the Service Usage Model as a functional requirement specification.

The Service Usage Model is also intended as an exemplar of a large Service Usage Model for e-Framework development.

Service Usage Model Metadata

The Service Usage Model Metadata contains basic labelling, classification and a version history for the Service Usage Model. It is expected that the e-Framework will modify these elements after submission.

Name

- FRED Service Usage Model Name: repository federation

- e-Framework Service Usage Model Registry Name: TBD

Classification

- Type: Domain
- Status: Unapproved
- Domains: Education (some of the system management functions might be appropriate for the IT domain)
- Domain Coverage: Single
- Deployment: Isolated
- Development Status: Proposed
- XOR: Genre
- Maturity: Immature
- Composition: Composite
- Purpose: Exemplar
- End Point: Transcoder
- Auth'd: Auth'd

Version

- FRED Version: 0.55
- e-Framework Service Usage Model Version: 1.0

Version History			
Version	Date	Author	Description
0.1	2007-02-14	DR	Initial partial version. hdl:FREDNA/61A190C19B4A49C19AC8348D476D1C2C
0.2	2007-02-21	DR	Partial version, most functionality and services.
0.3	2007-02-25	DR	Complete draft. See "to do" list at the end of the document. Unedited
0.31	2007-02-26	DR	Complete draft. See "to do" list at the end of the document. Edited
0.32	2007-03-14	NN	Queries, minor editorial
0.4	2007-03-26	DR	Edits, new diagrams (4, 5), modified some service descriptions, query responses, embedded Visio diagrams (1,2,4,5,6, not 3)
0.45	2007-04-10	DR	Added element descriptions, shuffled sections, editorial, accepted some changes from V0.4
0.46	2007-04-13	NN	Added a rationale and business analysis component
0.47	2007-04-25	NN	Editorial
0.48	2007-04-27	NN	Editorial: global change of service + resource nomenclature to service {resource}; conflate query functionalities; align diagram of business processes to rest of business analysis
0.49	2007-05-07	DR	Editorial, Accepted lots of changes. Deleted notes that have been resolved.
0.5	2007-05-09	NN	Editorial, tidying up outstanding comments, proceeded with conflation of similar functionalities.
0.51	2007-05-14	DR	More editorial and cleanup
0.52	2007-05-24	NN	Cleanup: Service Genre listings has genres not genre instances; notation names data source classes not instances; functionalities unconfated; replaced core/value-added with base/optional; moved consumer SUMs to Known Uses and out of Related SUMs.
0.53	2007-05-25	DR	Editorial. Revise Genre listing. Changes to get ready to submit to the eFramework.
0.54	2007-06-12	NN	Updated diagrams, prepared for three-way split of document (content, system provisioning, system management)
0.55	2007-06-13	NN	Draft for distribution to stakeholders; changed most diagrams, excised system functionality.

Notation

The Notation element includes conventions used to describe the Service Usage Model.

The words **MUST**, **MUST NOT**, **REQUIRED**, **SHALL**, **SHALL NOT**, **SHOULD**, **SHOULD NOT**, **RECOMMENDED**, **MAY**, and **OPTIONAL** in this document are to be interpreted as described in [RFC 2119].

Definitions of service usage model-specific terms used throughout (e.g., content object, collection, collection object, metadata object, registry, registry object, repository, repository object) are given in the service usage model glossary. Other terms, e.g., client, provider, resource, are used as defined in the e-Framework glossary.

The service usage model description that follows uses e-Framework service genre description elements as of 2006-12-09, updated to include draft e-Framework service classifications (V0.5) and revised to incorporate other elements based on FRED project experience needed to document the service usage model.

Service genres are named using a proposed e-Framework naming convention, consisting of the behaviour or function provided by the service genre (expressed as a verb) augmented with the name of the target resource class (or object class) in curly brackets; e.g., `authenticate {agent}` (applies to any agent), `obtain {content repository}` (applies to any content repository in the federation), `validate {metadata object}` (applies to any metadata object). Service genres and resources are also classified by type, e.g., **BASE** (essential), **VALUE ADDED** (not essential), **EXTERNAL** (exposed to users, default if not noted), or **INTERNAL** (not exposed to users).

Service usage models are classified by type, e.g., **BASE** (essential), **VALUE ADDED** (not essential). Related service usage model that describes an Application which relies on the service usage model are denoted **APPLICATION**. If not classified, the related service usage model may be used by, or may use this service usage model.

Items are tagged and identified in the metadata element using names assigned by the FRED project. Formal e-Framework names will be assigned by the e-Framework.

Prior to e-Framework publication, the version numbers shall be less than 1.0 and a new identifier is not assigned to each version. Revisions subsequent to publication by the e-Framework require a new identifier be assigned to each version.

Description

The Description element is an informal, standalone, non technical narrative description of the Service Usage Model (problem, process, business level capabilities and workflow).

The repository federation service usage model describes the structure of a collection of collaborating service genres and service usage models (along with their associated managed resources) used to provide all the functionality needed to implement a repository federation for content discovery.

The repository federation is an organized collection of constituent repositories and a central registry used to discover content from the repositories that participate in the repository federation. The service genres within the service usage model provide the functionality to manage the resources that support the capabilities of the federation.

The repository federation service usage model provides capabilities to:

- Add (and manage) the description of repositories to the repository federation (to enable discovery based on repository descriptions).
- Add (and manage) the description of collections to the repository federation (to enable discovery based on collection descriptions).
- Add (and manage) the description of content objects through metadata objects to the repository federation.
- Build and maintain a central registry from repository descriptions, collection descriptions, and metadata objects. The central registry supports all discovery processes.
- Discover content objects in the repositories that participate in the repository federation through search of the central registry (based on their metadata objects).
- Discover repositories that participate in the repository federation through search of the central registry.

- Discover collections in the repository federation through search of the central registry.
- Augment discovery metadata (either from user input or by computation and analysis of existing metadata).
- Access and obtain metadata objects describing content objects through the central registry.
- Access and obtain content objects discovered through the central registry.
- Expose the central registry to harvest so that it can participate in a federation of federations.

The full deployment of a repository federation service usage model also requires capabilities to:

- Provision and manage the central registry and its constituent parts.

Much of this functionality is not specific to repository generation, but applies to general management of repositories. This functionality is outlined in the distinct service usage models repository provisioning, manage discovery service, and manage repository, profiled to the requirements of repository federation. Those system-oriented service usage models are to be combined with this service usage model through a “wrapper” service usage model (“repository federation overview”).

The basic structure of the repository federation service usage model is a collection of collaborating service genres used to build and access the central registry resources. The service usage model defines a set of business functions that can be exposed to users; each function maps to a service end point for the repository federation. Each service end point responds to a user request and either adds information to the central registry or retrieves information from it. Individual and combined service genres, with a prescribed workflow, provide the described functionality.

The overall business functionality of the repository federation service usage model is illustrated in Figure 2 and Figure 3.

The repository federation service usage model is used with other service usage models, including an identity management service usage model and an identifier service usage model.

The defined repository federation service usage model must be specialized for a community of practice. This specialization will add policy, defined data models, standards used and behaviour constraints. These will be used to refine the service genres into service expressions and aid in developing the design for the complete application suite needed to implement and operate a repository federation.

Business Process Modelling

The Business Process Modelling element lists the business functions to be supported by the Service Usage Model. Business process modelling provides necessary context to motivate and understand the description in the remainder of the Service Usage Model. The business analysis motivating the Service Usage Model should be summarised within the Service Usage Model (to make it readable as a self-contained object), with fuller discussion possibly outside the Service Usage Model. Functions should be listed as groups, in business terms. The description should not be encumbered with either technical services or detailed workflows.

Under *Functionality*, services are split into Registry Domain functions, performed by users, and Registry System functions, used to manage the federation. Only Registry Domain functionality corresponds to higher-level business requirements; moreover, Registry System functions are not discussed in this service usage model. Registry Domain functions in turn are divided into Content Management, Content Discovery, and Content Delivery. Content Management functions are triggered by repository managers and registry managers. Content Discovery and Delivery functions are normally triggered by end users; they may also be triggered by an external registry seeking to add the registry to its own federation.

The following higher-level business functions have been identified as relevant to FRED, and are supported by services in the Service Usage Model.

NB: The Service Usage Model may also be used to support additional high-level business functions that are outside of the scope of FRED.

1. Content Management:

- *Register Repository with Registry:* A repository manager initiates a request that their repository joins a federation. Once the request is accepted, metadata about the repository is ingested into the registry. This enables all subsequent interactions of the repository with the registry, so that the repository is treated as a member of the federation.
- *Register Managed Collection with Registry:* A managed collection has access and metadata profiles which are possibly distinct from those of the repositories it is hosted on. The managed collection is hosted by repositories participating in the federation. Metadata about the managed collection is ingested into the registry. As a result, end users can be informed about the requirements specific to the managed collection when they discover any objects belonging to it.
- *Update Registration of Repository with Registry:* Metadata describing a repository has been ingested into the registry in order for the repository to participate in the federation. The content of the metadata becomes out of date. The updated version of the metadata is ingested into the registry, so that the registry continues to inform users accurately of the status of the repository.
- *Update Registration of Managed Collection with Registry:* Metadata describing a managed collection has been ingested into the registry in order for the federation to honour the managed collection requirements. The content of the metadata becomes out of date. The updated version of the metadata is ingested into the registry, so that the registry continues to inform users accurately of the managed collection requirements.
- *Push Metadata of Item into Registry:* A participating repository initiates a request for the registry to ingest the description of a content item the repository holds. When the registry complies, the ingestion results in the content item being registered in the registry, and discoverable through the registry by end users.
- *Pull Metadata of Item into Registry:* The registry initiates a request to ingest the description of a content item from a participating repository. This ingestion (known as *Harvest*) results in the content item being registered in the registry, and discoverable through the registry by end users. Harvest requests are typically periodic, and target all content items added or updated since the last harvest request.
- *Deactivate Metadata of Item in Registry:* A description of a content item has been ingested into the registry, allowing the item to be discovered through the registry. A repository or registry manager determines that the description should no longer be available through the registry, or because an alternate description (possibly with a distinct metadata schema) should be used for discovery instead. The description is made unavailable for discovery. It is not necessarily deleted.
- *Update Metadata of Item in Registry:* A description of a content item has been ingested into the registry, allowing the item to be discovered through the registry. A repository or registry manager determines that this description no longer accurately describes the item. An updated version of the description is ingested into the registry. As a result, the updated version is used for discovery of the item, leading to greater accuracy.
- *Classify Item:* classification taxonomy for items available through the registry is available. The classification can be used to enable discovery of items categorised under certain taxa (either through search or browse). An item is not classified according to the taxonomy on the repository holding it. A classification of the item according to the taxonomy is added as metadata to the registry by some authorised party. (This may be a manager, or a third party.) The new metadata can now be used for discovery by end users through the registry.
- *Annotate Item:* A user wishes to attach an annotation to an item available through the federation, as a general description. This annotation may be used in discovery. The user is not necessarily authorised to add annotations to the item as metadata stored on the repository holding the item. The user is authorised to add annotations to the item description stored on the registry. The new metadata can now be used for discovery by end users through the registry.
- *Recommend Item:* A user wishes to attach a recommendation (a Boolean value, or a taxon from a well-defined taxonomy) to an item available through the federation, as a general description. This recommendation may be used in discovery. The user is not necessarily authorised to add recommendations to the item as metadata stored on the repository holding the item. The user is authorised to add recommendations to the item description stored on the registry. The new metadata can now be used for discovery by end users through the registry.
- *Rate Item:* A user wishes to attach a rating (a numeric value) to an item available through the federation, as a general description. This rating may be used in discovery. The user is not necessarily authorised to add

ratings to the item as metadata stored on the repository holding the item. The user is authorised to add ratings to the item description stored on the registry. The new metadata can now be used for discovery by end users through the registry.

2. Content Discovery:

- *Discovery Item through Search on Registry:* An end user wishes to find out which objects registered in the federation (if any) conform to certain criteria. They issue a search request to the registry, which returns descriptions of all matching items. The descriptions may be used by the end user to request content delivery.
- *Discover Item through Browse on Registry:* An end user wishes to view a defined subset of objects registered in the federation. This subset may be an arbitrary subset; a thematic subset; or the entire set of objects. They issue a browse request to the registry, which returns descriptions of all items within the subset. The descriptions may be used by the end user to request content delivery.
- *Syndicate Item from Registry:* An end user subscribes to the syndication of metadata from the registry. As a result, whenever an item is newly registered in the registry (that is, a metadata description for an item is ingested into the registry, where none was previously available), the subscribing end user may become aware of the availability of the item through the federation by polling the registry. A notification may also be sent when there is a significant (rather than routine) update to the metadata description of an item in the registry.

3. Content Delivery:

- *Expose Registry for Harvesting:* The registry manager decides to have their registry participate in a federation of registries. The manager enables harvesting of items registered within the registry, so that the registry of registries can initiate a harvest request on this registry.
- *Obtain Item from Repository:* An end user has identified an item available on the federation to be accessed. This identification may have been undertaken through a content discovery service, or through a citation of the item independently available. The end user requests delivery of the item from the registry. The registry brokers the request as an arrangement between the user and the repository holding the item. Authentication and authorisation are arranged by the repository.
- *Obtain Metadata of Item from Repository:* An end user has identified an item available on the federation as being of interest. The metadata available on the item through the registry is not adequate for the user. The user requests any further metadata available on the item through the repository holding the item. The registry brokers the request as an arrangement between the user and the repository holding the item. Authentication and authorisation are arranged by the repository.

Service Usage Model Diagram

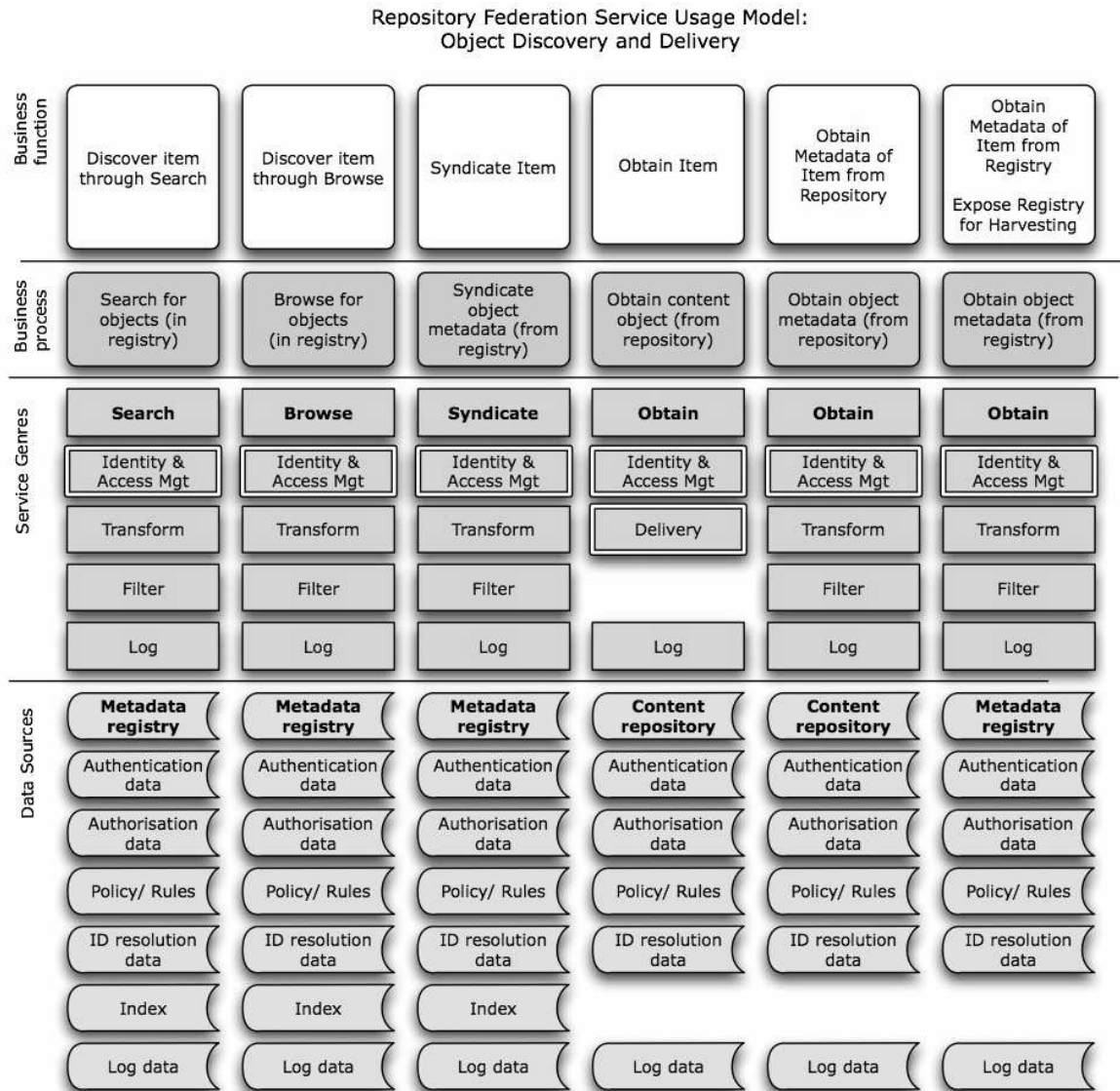


Figure 2: Service Usage Model (Object/Content Discovery & Delivery)

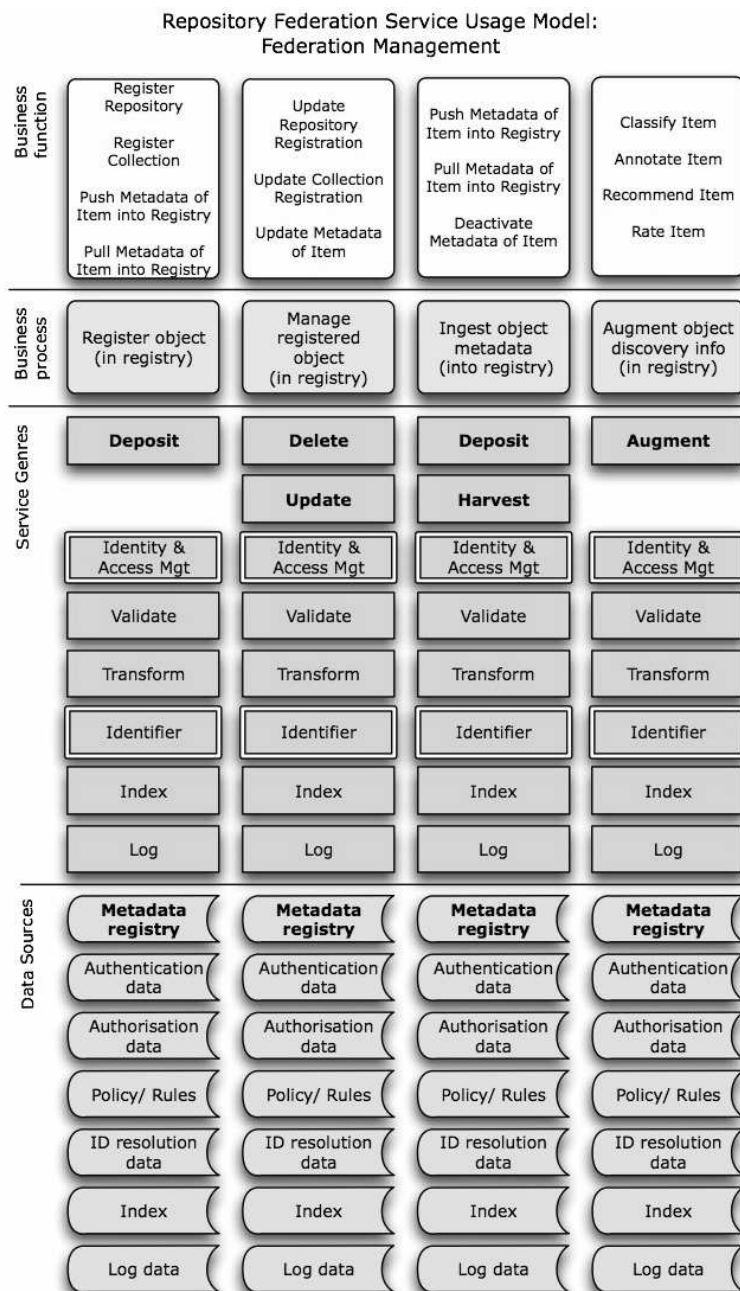


Figure 3: Service Usage Model (Federation/Content Management)

Usage Scenarios

The Usage Scenarios element is an informal, non technical description of how the Service Usage Model is used in support of business processes. An illustration of business workflows, expressed using services, is included. An illustration of an application using the components of the Service Usage Model may be included (but not a description of the design of the application). Full business process modelling and detailed diagrams are excluded. No critical or essential information required to understand the Service Usage Model should be included.

Details of the repository federation usage scenarios are documented separately in FRED documentation [links to latest versions of use cases].

Applicability

The Applicability element details how and when the Service Usage Model is used or not used. It represents specific constraints and assumptions on the use of the Service Usage Model. It is more specific and normative than the informal Usage Scenarios. No critical or essential information required to understand the Service Usage Model should be included.

The repository federation service usage model is applicable for discovery from a (central) metadata registry built through federation of metadata from repositories. It is not directly applicable to discovery via federated search.

The repository federation service usage model assumes a common metadata model used to federate content from all repositories that participate in the repository federation. The process of federating repositories that use different metadata models is not described.

The repository federation service usage model assumes that all the repositories that participate in the repository federation do so in a managed way, and agree to service-level agreements to participate in the repository federation, i.e., the repository federation is open only to defined, participating repositories. The nature of the service-level agreement is not specified.

The repository federation service usage model focuses on describing the capabilities of a registry that federates its constituent repositories. While this metadata registry may be further federated (a federation of federations) and while the service usage model describes the functionality to expose the registry for further federation, such details are not described.

The repository federation service usage model defines a trust environment. External access to the services and resources that are within the trust environment requires authorization or access controls. Within the trust environment, access to individual services and resources from other services within the trust environment SHOULD NOT require authorization or access controls.

The repository federation service usage model assumes that content remains in the original constituent repositories and that these repositories control access to the content. Access to and delivery of content from the constituent repositories is not part of the service usage model. Management and enforcement of digital rights, either within the registry or the constituent repositories, is not part of the service usage model. However, access to and delivery of metadata describing access, delivery, and digital rights is part of the service usage model, and is mediated through the repository and the collection object. Digital rights may be included within metadata objects.

The repository federation service usage model is based on a common identifier model and system used throughout to identify all objects and provide resolution services.

The repository federation service usage model is applicable both for creating end-user applications and imbedding repository federations in other applications and tools.

The repository federation service usage model does not specify functional requirements for portals, human user interfaces or capabilities (customisation, personalisation, help, etc.) used to develop end-user applications.

The repository federation service usage model includes rights licences as a type of managed resource. The service usage model does not include functionality for defining or managing digital rights and rights licences.

The repository federation service usage model does not specify security requirements.

Functionality

The Functionality element details and illustrates the behaviours provided by the Service Usage Model, in terms of services, workflows, messages, resources, data objects. It is not a technical description of the Service Usage Model, but it must provide sufficient information to develop the Structure & Organization of the Service Usage Model and to evaluate conformance of the Service Usage Model to the stated behaviours. It should not include implementation-specific information.

The functionality required in the deployment of a repository federation can be split into two major groups:

- *Registry Domain Functions*: functions performed by users (end users, services) dealing with content objects in repositories and registries, including building federations and the discovery and retrieval of content objects from federations and their constituent repositories. These are further subdivided as:
 - *Content Management Functions*: Creating and managing metadata objects within a repository federation.
 - *Content Discovery Functions*: Discovery of content objects from a repository federation.
 - *Content Delivery Functions*: Retrieval and access to content objects discovered through a repository federation.
- *Registry System Functions*: functions applied to the registry infrastructure management and operations. These are further subdivided as:
 - *Registry Provisioning Functions*: Creating a registry (and its constituent parts) to enable building a repository federation.
 - *Registry Management Functions*: Managing the operations of a registry.

This service usage model only covers Registry Domain functions.

These functions are further classified as:

- BASE (essential capabilities) versus VALUE ADDED (desirable but not essential repository federation services). The distinction is based both on technical requirements and community requirements.

NB: The functionality described does not imply a particular implementation strategy.

The mapping of high-level business requirements to system functionality is summarised as follows:

Content Delivery

High-Level Business Functions	System Functionality
<i>Discover Item through Browse on Registry</i>	<i>Obtain Metadata Object</i>
<i>Discover Item through Search on Registry</i>	<i>Obtain Metadata Object</i>
<i>Expose Registry for Harvesting</i>	<i>Expose Collection Object Metadata</i>
<i>Expose Registry for Harvesting</i>	<i>Expose Metadata Object from Registry</i>
<i>Expose Registry for Harvesting</i>	<i>Expose Repository Object</i>
<i>Obtain Item from Repository</i>	<i>Obtain Collection Object</i>
<i>Obtain Item from Repository</i>	<i>Obtain Content Object</i>
<i>Obtain Item from Repository</i>	<i>Obtain Repository Object</i>
<i>Obtain Metadata of Item from Repository</i>	<i>Obtain Collection Object</i>
<i>Obtain Metadata of Item from Repository</i>	<i>Obtain Repository Object</i>

Content Discovery

High-Level Business Functions	System Functionality
—	<i>Browse Collection Object Registry</i>
—	<i>Browse Repository Object Registry</i>
—	<i>Syndicate Collection Object Registry</i>
—	<i>Syndicate Repository Object Registry</i>
<i>Discover Item through Browse on Registry</i>	<i>Browse Content Object Registry</i>
<i>Discover Item through Search on Registry</i>	<i>Query Content Object Registry</i>
<i>Obtain Item from Repository</i>	<i>Query Collection Object Registry</i>
<i>Obtain Item from Repository</i>	<i>Query Repository Object Registry</i>
<i>Obtain Metadata of Item from Repository</i>	<i>Query Collection Object Registry</i>
<i>Obtain Metadata of Item from Repository</i>	<i>Query Repository Object Registry</i>
<i>Syndicate Item from Registry</i>	<i>Syndicate Content Object Registry</i>

Content Management

High-Level Business Functions	System Functionality
—	<i>Archive Content</i>
—	<i>Augment Collection Object Discovery Information</i>

High-Level Business Functions	System Functionality
—	<i>Augment Repository Object Discovery Information</i>
—	<i>Delete Collection Object</i>
—	<i>Delete Repository Object</i>
—	<i>Escrow Content</i>
<i>Annotate Item</i>	<i>Augment Metadata Object Discovery Information</i>
<i>Annotate Item</i>	<i>Generate Content Object Metadata Object</i>
<i>Classify Item</i>	<i>Augment Metadata Object Discovery Information</i>
<i>Classify Item</i>	<i>Generate Content Object Metadata Object</i>
<i>Deactivate Metadata of Item in Registry</i>	<i>Change Status Metadata Object</i>
<i>Deactivate Metadata of Item in Registry</i>	<i>Delete Metadata Object</i>
<i>Pull Metadata of Item into Registry</i>	<i>Expose Metadata Object to Registry</i>
<i>Pull Metadata of Item into Registry</i>	<i>Generate Content Object Metadata Object</i>
<i>Pull Metadata of Item into Registry</i>	<i>Ingest/Harvest Metadata Object Repositories</i>
<i>Push Metadata of Item into Registry</i>	<i>Generate Content Object Metadata Object</i>
<i>Push Metadata of Item into Registry</i>	<i>Ingest/Deposit Metadata Object</i>
<i>Rate Item</i>	<i>Augment Metadata Object Discovery Information</i>
<i>Rate Item</i>	<i>Generate Content Object Metadata Object</i>
<i>Recommend Item</i>	<i>Augment Metadata Object Discovery Information</i>
<i>Recommend Item</i>	<i>Generate Content Object Metadata Object</i>
<i>Register Managed Collection with Registry</i>	<i>Register Collection</i>
<i>Register Repository with Registry</i>	<i>Register Repository</i>
<i>Update Metadata of Item in Registry</i>	<i>Move Content to Collection</i>
<i>Update Metadata of Item in Registry</i>	<i>Move Content to Repository</i>
<i>Update Metadata of Item in Registry</i>	<i>Update Metadata Object</i>
<i>Update Metadata of Item in Registry</i>	<i>Update Repository Object</i>
<i>Update Registration of Managed Collection with Registry</i>	<i>Update Collection Object</i>

Content Management Functions

Register Repository: [BASE]

Add a repository to the repository federation, i.e., add a repository object to the repository registry.

Basic workflow steps (without error handling) are:

- Provide descriptive information about the repository, i.e., the repository object to be registered (the request). The description SHOULD conform to a designated standard.
- Authenticate and authorize (via access controls and business rules) the request.
- Validate the request and submitted repository object information.
- Transform the descriptive information (schema transform plus external to internal representation).
- Assign an identifier to the repository object.
- Add the repository to the repository federation. Update the catalogue of repositories in the repository federation. The catalogue update may require indexing to aid in discovery.
- Log the activity.
- Return status results.
- Return repository object identifier.

End Point: deposit {repository registry}

Supporting Service Genre(s): authenticate {agent}, authorize {access control policy/authorization data}, authorize {business rule/policy data}, validate {repository object}, schema transform {repository object}, register identifier {object}, index {repository registry}, activity log {workflow log}

Supporting Service Usage Model(s): identity, identifier, repository storage manager

Primary Resource: repository registry

Secondary Resource(s): authentication data, access control policy/authorization data, business rule/policy data, identifier resolution data, index data, workflow log

Object(s): repository object

Business Process: *Register Repository with Registry*

Delete Repository Object: [BASE]

Delete a repository from the repository federation, i.e., delete a repository object from the repository registry.

Basic workflow steps (without error handling) are:

- Identify the repository object for deletion (via repository object identifier) (the request).
- Authenticate and authorize (via access controls and business rules) the request.
- Validate the request and submitted repository object identifier.
- Delete the repository from the repository federation. Update the catalogue of repositories in the repository federation. The catalogue update may require reindexing to aid in discovery.
- Update repository object identifier resolution. (Indication that object is now unavailable.)
- Log the activity.
- Return status results.

End Point: delete {repository registry}

Supporting Service Genre(s): authenticate {agent}, authorize {access control policy/authorization data}, update identifier resolution data {object}, authorize {business rule/policy data}, index {repository registry}, activity log {workflow log}

Supporting Service Usage Model(s): identity, identifier, repository storage manager

Primary Resource: repository registry

Secondary Resource(s): authentication data, access control policy/authorization data, business rule/policy data, identifier resolution data, index data, workflow log

Object(s): repository object identifier

Business Process: —

Update Repository Object: [VALUE ADDED]

Change the description of a repository in the repository federation, i.e., change the description of a repository object in the repository registry. *Update Repository Object* is a monolithic operation, and is not applicable to incremental additions to subparts of an object (cf., *Augment Repository Object Discovery Information*).

Basic workflow steps (without error handling) are:

- Identify the repository object for update (via repository object identifier) and provide descriptive information about the repository, i.e., updates to the repository object (the request).
- Authenticate and authorize (via access controls and business rules) the request.
- Validate the request and submitted repository object information.
- Transform the descriptive information (schema transform plus external to internal representation).
- Update the repository description in the repository federation. Update the catalogue of repositories in the repository federation. The catalogue update may require reindexing to aid in discovery.
- Log the activity.
- Return status results.

End Point: update {repository registry}

Supporting Service Genre(s): authenticate {agent}, authorize {access control policy/authorization data}, authorize {business rule/policy data}, validate {repository object}, schema transform {repository object}, index {repository registry}, activity log {workflow log}

Supporting Service Usage Model(s): identity, identifier, repository storage manager

Primary Resource: repository registry

Secondary Resource(s): authentication data, access control policy/authorization data, business rule/policy data, identifier resolution data, index data, workflow log

Object(s): repository object, repository object identifier

Business Process: *Update Metadata of Item in Registry*

Augment Repository Object Discovery Information: [VALUE ADDED]

Add to the description of a repository in the repository federation, i.e., add metadata and descriptive information to a repository object in the repository registry to aid in discovery (rank, rate, classify, annotate, recommend) through user-provided information. *Augment Repository Object Discovery Information* applies when there is an incremental addition of an entire class of metadata description for the repository object. When there is an alteration or deletion of ingested metadata, *Update Repository Object* is applicable instead.

Basic workflow steps (without error handling) are:

- Identify the repository object for augmentation (via repository object identifier) and provide descriptive information about the repository, i.e., the augmentation information for the repository object (the request).
- Authenticate and authorize (via access controls and business rules) the request.
- Validate the request and submitted repository object information.
- Transform the descriptive information (schema transform plus external to internal representation).
- Augment the repository description in the repository federation. Update the catalogue of repositories in the repository federation. The catalogue update may require reindexing to aid in discovery.
- Log the activity.
- Return status results.

End Point: augment {repository registry}

Supporting Service Genre(s): authenticate {agent}, authorize {access control policy/authorization data}, authorize {business rule/policy data}, validate {repository object}, schema transform {repository object}, index {repository registry}, activity log {workflow log}

Supporting Service Usage Model(s): identity, identifier, repository storage manager

Primary Resource: repository registry

Secondary Resource(s): authentication data, access control policy/authorization data, business rule/policy data, identifier resolution data, index data, workflow log

Object(s): repository object, repository object identifier

Business Process: —

Register Collection: [BASE]

Add a collection to the repository federation, i.e., add a collection object to the collection registry.

Basic workflow steps (without error handling) are:

- Provide descriptive information about the collection and its relationship to repositories, i.e., the collection object (the request). The description SHOULD conform to a designated standard.
- Authenticate and authorize (via access controls and business rules) the request.
- Validate the request and submitted collection object information.
- Transform the descriptive information (schema transform plus external to internal representation).
- Assign an identifier to the collection object.
- Add a collection to the repository federation. Update the catalogue of collections in the repository federation. The catalogue update may require indexing to aid in discovery.
- Log the activity.
- Return status results.
- Return collection object identifier.

End Point: deposit {collection registry}

Supporting Service Genre(s): authenticate {agent}, authorize {access control policy/authorization data}, authorize {business rule/policy data}, validate {collection object}, schema transform {collection object}, register identifier {object}, index {collection registry}, activity log {workflow log}

Supporting Service Usage Model(s): identity, identifier, repository storage manager

Primary Resource: collection registry

Secondary Resource(s): authentication data, access control policy/authorization data, business rule/policy data, identifier resolution data, index data, workflow log

Object(s): collection object

Business Process: *Register Managed Collection with Registry*

Delete Collection Object: [BASE]

Delete a collection from the repository federation, i.e., delete a collection object from the collection registry.

Basic workflow steps (without error handling) are:

- Identify the collection object for deletion (via collection object identifier) (the request).
- Authenticate and authorize (via access controls and business rules) the request.
- Validate the request and submitted collection object identifier.
- Delete the collection from the repository federation. Update the catalogue of collections in the repository federation. The catalogue update may require reindexing to aid in discovery.
- Log the activity.
- Return status results.

End Point: delete {collection registry}
Supporting Service Genre(s): authenticate {agent}, authorize {access control policy/authorization data}, authorize {business rule/policy data}, index {repository registry}, activity log {workflow log}
Supporting Service Usage Model(s): identity, identifier, repository storage manager
Primary Resource: collection registry
Secondary Resource(s): authentication data, access control policy/authorization data, business rule/policy data, identifier resolution data, index data, workflow log
Object(s): collection object identifier
Business Process: —

Update Collection Object: [VALUE ADDED]

Change the description of a collection in the repository federation, i.e., change the description of a collection object in the collection registry.

Basic workflow steps (without error handling) are:

- Identify the collection object for update (via collection object identifier) and provide descriptive information about the collection, i.e., updates to the collection object (the request).
- Authenticate and authorize (via access controls and business rules) the request.
- Validate the request and submitted collection object information.
- Update the collection in the repository federation. Update the catalogue of collections in the repository federation. The catalogue update may require reindexing to aid in discovery.
- Log the activity.
- Return status results.

End Point: update {collection registry}
Supporting Service Genre(s): authenticate {agent}, authorize {access control policy/authorization data}, authorize {business rule/policy data}, validate {collection object}, schema transform {collection object}, index {collection registry}, activity log {workflow log}
Supporting Service Usage Model(s): identity, identifier, repository storage manager
Primary Resource: collection registry
Secondary Resource(s): authentication data, access control policy/authorization data, business rule/policy data, identifier resolution data, index data, workflow log
Object(s): collection object, collection object identifier
Business Process: *Update Registration of Managed Collection with Registry*

Augment Collection Object Discovery Information: [VALUE ADDED]

Add to the description of a collection in the repository federation, i.e., add metadata and descriptive information to a collection object in the collection registry to aid in discovery (rank, rate, classify, annotate, recommend).

Basic workflow steps (without error handling) are:

- Identify the collection object for augmentation (via collection object identifier) and provide descriptive information about the collection, i.e., augmentation information for the collection object (the request).
- Authenticate and authorize (via access controls and business rules) the request.
- Validate the request and submitted collection object information.
- Transform the descriptive information (schema transform plus external to internal representation).
- Augment the collection in the repository federation. Update the catalogue of collections in the repository federation. The catalogue update may require reindexing to aid in discovery.
- Log the activity.
- Return status results.

End Point: augment {collection registry}
Supporting Service Genre(s): authenticate {agent}, authorize {access control policy/authorization data}, authorize {business rule/policy data}, validate {collection object}, schema transform {collection object}, index {collection registry}, activity log {workflow log}
Supporting Service Usage Model(s): identity, identifier, repository storage manager
Primary Resource: collection registry
Secondary Resource(s): authentication data, access control policy/authorization data, business rule/policy data, identifier resolution data, index data, workflow log
Object(s): collection object, collection object identifier
Business Process: —

Expose Metadata Object from Repository: [BASE]

Expose a content repository for harvest, i.e., expose the metadata objects in a content repository for harvest.

This functionality is required for a content repository as a content management function (as a necessary prerequisite for *Ingest/Harvest Metadata Object Repositories*). Its equivalent *Expose Metadata Object from Registry*, applied to the metadata registry as a content delivery function, is value-added.

Basic workflow steps (without error handling) are:

- Accept a harvest request for metadata objects in the content repository (the request). The request is issued once for the entire federation.
- Authenticate and authorize (via repository access controls and business rules) the request.
- Validate the request.
- Harvest the metadata. Determine the harvestable metadata objects in the content repository (the harvest resource service genre applied to the content repository resource class).
- Return the harvest resource service genre results.

End Point: harvest {repository federation}

Supporting Service Genre(s): authenticate {agent}, authorize {access control policy/authorization data}, authorize {business rule/policy data}, activity log {workflow log}

Supporting Service Usage Model(s): identity, identifier, repository storage manager

Primary Resource: content repository

Secondary Resource(s): authentication data, access control policy/authorization data, business rule/policy data, identifier resolution data, index data, workflow log

Object(s): harvest request, metadata object dissemination, results set object

Business Process: *Pull Metadata of Item into Registry*

Ingest/Deposit Metadata Object: [BASE]

Add discoverable content to the registry, i.e., add a metadata object (or set of metadata objects) to the metadata registry. Includes the original ingestion, or adding additional metadata objects for the same source content object (e.g., metadata objects following distinct metadata schemata, different instances [*FRBR expressions*] of the metadata object in the same metadata schema).

Discoverable content **MUST** be added to the registry through at least one of the functions of *Ingest/Deposit Metadata Object* or *Ingest/Harvest Metadata Object Repositories*. Both functionalities are BASE, although any registry federation need only implement one.

Basic workflow steps (without error handling) are:

- Provide descriptive information about the content object to be added to the metadata registry, i.e., the metadata object (the request). The metadata object **SHOULD** conform to a designated standard. The request may be for a single item or a batch of items.
- Authenticate and authorize (via access controls and business rules) the request.
- Validate the overall request.
- For each object in the batch request:
 - Validate the individual request and submitted metadata object information.
 - Transform the descriptive information (schema transform plus external to internal representation).
 - Add to the descriptive information (e.g., time stamps) to complete the metadata object according to federation requirements.
 - Assign an identifier to the metadata object.
 - Add the discoverable content to the repository federation. Update the catalogue of metadata objects in the repository federation.
 - Store the metadata object.
 - Index the metadata object and catalogue.
 - If the metadata object is a new *work*, update relationship information of the metadata for the content object.
 - Log the activity.
 - Return status results.
 - Return metadata object identifier.
- Log the activity.
- Return status results.

End Point: deposit {metadata registry}

Supporting Service Genre(s): authenticate {agent}, authorize {access control policy/authorization data}, authorize {business rule/policy data}, validate {metadata object}, schema transform {metadata object}, register identifier {object}, index {metadata registry}, activity log {workflow log}

Supporting Service Usage Model(s): identity, identifier, repository storage manager

Primary Resource: metadata registry

Secondary Resource(s): authentication data, access control policy/authorization data, business rule/policy data, identifier resolution data, index data, workflow log

Object(s): metadata object

Business Process: *Push Metadata of Item into Registry*

Ingest/Harvest Metadata Object Repositories: [BASE]

Add discoverable content to the registry, i.e., add a metadata object (or set of metadata objects) to the metadata registry. Includes the original ingestion, or adding additional metadata objects for the source content object.

Basic workflow steps (without error handling) are:

- Trigger a harvest cycle (the request).
- Identify the repositories or collection in the repository federation to harvest.
- For each set to be harvested:
 - Send a harvest request to the constituent of the repository federation. The request returns a set of metadata objects. This functionality from the perspective of the constituent is detailed in *Expose Metadata Object to Registry*. The metadata objects SHOULD conform to a designated standard.
 - For each harvested metadata object:
 - Validate the harvested metadata object information.
 - Transform the descriptive information (schema transform plus external to internal representation).
 - Add to the descriptive information (e.g., time stamps) to complete the metadata object.
 - Assign an identifier to the metadata object.
 - Add the discoverable content to the repository federation. Update the catalogue of metadata objects in the repository federation.
 - Store the metadata object.
 - Index the metadata object and catalogue.
 - If the metadata object is a new *work*, update relationship information of the metadata for the content object.
 - Log the activity.
 - Log the activity.
- Log the activity.
- Return status results.

End Point: harvest {repository federation}

Supporting Service Genre(s): authenticate {agent}, authorize {access control policy/authorization data}, authorize {business rule/policy data}, validate {metadata object}, schema transform {metadata object}, register identifier {object}, index {metadata registry}, activity log {workflow log}

Supporting Service Usage Model(s): identity, identifier, repository storage manager

Primary Resource: metadata registry

Secondary Resource(s): authentication data, access control policy/authorization data, business rule/policy data, identifier resolution data, index data, workflow log

Object(s): harvest request, repository object dissemination, results set

Business Process: *Pull Metadata of Item into Registry*

Delete Metadata Object: [BASE]

Delete discoverable content from the registry, i.e., delete a metadata object (or set of metadata objects) from the metadata registry.

Basic workflow steps (without error handling) are:

- Identify the metadata object for deletion (via metadata object identifier) (the request). The request may be for a single item or a batch of items.
- Authenticate and authorize (via access controls and business rules) the request.
- Validate the overall request.

- For each object in the batch request:
 - Validate the request and submitted metadata object identifier.
 - Delete the metadata object from the repository federation. Update the catalogue of metadata objects in the repository federation.
 - Delete the metadata object.
 - De-index the metadata object and catalogue.
 - Log the activity.
- Log the activity.
- Return status results.

End Point: delete {metadata registry}

Supporting Service Genre(s): authenticate {agent}, authorize {access control policy/authorization data}, authorize {business rule/policy data}, index {metadata registry}, activity log {workflow log}

Supporting Service Usage Model(s): identity, identifier, repository storage manager

Primary Resource: metadata registry

Secondary Resource(s): authentication data, access control policy/authorization data, business rule/policy data, identifier resolution data, index data, workflow log

Object(s): metadata object identifier

Business Process: *Deactivate Metadata of Item in Registry* (extension)

Update Metadata Object: [VALUE ADDED]

Change the description of discoverable content in the registry, i.e., change the description of a metadata object or set of metadata objects in the metadata registry (content object or metadata object moves to a new location, repository, collection, etc.).

Basic workflow steps (without error handling) are:

- Identify the metadata object for update (via metadata object identifier) (the request) and provide descriptive information about the content object, i.e., updates to the metadata object. The request may be for a single item or a batch of items.
- Authenticate and authorize (via access controls and business rules) the request.
- Validate the overall request.
- For each object in the batch request:
 - Validate the individual request and submitted metadata object information.
 - Transform the descriptive information (schema transform plus external to internal representation).
 - Add to the descriptive information (e.g., time stamps) to complete the metadata object.
 - Update the discoverable content in the repository federation. Update the catalogue of metadata objects in the repository federation.
 - Update the metadata object.
 - Re-index the metadata object and catalogue.
 - If the metadata object is a new *work*, update relationship information of the metadata for the content object.
 - Log the activity.
 - Return status results.
- Log the activity.
- Return status results.

End Point: update {metadata registry}

Supporting Service Genre(s): authenticate {agent}, authorize {access control policy/authorization data}, authorize {business rule/policy data}, validate {metadata object}, schema transform {metadata object}, index {metadata registry}, activity log {workflow log}

Supporting Service Usage Model(s): identity, identifier, repository storage manager

Primary Resource: metadata registry

Secondary Resource(s): authentication data, access control policy/authorization data, business rule/policy data, identifier resolution data, index data, workflow log

Object(s): metadata object, metadata object identifier

Business Process: *Update Metadata of Item in Registry*

Change Status Metadata Object: [VALUE ADDED]

Change the status of discoverable content in the registry, i.e., change the status of a metadata object or set of metadata objects in the metadata registry (status determines participation of the metadata object or content object in responses to other requests). (The function is different than *Delete Metadata Object*: see business process *Deactivate Metadata of Item in Registry*.)

Basic workflow steps (without error handling) are:

- Identify the metadata object for status change (via metadata object identifier) (the request). The request may be for a single item or a batch of items.
- Authenticate and authorize (via access controls and business rules) the request.
- Validate the overall request.
- For each object in the batch request:
 - Validate the individual request and submitted metadata object identifier.
 - Change the status of the discoverable content in the repository federation. Change the status in the catalogue of metadata objects in the repository federation.
 - Log the activity.
 - Return status results.
- Return status results.

End Point: change status {metadata registry}

Supporting Service Genre(s): authenticate {agent}, authorize {access control policy/authorization data}, authorize {business rule/policy data}, activity log {workflow log}

Supporting Service Usage Model(s): identity, identifier, repository storage manager

Primary Resource: metadata registry

Secondary Resource(s): authentication data, access control policy/authorization data, business rule/policy data, identifier resolution data, index data, workflow log

Object(s): metadata object, metadata object identifier

Business Process: *Deactivate Metadata of Item in Registry*

Augment Metadata Object Discovery Information: [VALUE ADDED]

Add to the description of discoverable content in the registry i.e., add metadata and descriptive information to a metadata object in the metadata registry to aid in discovery (rank, rate, classify, annotate, recommend).

Basic workflow steps (without error handling) are:

- Identify the metadata object for augmentation (via metadata object identifier) (the request) and provide descriptive information about the content object, i.e., augmentation information for the metadata object. The request may be for a single item or a batch of items.
- Authenticate and authorize (via access controls and business rules) the request.
- Validate the overall request.
- For each object in the batch request:
 - Validate the individual request and submitted metadata object information.
 - Transform the descriptive information (schema transform plus external to internal representation).
 - Add to the descriptive information (e.g., time stamps) to complete the metadata object.
 - Augment the discoverable content in the repository federation. Update the catalogue of metadata objects in the repository federation.
 - Update the metadata object.
 - Re-index the metadata object and catalogue.
 - Log the activity.
 - Return status results.
- Log the activity.
- Return status results.

End Point: augment {metadata registry}

Supporting Service Genre(s): authenticate {agent}, authorize {access control policy/authorization data}, authorize {business rule/policy data}, validate {metadata object}, schema transform {metadata object}, index {metadata registry}, activity log {workflow log}

Supporting Service Usage Model(s): identity, identifier, repository storage manager

Primary Resource: metadata registry

Secondary Resource(s): authentication data, access control policy/authorization data, business rule/policy data, identifier resolution data, index data, workflow log

Object(s): metadata object, metadata object identifier

Business Process: *Classify Item; Annotate Item; Recommend Item; Rate Item*

Generate Content Object Metadata Object: [VALUE ADDED]

Generate metadata for content, i.e., from the source content object create a metadata object for discovery.

Basic workflow steps (without error handling) are:

- Identify the repositories or collection in the repository federation that contain discoverable content.
- For each set:
 - Send an *obtain* request to the constituent of the repository federation. The request returns a set of content objects.
 - For each obtained content object:
 - Generate the metadata. The metadata objects SHOULD conform to a designated standard.
 - Log the activity.
 - Return the generated metadata objects.
 - Return status results.
- Return status results.

End Point: generate {content object basic metadata object}

Supporting Service Genre(s): authenticate {agent}, authorize {access control policy/authorization data}, authorize {business rule/policy data}, obtain resource, activity log {workflow log}

Supporting Service Usage Model(s): identity, repository storage manager

Primary Resource: repository

Secondary Resource(s): authentication data, access control policy/authorization data, business rule/policy data, repository registry, collection registry, workflow log

Object(s): repositories or collections

Business Processes: *Classify Item; Annotate Item; Recommend Item; Rate Item*. (Metadata augments existing registered metadata) *Push Metadata of Item into Registry; Pull Metadata of Item into Registry*. (Metadata not already registered)

Escrow Content: [VALUE ADDED]

Make an escrow copy of content registered with the federation.

Basic workflow steps (without error handling) are:

- Provide the registered content to be escrowed, i.e., the content object (the request). The request may be for a single item or a batch of items. The content is obtained through the existing services in the federation.
- Authenticate and authorize (via access controls and business rules) the request.
- Validate the overall request.
- For each object in the batch request:
 - Validate the individual request and submitted content object.
 - Add core metadata (e.g., time stamps) to complete the information associated with the escrow copy of the content object.
 - Assign an identifier to the escrow copy of the content object.
 - Add the escrow copy of the content object to the escrow repository.
 - Log the activity.
 - Return status results.
- Return status results.

End Point: escrow deposit {escrow repository}

Supporting Service Genre(s): authenticate {agent}, authorize {access control policy/authorization data}, authorize {business rule/policy data}, register identifier {object}, activity log {workflow log}

Supporting Service Usage Model(s): identity, identifier, repository storage manager

Primary Resource: escrow repository

Secondary Resource(s): authentication data, access control policy/authorization data, business rule/policy data, identifier resolution data, workflow log

Object(s): content object

Business Process: —

Archive Content: [VALUE ADDED]

Make an archival copy of content registered with the federation.

Basic workflow steps (without error handling) are:

- Provide the content to be archived, i.e., the content object (the request). The request may be for a single item or a batch of items.
- Authenticate and authorize (via access controls and business rules) the request.
- Validate the overall request.
- For each object in the batch request:
 - Validate the individual request and submitted content object.
 - Add core metadata (e.g., time stamps) to complete the information associated with the archive copy of the content object.
 - Assign an identifier to the content object.
 - Add the archive copy of the content object to the archive repository.
 - Log the activity.
 - Return status results.
- Return status results.

End Point: archive deposit {archive repository}

Supporting Service Genre(s): authenticate {agent}, authorize {access control policy/authorization data}, authorize {business rule/policy data}, register identifier {object}, activity log {workflow log}

Supporting Service Usage Model(s): identity, identifier, repository storage manager

Primary Resource: archive repository

Secondary Resource(s): authentication data, access control policy/authorization data, business rule/policy data, identifier resolution data, workflow log

Object(s): content object

Business Process: —

Move Content to Repository: [VALUE ADDED]

Move a content object from one repository within the federation to another repository within the federation. The move requires that the metadata object and content object identifier location resolution data be changed.

Basic workflow steps (without error handling) are:

- Provide the content to be moved, source and destination, i.e., the content object, source repository and destination repository (the request).
- Validate the request.
- Update the content object identifier indicating the new repository location.
- Update the metadata object indicating the new repository location.
- Log the activity.
- Return status results.

End Point: move content object {repository registry}

Supporting Service Genre(s): authenticate {agent}, authorize {access control policy/authorization data}, authorize {business rule/policy data}, update identifier resolution data {object}, index {metadata registry}, activity log {workflow log}

Supporting Service Usage Model(s): identity, identifier, repository storage manager

Primary Resource: metadata registry

Secondary Resource(s): authentication data, access control policy/authorization data, business rule/policy data, identifier resolution data, index data, workflow log

Object(s): content object identifier

Business Process: *Update Metadata of Item in Registry*

Move Content to Collection: [VALUE ADDED]

Move a content object from one collection within the federation to another collection within the federation. The move requires that the metadata object and content object identifier location resolution data be changed.

Basic workflow steps (without error handling) are:

- Provide the content to be moved, source and destination, i.e., the content object, source collection and destination collection (the request).
- Validate the request.
- Update the content object identifier indicating the new collection location if necessary (per policy).
- Update the metadata object indicating the new collection location if necessary (per policy).

- Log the activity.
- Return status results.

End Point: move content object {collection registry}

Supporting Service Genre(s): authenticate (agent), authorize {access control policy/authorization data}, authorize {business rule/policy data}, update identifier resolution data {object}, index {metadata registry}, activity log {workflow log}

Supporting Service Usage Model(s): identity, identifier, repository storage manager

Primary Resource: metadata registry

Secondary Resource(s): authentication data, access control policy/authorization data, business rule/policy data, identifier resolution data, index data, workflow log

Object(s): content object identifier

Business Process: *Update Metadata of Item in Registry*

Content Discovery Functions

Query Content Object Registry: [BASE]

Query the metadata registry to discover content objects (via their metadata objects).

Basic workflow steps (without error handling) are:

- Provide the query (the request). The request will specify the dissemination and format. The request SHOULD be expressed in a standard query language.
- Authenticate and authorize (via access controls and business rules) the request.
- Validate the request.
- Transform/Enhance the query (Transform: spell check, language translation, etc.; Enhance: synonyms, etc.).
- Query/search the metadata registry, returning a set of metadata object identifiers and corresponding collection object identifiers.
- Log the query and results.
- Build a results set. For each item in the query response:
 - Filter the metadata object (to comply with metadata object access controls and obligations).
 - Retrieve a dissemination of the metadata object from the metadata registry.
 - Transform the metadata object to the requested results set format.
 - Add the item to the results set.
- Transform the complete results set (rendering transform, schema transform).
- Log the activity.
- Return the results set.
- Return status results.

End Point: search {metadata registry}

Supporting Service Genre(s): authenticate (agent), authorize {access control policy/authorization data}, authorize {business rule/policy data}, synonym transform (query object), spell check transform (query object), language translate transform (query object), filter {metadata registry}, schema transform {metadata object}, render transform {metadata object}, query/results log (query log), activity log {workflow log}

Supporting Service Usage Model(s): identity, identifier, repository storage manager

Primary Resource: metadata registry

Secondary Resource(s): authentication data, access control policy/authorization data, business rule/policy data, identifier resolution data, index data, workflow log, query log

Object(s): query object, results set object

Business Process: *Discover Item through Search on Registry*

Query Repository Object Registry: [VALUE ADDED]

Query the repository registry to discover repository objects.

Basic workflow steps (without error handling) are:

- Provide the query (the request). The request will specify the dissemination and format. The request SHOULD be expressed in a standard query language.
- Authenticate and authorize (via access controls and business rules) the request.
- Validate the request.

- Transform/Enhance the query (Transform: spell check, language translation, etc.; Enhance: synonyms, etc.).
- Query/search the repository registry, returning a set of repository object identifiers.
- Log the query and results.
- Build a results set. For each item in the query response:
 - Filter the object (to comply with content access controls and obligations).
 - Retrieve a dissemination of the repository object from the repository registry.
 - Transform the repository object to the requested results set format.
 - Add the item to the results set.
- Transform the complete results set (rendering transform, schema transform).
- Log the activity.
- Return the results set.
- Return status results.

End Point: search {repository registry}

Supporting Service Genre(s): authenticate (agent), authorize {access control policy/authorization data}, authorize {business rule/policy data}, synonym transform (query object), spell check transform (query object), language translate transform (query object), filter {repository registry}, schema transform {repository object}, render transform {repository object}, query/results log (query log), activity log {workflow log}

Supporting Service Usage Model(s): identity, identifier, repository storage manager

Primary Resource: repository registry

Secondary Resource(s): authentication data, access control policy/authorization data, business rule/policy data, identifier resolution data, index data, workflow log, query log

Object(s): query object, results set object

Business Process: *Obtain Item from Repository, Obtain Metadata of Item from Repository*

Query Collection Object Registry: [VALUE ADDED]

Query the collection registry to discover collection objects.

Basic workflow steps (without error handling) are:

- Provide the query (the request). The request will specify the dissemination and format. The request SHOULD be expressed in a standard query language.
- Authenticate and authorize (to comply with access controls and business rules) the request.
- Validate the request.
- Transform/Enhance the query (Transform: spell check, language translation, etc.; Enhance: synonyms, etc.).
- Query/search the collection registry, returning a set of collection objects identifiers.
- Log the query and results.
- Build a results set. For each item in the query response:
 - Filter the object (to comply with content access controls and obligations).
 - Retrieve a dissemination of the collection object from the collection registry.
 - Transform the collection object to the requested results set format.
 - Add the item to the results set.
- Transform the complete results set (rendering transform, schema transform).
- Log the activity.
- Return the results set.
- Return status results.

End Point: search {collection registry}

Supporting Service Genre(s): authenticate (agent), authorize {access control policy/authorization data}, authorize {business rule/policy data}, synonym transform (query object), spell check transform (query object), language translate transform (query object), filter {collection registry}, schema transform {collection object}, render transform {collection object}, query/results log (query log), activity log {workflow log}

Supporting Service Usage Model(s): identity, identifier, repository storage manager

Primary Resource: collection registry

Secondary Resource(s): authentication data, access control policy/authorization data, business rule/policy data, identifier resolution data, index data, workflow log, query log

Object(s): query object, results set object

Business Process: *Obtain Item from Repository, Obtain Metadata of Item from Repository*

Browse Content Object Registry: [VALUE ADDED]

Browse the metadata registry to discover content objects (via their metadata objects).

Basic workflow steps (without error handling) are:

- Authenticate and authorize (via access controls and business rules) the request.
- Determine the initial browsing dissemination of the metadata objects.
- Log the session start.
- Begin session. Loop:
 - Obtain the current browsing dissemination of the metadata objects.
 - Build the browsing dissemination. For each object in the dissemination request
 - Retrieve a dissemination of the metadata object from the metadata registry.
 - Filter the object (via content access controls).
 - Transform the metadata object to the requested browsing set format.
 - Add the item to the results set.
 - Transform the complete browsing results set (rendering transform, schema transform).
 - Log the activity.
 - Return the dissemination for display.
 - Wait for user request to change the requested browsing dissemination.
- End session.
- Log the session end.

End Point: browse {metadata registry}

Supporting Service Genre(s): authenticate {agent}, authorize {access control policy/authorization data}, authorize {business rule/policy data}, filter {metadata registry}, schema transform {metadata object}, render transform {metadata object}, activity log {workflow log}

Supporting Service Usage Model(s): identity, identifier, repository storage manager, browse

Primary Resource: metadata registry

Secondary Resource(s): authentication data, access control policy/authorization data, business rule/policy data, identifier resolution data, index data, workflow log

Object(s): metadata object identifier, metadata object dissemination, results set object

Business Process: *Discover Item through Browse on Registry*

Browse Repository Object Registry: [VALUE ADDED]

Browse the repository registry to discover repository objects.

Basic workflow steps (without error handling) are:

- Authenticate and authorize (via access controls and business rules) the request.
- Determine the initial browsing dissemination of the repository objects.
- Log the session start.
- Begin session. Loop:
 - Obtain the current browsing dissemination of the repository objects.
 - Build the browsing dissemination. For each object in the dissemination request
 - Retrieve a dissemination of the repository object from the repository registry.
 - Filter the object (via content access controls).
 - Transform the repository object to the requested browsing set format.
 - Add the item to the results set.
 - Transform the complete browsing results set (rendering transform, schema transform).
 - Log the activity.
 - Return the dissemination for display.
 - Wait for user request to change the requested browsing dissemination.
- End session.
- Log the session end.

End Point: browse {repository registry}

Supporting Service Genre(s): authenticate {agent}, authorize {access control policy/authorization data}, authorize {business rule/policy data}, filter {repository registry}, schema transform {repository object}, render transform {repository object}, activity log {workflow log}

Supporting Service Usage Model(s): identity, identifier, repository storage manager, browse

Primary Resource: repository registry

Secondary Resource(s): authentication data, access control policy/authorization data, business rule/policy data, identifier resolution data, index data, workflow log

Object(s): repository object identifier, repository object dissemination, results set object

Business Process: —

Browse Collection Object Registry: [VALUE ADDED]

Browse the collection registry in the federation to discover collection objects.

Basic workflow steps (without error handling) are:

- Authenticate and authorize (via access controls and business rules) the request.
- Determine the initial browsing dissemination of the collection objects.
- Log the session start.
- Begin session. Loop:
 - Obtain the current browsing dissemination of the collection objects.
 - Build the browsing dissemination. For each object in the dissemination request
 - Retrieve a dissemination of the collection object from the collection registry.
 - Filter the object (via content access controls).
 - Transform the collection object to the requested browsing set format.
 - Add the item to the results set.
 - Transform the complete browsing results set (rendering transform, schema transform).
 - Log the activity.
 - Return the dissemination for display.
 - Wait for user request to change the requested browsing dissemination.
- End session.
- Log the session end.

End Point: browse {collection registry}

Supporting Service Genre(s): authenticate {agent}, authorize {access control policy/authorization data}, authorize {business rule/policy data}, filter {collection registry}, schema transform {collection object}, render transform {collection object}, activity log {workflow log}

Supporting Service Usage Model(s): identity, identifier, repository storage manager, browse

Primary Resource: collection registry

Secondary Resource(s): authentication data, access control policy/authorization data, business rule/policy data, identifier resolution data, index data, workflow log

Object(s): collection object identifier, collection object dissemination, results set object

Business Process: —

Syndicate Content Object Registry: [VALUE ADDED]

Provide syndication and notification facilities to discover content objects in the metadata registry.

Basic workflow steps (without error handling) are:

- Provide syndication request for content objects (the request).
- Authenticate and authorize (via access controls and business rules) the request.
- Validate the overall request.
- Determine the set of content objects in the metadata registry that should be part of the results set (new objects based on time stamps).
- Build a results set. For each object in the syndication set:
 - Retrieve a syndication dissemination of the metadata object from the metadata registry.
 - Filter the object (via content access controls).
 - Transform the metadata object to the syndication results set format.
 - Add the item to the results set.
- Transform the complete results set (rendering transform, schema transform).
- Log the activity.
- Return the results set.
- Return status results.

End Point: syndicate {metadata registry}

Supporting Service Genre(s): authenticate {agent}, authorize {access control policy/authorization data}, authorize {business rule/policy data}, filter {metadata registry}, schema transform {metadata object}, render transform {metadata object}, activity log {workflow log}

Supporting Service Usage Model(s): identity, identifier, repository storage manager

Primary Resource: metadata registry

Secondary Resource(s): authentication data, access control policy/authorization data, business rule/policy data, identifier resolution data, index data, workflow log

Object(s): syndicate request, metadata object dissemination, results set object

Business Process: *Syndicate Item from Registry*

Syndicate Repository Object Registry: [VALUE ADDED]

Provide syndication and notification facilities to discover repository objects in the repository registry.

Basic workflow steps (without error handling) are:

- Provide a syndication request for repository objects (the request).
- Authenticate and authorize (via access controls and business rules) the request.
- Validate the overall request.
- Determine the set of repository objects in the repository registry that should be part of the results set (new objects based on time stamps).
- Build a results set. For each object in the syndication set:
 - Retrieve a syndication dissemination of the repository object from the repository registry.
 - Filter the object (via content access controls).
 - Transform the repository object to the syndication results set format.
 - Add the item to the results set.
- Transform the complete results set (rendering transform, schema transform).
- Log the activity.
- Return the results set.
- Return status results.

End Point: syndicate {repository registry}

Supporting Service Genre(s): authenticate {agent}, authorize {access control policy/authorization data}, authorize {business rule/policy data}, filter {repository registry}, schema transform {repository object}, render transform {repository object}, activity log {workflow log}

Supporting Service Usage Model(s): identity, identifier, repository storage manager

Primary Resource: repository registry

Secondary Resource(s): authentication data, access control policy/authorization data, business rule/policy data, identifier resolution data, index data, workflow log

Object(s): syndicate request, repository object dissemination, results set object

Business Process: —

Syndicate Collection Object Registry: [VALUE ADDED]

Provide syndication and notification facilities to discover collection objects in the collection registry.

Basic workflow steps (without error handling) are:

- Provide a syndication request for collection objects (the request).
- Authenticate and authorize (via access controls and business rules) the request.
- Validate the overall request.
- Determine the set of collection objects in the collection registry that should be part of the results set (new objects based on time stamps).
- Build a results set. For each object in the syndication set:
 - Retrieve a syndication dissemination of the collection object from the collection registry.
 - Filter the object (via content access controls).
 - Transform the collection object to the syndication results set format.
 - Add the item to the results set.
- Transform the complete results set (rendering transform, schema transform).
- Log the activity.
- Return the results set.
- Return status results.

End Point: syndicate {collection registry}
Supporting Service Genre(s): authenticate {agent}, authorize {access control policy/authorization data}, authorize {business rule/policy data}, filter {collection registry}, schema transform {collection object}, render transform {collection object}, activity log {workflow log}
Supporting Service Usage Model(s): identity, identifier, repository storage manager, browse
Primary Resource: collection registry
Secondary Resource(s): authentication data, access control policy/authorization data, business rule/policy data, identifier resolution data, index data, workflow log
Object(s): syndicate request, collection object dissemination, results set object
Business Process: —

Content Delivery Functions

Obtain Metadata Object: [BASE]

Obtain the metadata for a content object or set of content objects, i.e., obtain a set of metadata objects. The details of the request will specify if the metadata object is to come from the metadata registry or source repository and the metadata dissemination to be returned.

Basic workflow steps (without error handling) are:

- Identify the metadata object for retrieval (via metadata object identifier) (the request). The request may be for a single item or a batch of items.
- Authenticate and authorize (via access controls and business rules) the request.
- Validate the overall request.
- Build a results set. For each object in the batch request:
 - Validate the individual request and submitted metadata object identifier.
 - If the request is for the metadata object in the metadata registry, retrieve the metadata object from the metadata registry.
 - Retrieve a dissemination of the metadata object from the metadata registry.
 - Filter the object (via content access controls).
 - Transform the metadata object to the requested results set format.
 - Add the item to the results set.
 - If the request is for the metadata object in the source repository, retrieve the metadata object from the source repository.
 - Resolve the identifier.
 - Send the request to the obtain service end point of the identified source repository.
 - Retrieve a dissemination of the metadata object from the source repository.
 - Filter the object (via content access controls).
 - Transform the metadata object to the requested results set format.
 - Add the item to the results set.
- Transform the complete results set (rendering transform, schema transform).
- Log the activity.
- Return the results set.
- Return status results.

End Point: obtain {metadata registry}; obtain {content repository}
Supporting Service Genre(s): authenticate {agent}, authorize {access control policy/authorization data}, authorize {business rule/policy data}, resolve identifier {object}, filter {metadata registry}, schema transform {metadata object}, render transform {metadata object}, activity log {workflow log}
Supporting Service Usage Model(s): identity, identifier, repository storage manager, deliver
Primary Resource: metadata registry
Secondary Resource(s): authentication data, access control policy/authorization data, business rule/policy data, identifier resolution data, workflow log
Object(s): metadata object identifier, metadata object dissemination, results set object
Business Process: *Discover Item through Search on Registry, Discover Item through Browse on Registry*

Obtain Content Object: [BASE]

Obtain a content object or set of content objects. The details of the request will specify the source (source repository, archive, escrow) and dissemination format.

Basic workflow steps (without error handling) are:

- Identify the content object for retrieval (via content object identifier) (the request). The request may be for a single item or a batch of items.
- Authenticate and authorize (via access controls and business rules) the request.
- Validate the overall request.
- Build a results set. For each object in the batch request:
 - Validate the individual request and submitted metadata object identifier.
 - If the request is for a content object in the archive repository or escrow repository:
 - Retrieve a dissemination of the content object from the repository.
 - Filter the object (via content access controls).
 - Transform the content object to the requested packaging format.
 - Add the item to the results set.
 - If the request is for a content object in a source repository, retrieve the content object from the source repository.
 - Resolve the identifier.
 - Send the request to the obtain service end point of the identified source repository.
 - Retrieve a dissemination of the content object from the source repository.
 - Filter the object (via content access controls).
 - Transform the content object to the requested packaging format.
 - Add the item to the results set.
- Transform the complete results set (rendering transform, schema transform).
- Log the activity.
- Return the results set.
- Return status results.

End Point: obtain {content repository}

Supporting Service Genre(s): authenticate {agent}, authorize {access control policy/authorization data}, authorize {business rule/policy data}, resolve identifier {object}, filter {repository}, package {content object}, schema transform {content object}, render transform {content object}, activity log {workflow log}

Supporting Service Usage Model(s): identity, identifier, repository storage manager, deliver

Primary Resource: content repository, escrow repository, archive repository

Secondary Resource(s): authentication data, access control policy/authorization data, business rule/policy data, identifier resolution data, workflow log

Object(s): content object identifier, content object dissemination, results set object

Business Process: *Obtain Item from Repository*

Obtain Repository Object: [BASE]

Obtain the metadata about a repository or a set of repositories, i.e., obtain a set of repository objects from the repository registry. The details of the request will specify the metadata dissemination to be returned.

Basic workflow steps (without error handling) are:

- Identify the repository object for retrieval (via repository object identifier) (the request). The request may be for a single item or a batch of items.
- Authenticate and authorize (via access controls and business rules) the request.
- Validate the overall request.
- Build a results set. For each object in the batch request:
 - Validate the individual request and submitted repository object identifier.
 - Retrieve a dissemination of the repository object from the repository registry.
 - Filter the object (via content access controls).
 - Transform the repository object to the requested results set format.
 - Add the item to the results set.
- Transform the complete results set (rendering transform, schema transform).
- Log the activity.
- Return the results set.
- Return status results.

End Point: obtain {repository registry}

Supporting Service Genre(s): authenticate {agent}, authorize {access control policy/authorization data}, authorize {business rule/policy data}, resolve identifier {object}, filter {repository registry}, schema transform {repository object}, render transform {repository object}, activity log {workflow log}

Supporting Service Usage Model(s): identity, identifier, repository storage manager, deliver

Primary Resource: repository registry

Secondary Resource(s): authentication data, access control policy/authorization data, business rule/policy data, identifier resolution data, workflow log

Object(s): repository object identifier, repository object dissemination, results set object

Business Process: *Obtain Item from Repository, Obtain Metadata of Item from Repository*

Obtain Collection Object: [BASE]

Obtain the metadata about a collection or a set of collections, i.e., obtain a set of collection objects from the collection registry. The details of the request will specify the metadata dissemination to be returned.

Basic workflow steps (without error handling) are:

- Identify the collection object for retrieval (via collection object identifier) (the request). The request may be for a single item or a batch of items.
- Authenticate and authorize (via access controls and business rules) the request.
- Validate the overall request.
- Build a results set. For each object in the batch request:
 - Validate the individual request and submitted collection object identifier.
 - Retrieve a dissemination of the collection object from the collection registry.
 - Filter the object (via content access controls).
 - Transform the collection object to the requested results set format.
 - Add the item to the results set.
- Transform the complete results set (rendering transform, schema transform).
- Log the activity.
- Return the results set.
- Return status results.

End Point: obtain {collection registry}

Supporting Service Genre(s): authenticate {agent}, authorize {access control policy/authorization data}, authorize {business rule/policy data}, resolve identifier {object}, filter {collection registry}, schema transform {collection object}, render transform {collection object}, activity log {workflow log}

Supporting Service Usage Model(s): identity, identifier, repository storage manager, browse, deliver

Primary Resource: collection registry

Secondary Resource(s): authentication data, access control policy/authorization data, business rule/policy data, identifier resolution data, workflow log

Object(s): collection object identifier, collection object dissemination, results set object

Business Process: *Obtain Item from Repository, Obtain Metadata of Item from Repository*

Expose Metadata Object from Registry: [VALUE ADDED]

Expose the registry for harvest, i.e., expose the metadata objects in the metadata registry for harvest.

Basic workflow steps (without error handling) are:

- Accept a harvest request for metadata objects in the metadata registry (the request).
- Authenticate and authorize (via access controls and business rules) the request.
- Validate the request.
- Harvest the metadata. Determine the harvestable metadata objects in the metadata registry (the harvest resource service genre applied to the metadata registry resource).
- Log the activity.
- Return the harvest resource service genre results.

End Point: harvest {metadata registry}

Supporting Service Genre(s): authenticate {agent}, authorize {access control policy/authorization data}, authorize {business rule/policy data}, activity log {workflow log}

Supporting Service Usage Model(s): identity, identifier, repository storage manager

Primary Resource: metadata registry

Secondary Resource(s): authentication data, access control policy/authorization data, business rule/policy data, identifier resolution data, index data, workflow log

Object(s): harvest request, metadata object dissemination, results set object

Business Process: *Expose Registry for Harvesting*

Expose Repository Object: [VALUE ADDED]

Expose the repository objects in the repository registry for harvest.

Basic workflow steps (without error handling) are:

- Provide a harvest request for repository objects in the repository registry (the request).
- Authenticate and authorize (via access controls and business rules) the request.
- Validate the request.
- Harvest the metadata. Determine the harvestable repository objects in the repository registry (the harvest resource service genre applied to the repository registry resource).
- Log the activity.
- Return the harvest resource service genre results.

End Point: harvest {repository registry}

Supporting Service Genre(s): authenticate {agent}, authorize {access control policy/authorization data}, authorize {business rule/policy data}, activity log {workflow log}

Supporting Service Usage Model(s): identity, identifier, repository storage manager

Primary Resource: repository registry

Secondary Resource(s): authentication data, access control policy/authorization data, business rule/policy data, identifier resolution data, index data, workflow log

Object(s): harvest request, repository object dissemination, results set object

Business Process: *Expose Registry for Harvesting*

Expose Collection Object Metadata: [VALUE ADDED]

Expose the collection objects in the collection registry for harvest.

Basic workflow steps (without error handling) are:

- Provide a harvest request for collection objects in the collection registry (the request).
- Authenticate and authorize (via access controls and business rules) the request.
- Validate the request.
- Harvest the metadata. Determine the harvestable collection objects in the collection registry (the harvest resource service genre applied to the collection registry resource).
- Log the activity.
- Return the harvest resource service genre results.

End Point: harvest {collection registry}

Supporting Service Genre(s): authenticate {agent}, authorize {access control policy/authorization data}, authorize {business rule/policy data}, activity log {workflow log}

Supporting Service Usage Model(s): identity, identifier, repository storage manager, browse

Primary Resource: collection registry

Secondary Resource(s): authentication data, access control policy/authorization data, business rule/policy data, identifier resolution data, index data, workflow log

Object(s): harvest request, collection object dissemination, results set object

Business Process: *Expose Registry for Harvesting*

Other Functions

No other functionality is defined. The functionality that is defined MAY be extended.

Structure & Organization

The Structure & Organization element is a normative, technical description of the Service Usage Model that documents and illustrates the service-oriented operational view of the Service Usage Model, in terms of services, resources and their messaging interactions. The Structure & Organization element is required to implement applications that use the Service Usage Model, but the Service Usage Model must be understandable without reference to the contents of the element.
NB: The Structure & Organization section does not describe how to implement applications that use the Service Usage Model, but describes how to implement elements of the Service Usage Model itself.

The structure of the repository federation service usage model is illustrated on Figure 1. The core of the repository federation is the central registry which includes core resources to manage a metadata registry, collection registry,

and repository registry. In addition, the central registry manages supporting resources, including system data, policy and rule data, and logs. The central registry also uses resources that are managed by other service usage models, including access control, user and identity data that are key resources of an identity service usage model and identifier resolution data that is part of an identifier service usage model. A revised view of the repository federation service usage model, combined with Registry System services to provide complete repository federation functionality, is illustrated in Figure 5.

The repository federation service usage model exposes a collection of service end points, modelled as service genres. The core functionality of the service usage model is mapped to these service genres. Typically one function is mapped to one service end point, supported by a single service genre. The service genres function in the same fashion, accepting a request from the user and then performing the requested operation. When valid, the operation either modifies the resources of the central registry or disseminates information about the objects in the central registry. The workflows for all functions follow this similar pattern.

Other than working on a shared resource set, the service genres are not interdependent. All service end points do rely on a common set of service genres, including those for identity management and access control.

Most of the service genres function within a trust environment. External requests are validated, but inside the trust environment, service requests are accepted as is from trusted services.

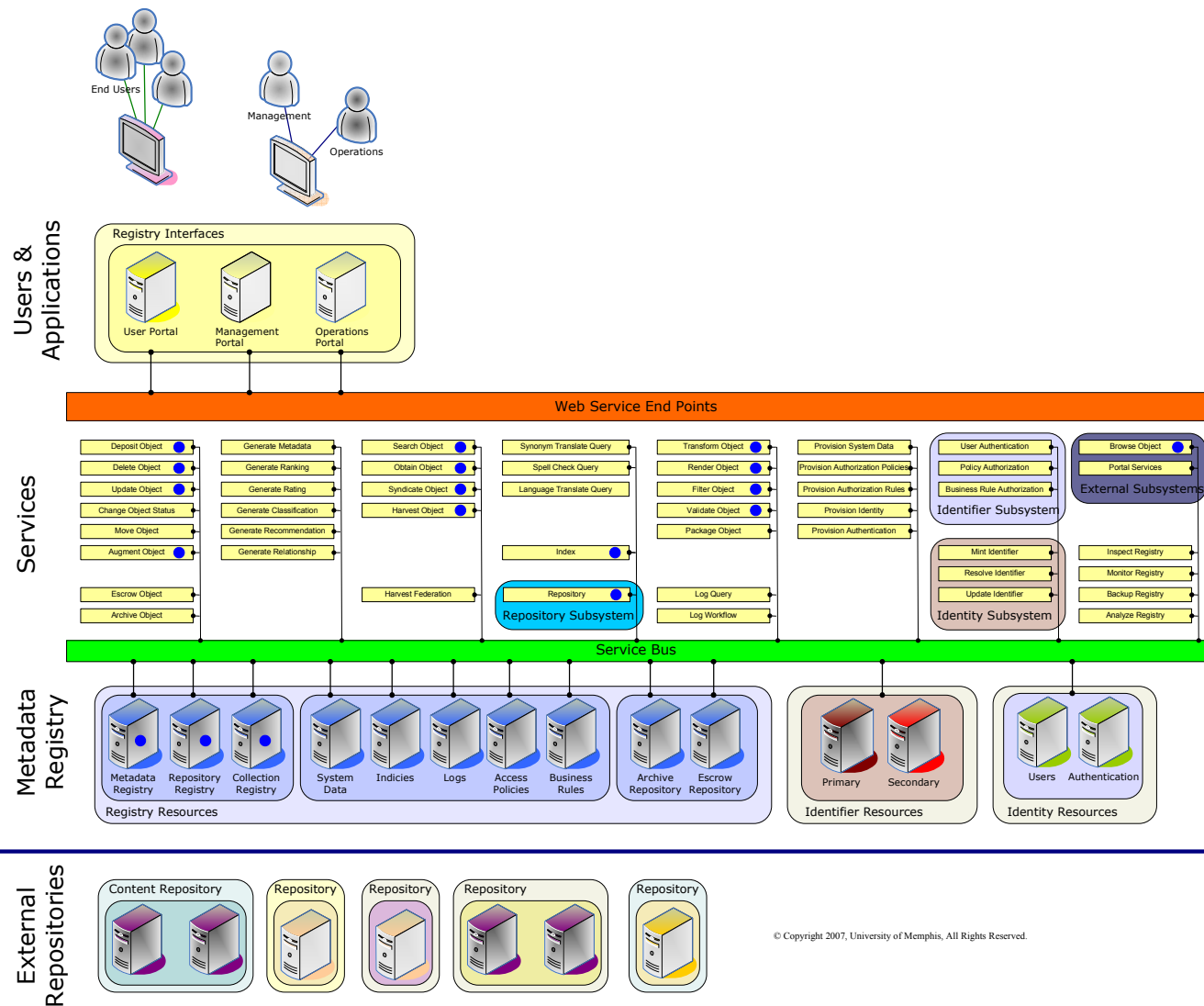
Figure 6 illustrates an ESB-oriented view of a typical workflow in the repository federation service usage model. The shared resources are shown at the bottom on the diagram. The identifier and identity resources are assumed to be part of an external environment. Typical workflows of data in the repository federation are illustrated with UML Activity diagrams in Figures 7 and 8.

There is no defined overall application. The repository federation service usage model is assumed to be used within a set of applications that are used to provide end user access to the repository federation, provide management and provide operational support. Operations may require transactional- and session-based controls.

The repository federation is based on the metadata (metadata objects, collection objects, repository objects) from the constituent repositories that participate in the repository federation. These constituent repositories and any service end points that they expose are not part of the overall repository federation service usage model. However, the service end points from these repositories may be accessed to provide some of the functionality of the repository federation service usage model, e.g., harvesting or obtaining content from a repository.

The shared resource set within the repository federation service usage model is composed of domain resources, including a metadata registry, a collection registry, and a repository registry. In addition there is a collection of other resources in the service usage model as illustrated in Figure 5.

The following is an illustration of data movement within a repository federation with the structure outlined:



© Copyright 2007, University of Memphis, All Rights Reserved.

Figure 5: Repository Federation and Registry System Design

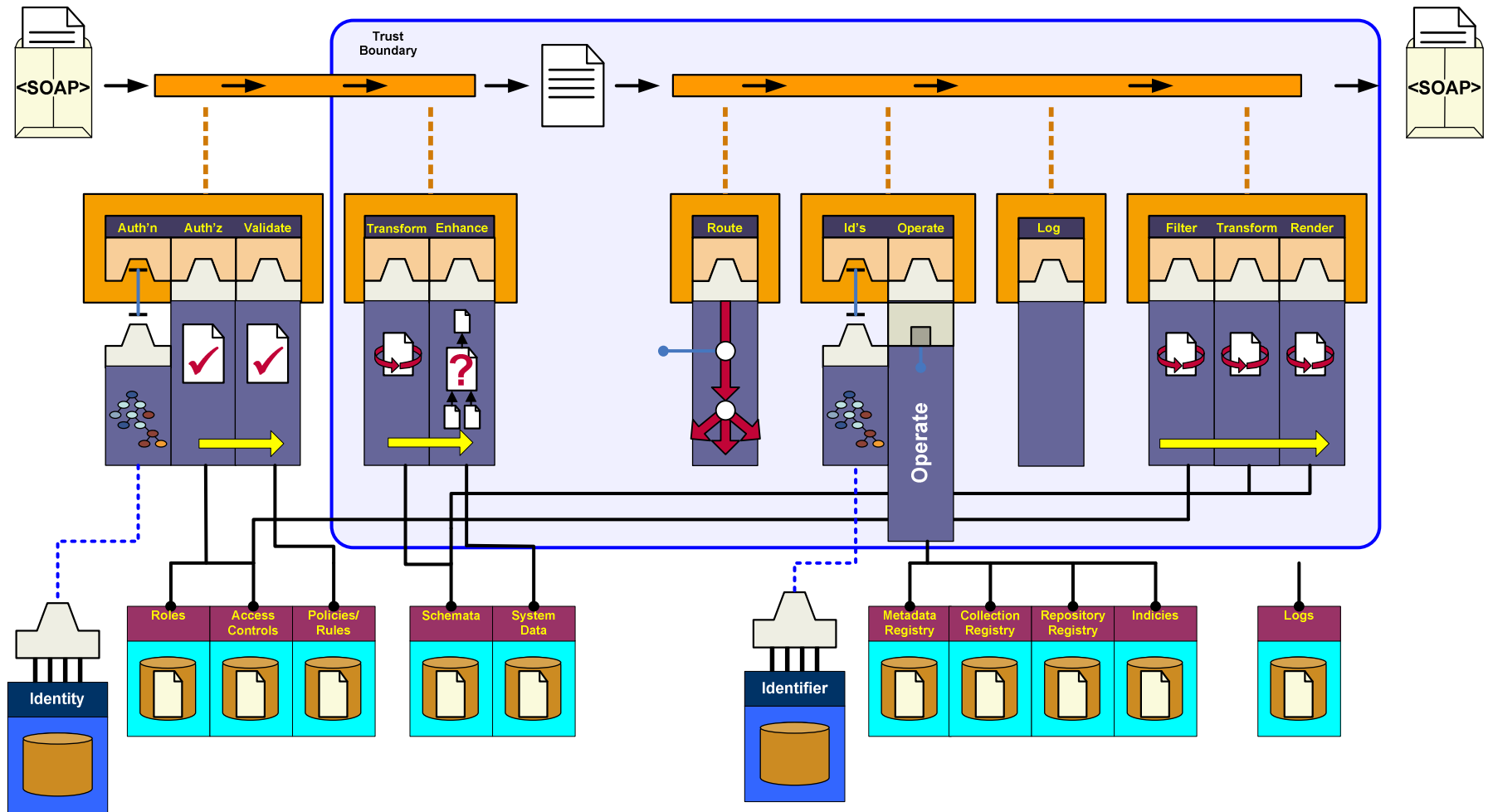


Figure 6: Repository Federation Service Usage Model workflow (ESB Notation)

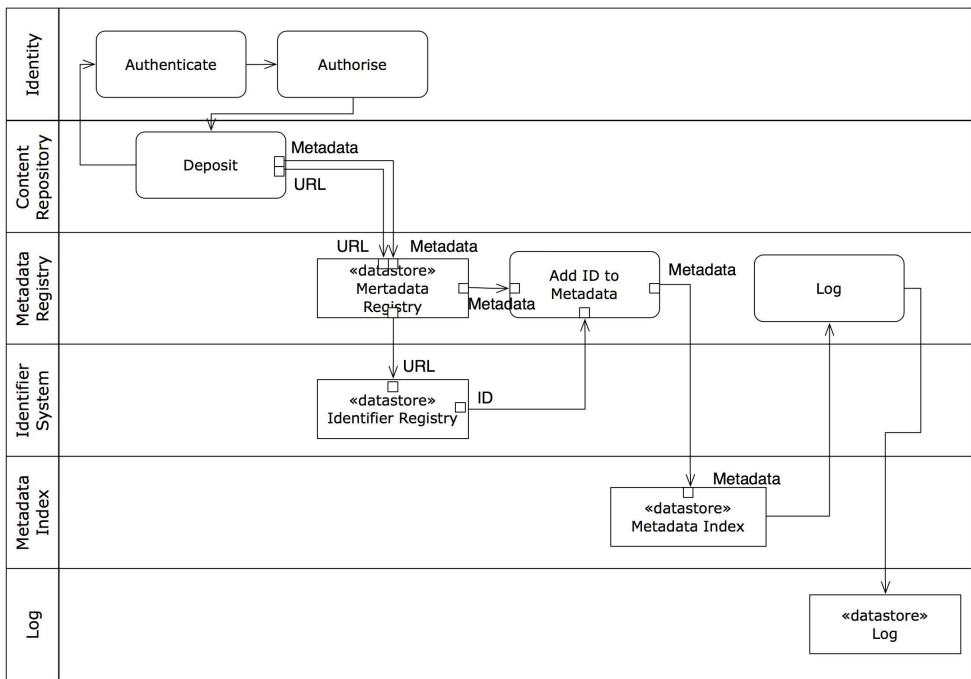


Figure 7: Workflow for *Ingest/Deposit Metadata Object = Add Metadata Description to Registry* (UML Activity Diagram)

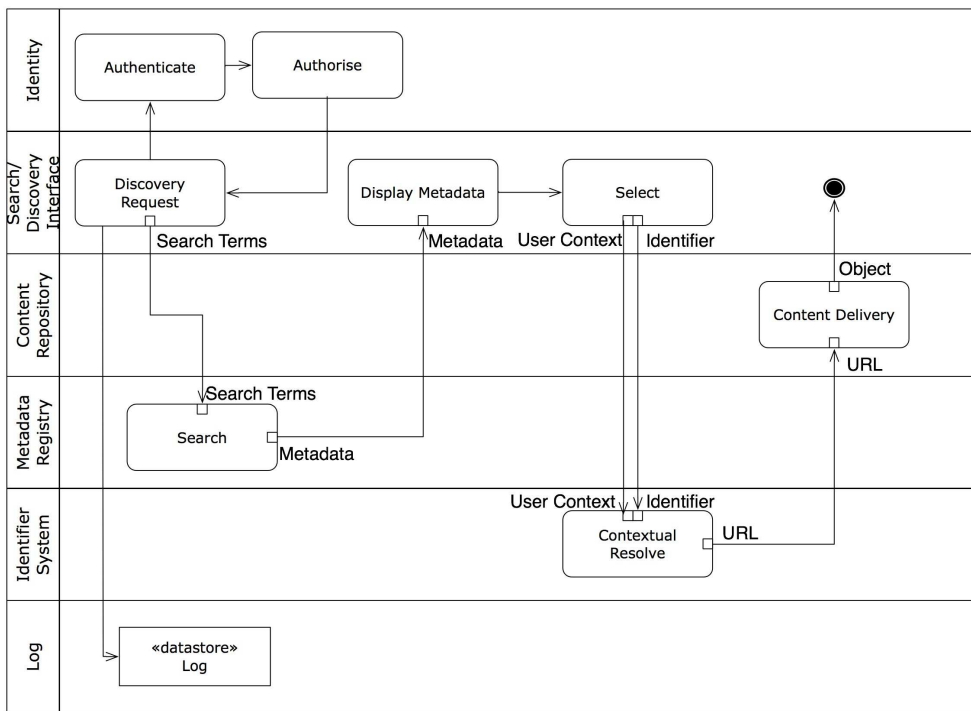


Figure 8: Workflow for *Query Content Object Registry and Obtain Content Object = Discovery-to-Delivery* (UML Activity Diagram)

Design Decisions & Tradeoffs

The Design Decisions & Tradeoffs element documents overall choices, tradeoffs and their implications on the design of the Service Usage Model itself. It does not address the design of applications or the internal implementation of the elements of the Service Usage Model. No critical or essential information required to understand the Service Usage Model should be included.

Capabilities and features of the repository federation service usage model are based on a core set of features and requirements (refer to <http://www.cordra.net>). Specific limitations are defined in the Applicability Section of the Service Usage Model Definition.

Only essential functions needed to provide basic capabilities are considered core features of the repository federation service usage model. Features and functionality that could be layered on core features or built as composite services are labelled as value added. This distinction need not be carried directly into an implementation.

The service genres used in this service usage model provide service end points used to access the functionality of the repository federation. Providing a service end point was the criterion for defining a service genre. Many of the service genres described map directly to one of the exposed functionalities of the repository federation. These service genres could also be expressed as small service usage models that capture all of the workflow and logic presented in the functionality and which are built on smaller specific service genres. The decision to use service genres instead of small service usage models permits building small reusable components rather than components that encode more workflow and processing.

Most of the service genres are defined to provide a single service end point. More complex service genres that provide a collection of related behaviours could be defined as an alternative approach. The decision to use multiple service genres again permits building small components rather than components that combine potentially different behaviours. In some cases, closely related functionality has been combined in a single service genre to eliminate creating a proliferation of service genres that are almost identical.

Functions applied to content are designed as batch activities, assuming there are multiple objects to be processed. Other functions are designed to deal with only a single object. Batch functions are assumed when the user may need to process multiple objects. Most of the batch functions can be simplified to single object functions or vice versa if required.

The repository federation service usage model assumes services are part of a trust environment and that full authentication controls are not required to use services within the trust environment. Authorization processes are still required within the trust environment, and to gain initial access to the trust environment. Providing the trust environment of components simplifies the management of authentication and authorization by individual services and for individual users but opens the potential for components to be improperly used outside of the trust environment. Such use will compromise the integrity of a federated metadata registry.

The repository federation service usage model has not been specialized to the needs of a particular community. An individual community of practice will need to specialise the service usage model with their specific data models, policies and rules.

Many of the functions are applied to similar resources. A single service genre is assumed, e.g., *expose resource*. The service genres presented are defined in terms of the resource type, e.g., *expose metadata object* and *expose collection object*.

Specific Service Genre Modelling Decisions

Management of digital rights and digital rights service genres are not included. Digital rights is assumed to be part of a separate service usage model that is combined with the repository federation service usage model. The overall scope of such a service usage model is not presented herein.

An *Expose Content Object* function, comparable to *Expose Metadata Object to Registry*, is possible additional functionality in the broad scope of the repository domain, which allows centralized delivery of content objects from a content object registry. However this functionality is not included in this service usage model.

Implementation Guide & Dependencies

The Implementation Guide & Dependencies element describes issues of concern in implementing applications that use the Service Usage Model, including organization, performance, behaviours, representations, policies. Resolution of issues discussed is deferred to the actual application design. No critical or essential information required to understand the Service Usage Model should be included.

The repository federation service usage model does not specify the design of a particular application or system around a particular implementation. These guidelines apply to a general federated metadata registry application that provides the functionality described.

Value-added capabilities could be built by service composition. A federated metadata registry application may directly implement more than the core features or may directly implement value-added capabilities. Direct implementation may result in improved performance.

The repository federation service usage model assumes a federated metadata registry application will integrate with an existing identity management system.

The repository federation service usage model assumes a federated metadata registry application will integrate with an existing identifier system. Management of identifiers and assignment of identifiers to objects are considered part of the details of the service genres within the identifier service usage model. Explicit functionality describing identifier management and assignment is not described herein.

Several services operate on sets or batches of data. Details of batch process (including flow control) and transaction processing are deferred to the federated metadata registry applications and the service expressions that specialize the service genre.

The repository federation service usage model does not specify details of the infrastructure or components required to implement a federated metadata registry application. A federated metadata registry instance may require distributed or replicated servers and multiple service instances and resource sets for load balancing and performance.

The repository federation service usage model assumes content within the federated metadata registry application is stored and managed by enterprise-level data management components.

The repository federation service usage model assumes content within the federated metadata registry application uses a formal indexing process to aid in discovery and search. The primary discovery process is through the formal index data.

Some steps (computing recommendations, reindexing) may be performed asynchronously with user requests. The implementation should be designed to ensure consistency of behaviour and results.

Applicable Standards

The Applicable Standards element lists domain-specific standards required to implement the Service Usage Model as a whole. Standards are described in terms of name, version and citation link. Conformance requirements and extensions should be noted. Standards used to implement applications are excluded. No critical or essential information required to understand the Service Usage Model should be included.

None. No standards are directly applicable to the service usage model as a whole.

The service genres that are a part of the service usage model SHALL BE defined in terms of applicable standards and specifications. Applicable classes of standards and examples include:

- Repository and Registry Description Standards: Standards that define the data models used to describe a registry, repository or content collection, e.g., ISO 2146.
- Content Object Metadata Standards: Standards that describe content object metadata, e.g., DC, LOM, MODS.

- Collection and Repository Description Standards: Standards that define the data models used to describe a content object collection or repository collection.
- Content Packaging and Exchange Standards: Standards used to exchange content objects between clients and the registry, e.g., Content Packaging, METS, MPEG21.
- Harvest Standards: Standards used to harvest objects from a repository, e.g., OAI-PMH.
- Query Communication Standards: Standards for passing queries and query results between a client and the federation, e.g., SRW/SRU, SQL.
- Query Language Standards: Standards for expressing discovery queries, e.g., CQL, A9/Open Search.
- Identifier Standards: Standards used to identify registry and content objects, e.g., HANDLE, OpenURI, Info.
- Auth Standards: Standards supporting authorization and authentication of users and requests, e.g., SAML, XACML.

Known Uses

The Known Uses element documents actual uses of the Service Usage Model in applications and systems, including how used, extensions, limits.

Actual: None as documented.

Potential Application: The TILIS content registry is an example of a registry and repository federation that follows this service usage model.

Potential Application: The ADL-Registry (ADL-R) is an example of a registry and repository federation that follows this service usage model.

Potential Service Usage Model:

portal: [VALUE ADDED] Vx.xx. [[link to service usage model](#)]

The portal service usage model is an integrated set of service genres used to develop a portal-based user interface. The portal service usage model can be used to develop both end user and management portals for a repository federation. Both TILIS and ADL-R use portals to interact with their repository federations.

Resources Used

The Resources element documents all Resources used in the Service Usage Model. Resources are identified by name and version.

The service genres and service usage models within the repository federation service usage model manage the resources. The resources needed to provide essential functionality are labelled [BASE] while others are labelled [VALUE ADDED]. Identified resources include:

metadata registry: [BASE]

Managed storage for metadata objects describing the content objects exposed through the repository federation.

collection registry: [BASE]

Managed storage for collection objects describing the collections exposed through the repository federation.

repository registry: [BASE]

Managed storage for repository objects describing the repositories exposed through the repository federation.

system data: [BASE]

Managed storage for system level objects used to describe, manage and control the registry (vocabularies, schemata, registry descriptions).

access control policy/authorization data: [BASE]

Managed storage for access control policies used to control access to objects managed by the registry.

business rule/policy data: [BASE]

Managed storage for policy and business rules used to control operations of the registry.

archival repository: [VALUE ADDED]

Managed archival storage of content objects separate from their instance in the constituent repositories in the repository federation.

escrow repository: [VALUE ADDED]

Managed escrow storage of content objects separate from their instance in the constituent repositories in the repository federation.

rights licences: [VALUE ADDED]

DRM licences used to control content access.

index data: [BASE]

Indexing information for the metadata registry used in search and discovery.

identifier resolution data: [BASE]

Identifier attributes used in identifier resolution and object access.

identity data: [BASE]

Information about user identities for those who access the registry.

authentication data: [BASE]

Information about authentication of users for those who access the registry.

workflow log: [VALUE ADDED]

Logs of registry operations.

query log: [VALUE ADDED]

Logs of registry queries and search results.

Services (Service Genres) Used

The Service Genres Used element documents all Service Genres used in the Service Usage Model. Service Genres are identified by name and version.
--

The repository federation service usage model includes the service genres listed below. The service genres needed to provide essential functionality are labelled [BASE] while others are labelled [VALUE ADDED]. The service genres that are required for internal operations within the repository federation service usage model but not exposed to end users as service end points are labelled [INTERNAL].

NB: These internal service genres MAY provide useful end user functionality for other applications, but such uses are not considered as part of the repository federation service usage model.

deposit: Vx.xx. [link to service genre]

Add an object (or set of objects) to a registry.

- deposit {metadata registry}: [BASE] Add a metadata object (or set of metadata objects) to the metadata registry.
- deposit {collection registry}: [BASE] Add a collection object to the collection registry.
- deposit {repository registry}: [BASE] Add a repository object to the repository registry.

archive: Vx.xx. [link to service genre]

Archive an object (or set of objects).

- archive {archival repository}: [VALUE ADDED]
Add a content object (or a set of objects) to the archival repository.

escrow: Vx.xx. [link to service genre]

Escrow an object (or set of objects).

- escrow deposit {escrow repository}: [VALUE ADDED]
Add a content object (or a set of objects) to the escrow repository.

delete: Vx.xx. [link to service genre]

Delete an object (or set of objects) from a registry.

- delete {metadata registry}: [BASE]
Delete a metadata object (or set of metadata objects) from the metadata registry.
- delete {collection registry}: [BASE]
Delete a collection object from the collection registry.
- delete {repository registry}: [BASE]
Delete a repository object from the repository registry.

update: Vx.xx. [link to service genre]

Update an object (or set of objects) in a registry. Update is the complete replacement of an object instance by a new object (but maintaining the object identifier).

- update {metadata registry}: [BASE]
Update a metadata object (or set of metadata objects) in the metadata registry. This is the complete replacement of an object instance by a new object (but maintaining the object identifier).
- update {collection registry}: [BASE]
Update a collection object in the collection registry. This is the complete replacement of an object instance by a new object (but maintaining the object identifier).
- update {repository registry}: [BASE]
Update a repository object in the repository registry. This is the complete replacement of an object instance by a new object (but maintaining the object identifier).

change status: Vx.xx. [link to service genre]

Set or change the status of an object (or set of objects) in a registry.

- change status {metadata registry}: [BASE]
Set or change the status of a metadata object (or set of metadata objects) in the metadata registry.

move: Vx.xx. [link to service genre]

Move an object from one container (conceptual or physical) to another.

- move content object {collection registry}: [VALUE ADDED]
Move a content object from one collection to another.
- move content object {repository registry}: [VALUE ADDED]
Move a content object from one repository to another.

augment: Vx.xx. [link to service genre]

Augment an object (or set of objects) in a registry. Add or update (replace) an entire subset of metadata, e.g., rankings, ratings, annotations. A single service genre is defined for any type of metadata augmentation, not one per type.

- augment {metadata registry}: [VALUE ADDED]
Augment a metadata object (or set of metadata objects) in the metadata registry. Add or update (replace) an entire subset of metadata, e.g., rankings, ratings, annotations.
- augment {collection registry}: [VALUE ADDED]
Augment a collection object in the collection registry. Add or update (replace) an entire subset of metadata, e.g., rankings, ratings, annotations.
- augment {repository registry}: [VALUE ADDED]
Augment a repository object in the repository registry. Add or update (replace) an entire subset of metadata, e.g., rankings, ratings, annotations.

search: Vx.xx. [link to service genre]

Search a registry, using a query expressed in a standard query language, returning, at a minimum, a set of object identifiers.

- **search {metadata registry}: [BASE]**
Search the metadata registry, using a query expressed in a standard query language, returning a set of metadata object identifiers and corresponding collection object identifiers.
- **search {collection registry}: [BASE]**
Search the collection registry, returning a set of collection objects identifiers.
- **search {repository registry}: [BASE]**
Search the repository registry, returning a set of repository object identifiers.

obtain: Vx.xx. [link to service genre]

Retrieve an object (or a set of objects) from a registry given the object identifier(s).

- **obtain {metadata registry}: [BASE]**
Retrieve a metadata object (or a set of objects) from the metadata registry given the metadata object identifier(s).
- **obtain {collection registry}: [BASE]**
Retrieve a collection object (or a set of objects) from the collection registry given the collection registry identifier(s).
- **obtain {repository registry}: [BASE]**
Retrieve a repository object (or a set of objects) from the repository registry given the repository registry identifier(s).
- **obtain {archival repository}: [VALUE ADDED]**
Retrieve a content object (or a set of objects) from the archival repository given the content object identifier(s).
- **obtain {escrow repository}: [VALUE ADDED]**
Retrieve a content object (or a set of objects) from the escrow repository given the content object identifier(s).
- **obtain {content repository}: [BASE]**
Retrieve a content object (or a set of objects) from a source constituent repository in the repository federation given the content object identifier(s).

browse: Vx.xx. [link to service genre]

Retrieve an object (or a set of objects) from a registry to enable browsing of the registry. **NB:** The overall browsing process is described by the browse service usage model.

- **browse {metadata registry}: [VALUE ADDED]**
Retrieve a metadata object (or a set of objects) from the metadata registry to enable browsing of the metadata registry.
- **browse {collection registry}: [VALUE ADDED]**
Retrieve a collection object (or a set of objects) from the collection registry to enable browsing of the collection registry.
- **browse {repository registry}: [VALUE ADDED]** Retrieve a repository object (or a set of objects) from the repository registry to enable browsing of the repository registry.

syndicate: Vx.xx. [link to service genre]

Retrieve an object (or a set of objects) from a registry to support syndication of the registry.

- **syndicate {metadata registry}: [VALUE ADDED]**
Retrieve a metadata object (or a set of objects) from the metadata registry to support syndication of the metadata registry.
- **syndicate {collection registry}: [VALUE ADDED]**
Retrieve a collection object (or a set of objects) from the collection registry to support syndication of the collection registry.
- **syndicate {repository registry}: [VALUE ADDED]**
Retrieve a repository object (or a set of objects) from the repository registry to support syndication of the repository registry.

harvest: Vx.xx. [[link to service genre](#)]

Expose an interface to harvest objects from a registry.

- harvest {metadata registry}: [VALUE ADDED]
Expose an interface to harvest metadata objects from the metadata registry.
- harvest {collection registry}: [VALUE ADDED]
Expose an interface to harvest collection objects from the collection registry.
- harvest {repository registry}: [VALUE ADDED]
Expose an interface to harvest repository objects from the repository registry.
- harvest {repository federation}: [VALUE ADDED]
Harvest metadata objects from the constituent repositories in the repository federation.

authenticate: Vx.xx. [[link to service genre](#)]

Authenticate a party (including a service acting on behalf of a user) using information from the authentication data store

- authenticate {agent}: [BASE/INTERNAL]
Authenticate a user agent (or a service acting on behalf of a user agent) using information from the authentication data store. This service genre is part of the identity service usage model.

authorize: Vx.xx. [[link to service genre](#)]

Authorize a user (or a service on behalf of a user) to perform a particular behaviour or access a particular resource using business rules and policies from a nominated data store.

- authorize {access control policy/authorization data}: [BASE/INTERNAL]
Authorize a user (or a service acting on behalf of a user) to perform a particular behaviour or access a particular resource using access control policies in the access control policy/authorization data store. This service genre is part of the authorization service usage model.
- authorize {business rule/policy data}: [BASE/INTERNAL]
Authorize a user (or a service on behalf of a user) to perform a particular behaviour or access a particular resource using business rules and policies in the business rule/policy data store.

generate: Vx.xx. [[link to service genre](#)]

Generate an object based on information available in another object.

- generate {content object basic metadata object}: [VALUE ADDED]
Generate a metadata object containing basic cataloguing and descriptive metadata for/from a content object. The metadata object SHALL conform to an identified metadata standard. This service genre is applied to a content object.
- generate {content object classification metadata object}: [VALUE ADDED]
Generate/compute classification metadata for a content object. This service genre is applied to a content object.
- generate {content object ranking metadata object}: [VALUE ADDED]
Generate/compute ranking metadata for a content object. This service genre is applied to a content object.
- generate {content object rating metadata object}: [VALUE ADDED]
Generate/compute rating metadata for a content object. This service genre is applied to a content object.
- generate {content object recommendation metadata object}: [VALUE ADDED]
Generate/compute recommendation metadata for a content object. This service genre is applied to a content object.
- generate {content object relationship metadata object}: [VALUE ADDED]
Generate/compute relationship metadata for content objects. This service genre is applied to a content object.

synonym transform: Vx.xx. [[link to service genre](#)]

Apply a synonym transform an object.

- synonym transform {query object}: [VALUE ADDED/INTERNAL]
Transform a query through synonym transformations.

spell check transform: Vx.xx. [[link to service genre](#)]

Apply a spell check transform an object.

- spell check transform {query object}: [VALUE ADDED/INTERNAL]
Transform a query through spell check transformations.

translate transform: Vx.xx. [link to service genre]

Apply a translation transform an object.

- language translate transform {query object}: [VALUE ADDED/INTERNAL]
Transform a query through language transformations.

schema transform: Vx.xx. [link to service genre]

Apply a schema transform an object.

- schema transform {metadata object}: [VALUE ADDED/INTERNAL]
Transform a metadata object from one representation to another.
- schema transform {collection object}: [VALUE ADDED/INTERNAL]
Transform a collection object from one representation to another.
- schema transform {repository object}: [VALUE ADDED/INTERNAL]
Transform a repository object from one representation to another.
- schema transform {content object}: [VALUE ADDED/INTERNAL]
Transform a content object from one representation to another.

render transform: Vx.xx. [link to service genre]

Apply a render transform an object.

- render transform {metadata object}: [BASE/INTERNAL]
Apply a rendering transform (XSLT) to metadata object.
- render transform {collection object}: [BASE/INTERNAL]
Apply a rendering transform (XSLT) to collection object.
- render transform {repository object}: [BASE/INTERNAL]
Apply a rendering transform (XSLT) to repository object.
- render transform {content object}: [BASE/INTERNAL]
Apply a rendering transform (XSLT) to content object.

filter: Vx.xx. [link to service genre]

Filter (via provided data) the object (or set of objects) as part of an access control workflow.

- filter {metadata registry}: [VALUE ADDED/INTERNAL]
Filter (via authorization data) the metadata object (or a set of objects) retrieved from the metadata registry as part of a metadata object access workflow. Internally, the filter service genre uses the access control policy/authorization data resource.
- filter {collection registry}: [VALUE ADDED/INTERNAL]
Filter (via authorization data) the collection object (or a set of objects) retrieved from the collection registry as part of a collection object access workflow. Internally, the filter service genre uses the access control policy/authorization data resource.
- filter {repository registry}: [VALUE ADDED/INTERNAL]
Filter (via authorization data) the repository object (or a set of objects) retrieved from the repository registry as part of a repository object access workflow. Internally, the filter service genre uses the access control policy/authorization data resource.
- filter {repository}: [VALUE ADDED/INTERNAL]
Filter (via authorization data) the content object (or a set of objects) retrieved from a repository as part of a repository access workflow. Internally, the filter service genre uses the access control policy/authorization data resource.

validate: Vx.xx. [link to service genre]

Validate the representation of an object.

- validate {metadata object}: [BASE/INTERNAL]
Validate the representation of a metadata object.

- validate {collection object}: [BASE/INTERNAL]
Validate the representation of a collection object.
- validate {repository object}: [BASE/INTERNAL]
Validate the representation of a repository object.

register identifier: Vx.xx. [link to service genre]

Create an identifier including resolution data, and publish the identifier to the identifier resolution data store. This service genre is part of the identifier service usage model.

- register identifier {object}: [BASE/INTERNAL]
Create and publish an identifier for an object. Internally, the service genre uses the identifier resolution data resource.

resolve identifier: Vx.xx. [link to service genre]

Resolve an identifier and return the resolution data from the identifier resolution data store. This service genre is part of the identifier service usage model.

- resolve identifier {identifier object}: [BASE/INTERNAL]
Resolve an identifier object. Internally, the service genre uses the identifier resolution data resource.

update identifier resolution data: Vx.xx. [link to service genre]

Update the resolution data in the identifier resolution data store. This service genre is part of the identifier service usage model.

- update identifier resolution data {identifier object}: [BASE/INTERNAL]
Update resolution data for an identifier object. Internally, the service genre uses the identifier resolution data resource.

package: Vx.xx. [link to service genre]

Apply a packaging transform (e.g., IMS Content Packaging, METS, MPEG21) to an object (or a set of objects).

- package {content object}: [VALUE ADDED]
Apply a packaging transform to a content object (or a set of content objects).

log: Vx.xx. [link to service genre]

Log an activity or action in a log object.

- activity log {workflow log}: [VALUE ADDED/INTERNAL]
Log an activity/action in the workflow log.
- query/results log {query log}: [VALUE ADDED/INTERNAL]
Log a query or query results in the query log.

Related Service Usage Models

The Related Service Usage Models element documents and illustrates how the Service Usage Model is related to other Service Usage Models required to provide its stated behaviours. Related Service Usage Models are identified by name and version. No critical or essential information required to understand the Service Usage Model should be included.

The service usage models listed below are assumed to exist.

identifier: [BASE] Vx.xx. [link to service usage model]

The identifier service usage model is an integrated set of service genres that provides the entire identifier infrastructure used by the repository federation service usage model. Functionality includes provisioning and management of the identifier system and the creation, registering, publishing, managing and resolving of individual identifiers.

repository storage manager: [BASE] Vx.xx. [link to service usage model]

The storage manager service usage model is an integrated set of service genres that provides access operators (CRUD, versioning, audit, ACID transactions) for a stored data collection, e.g., the metadata registry, collection registry, repository registry, archival repository, escrow repository.

identity: [BASE] Vx.xx. [link to service usage model]

The identity service usage model is an integrated set of service genres that provides the entire user identity infrastructure used by the repository federation service usage model. Functionality includes provisioning and management of the identity system, the creation of user right and roles, user authentication and user authorization.

NB: Authentication and authorization are combined within a single service usage model.

browse: [VALUE ADDED] Vx.xx. [link to service usage model]

The browse service usage model provides an integrated set of service genres used to develop browser-based interfaces for a repository federation.

rights management: [VALUE ADDED] Vx.xx. [link to service usage model]

The rights management service usage model is an integrated set of service genres that provides rights management for content (including metadata) in the repository federation. Functionality includes provisioning and management of the rights system and the creation and management of rights expressions.

delivery: [VALUE ADDED] Vx.xx. [link to service usage model]

The delivery service usage model integrates a set of service genres to provide support for discovery-to-delivery of content objects from a repository federation.

repository provisioning: [BASE] Vx.xx. [link to service usage model]

The repository provisioning service usage model integrates a set of service genres to provide support for creating a repository that has authorisation and authentication. It maps instances of users and user authentication and authorisation to the infrastructure provided by the identity service usage model, and orchestrates identity services with other repository services. It also provisions the business rules and system data structures used by a repository in its operations.

manage discovery service: [BASE] Vx.xx. [link to service usage model]

The manage discovery service service usage model integrates a set of service genres to provide support for the ongoing maintenance of infrastructure required by a discovery service, outside the content and metadata of the objects targeted by the service. Such infrastructure includes indexes, search rankings, and classifications.

manage repository: [BASE] Vx.xx. [link to service usage model]

The manage repository service usage model integrates a set of service genres to provide support for the ongoing maintenance of a repository. Such maintenance includes monitoring and analysing repository behaviour, and inspecting repository contents.

Service Patterns/CORE Service Usage Models Used

The Service Patterns/CORE Service Usage Models element documents all CORE Service Usage Models actually used in the Service Usage Model. CORE Service Usage Models are identified by name and version.
--

Actual: None as documented.

Potential: AVTEROFTRL workflow, Vx.xx. [link to CORE service usage model]

Many of the registry operations follow a common workflow pattern that could be abstracted. The AVTEROFTRL workflow core service usage model defines this pattern.

Steps in this commonly reoccurring pattern include:

- Authenticate and authorize: authenticate and apply business rules to validate the request/operation [A].
- Validate: validate the data object associated with the request/operation [V].
- Transform: apply a (schema) transform to the data object associated with the request [T].
- Enhance: enhance or modify the data object associated with the request (adding data not in the original request) [E].
- Route: route the request to a specific operator [R].
- Operate: perform the requested operation and obtain results [O].
- Filter: filter the results (apply access controls and other filters to the results) [F].

- Transform: apply a (schema) transform to the results data object [T].
- Render: apply a rendering transform to the results data object [R].
- Log: log the activity [L].

Of these steps only AVROL (Authenticate and Authorize; Validate; Route; Operate; Log) are compulsory and recur for every instance of the workflow. The remaining steps are all optional, and are included in the workflow only as required by the business logic. If steps are omitted, then the data input to the workflow is what is required as input to the operation, and the operation output produces what is required by the workflow consumer, with no further modification.

Potential: ROAP obtain, Vx.xx. [[link to CORE service usage model](#)]

The ROAP obtain core service usage model is a combination of four services used to access an object given an object identifier. Steps in this commonly reoccurring pattern include:

- A request, in the form of an identifier, is resolved (including multiple resolution based on a FRBR model to obtain the appropriate work, expression, manifestation or copy) yielding the source resource that manages the object [R].
- An obtain accessor is used to get the object [O].
- Subject the object to access control or obligation filtering [A].
- The resulting object dissemination is encoded and packaged in the requested format [P].

Search is used as a precursor to the ROAP obtain process to discover the object identifier. Selected object identifiers are then passed through the ROAP obtain workflow to obtain the object. The ROAP obtain service usage model itself is an operator in an AVTEROFTRL workflow.

References

The References element includes references and bibliographic citations to works needed to understand the Service Usage Model.

Glossary

The Glossary element defines domain-specific terms used in documenting the Service Usage Model.

collection: A named, managed set of content objects. A collection may span multiple repositories. A collection is a type of resource.

collection object: Descriptive metadata, objective or subjective, for a collection. A collection object is an object managed by a collection resource.

content object: An object, digital or physical; part of a managed collection of content objects.

FRBR: Functional Requirements for Bibliographic Records: A standard for the modelling of entities in library cataloguing, and for the relations between them. The FRBR ontology of entities is used in the Service Usage Model to differentiate content objects which may be regarded as the same at some underlying levels but not others.

FRBR Work: An abstract representation of a content object as a presentation of content. The most general representation of a element of content, independent of its presentation, encoding or storage location. Two content objects present the same FRBR Work if their content is deemed to be underlyingly the same, ignoring revisions and reformulations. Two content objects belong to distinct FRBR Works if their content is significantly different.

FRBR Expression: An abstract representation of a content object as a presentation of content. Two content objects present the same FRBR Expression if their content is identical, and the objects differ only in the presentation of that content (including the file format). Two content objects are distinct FRBR expressions of the same FRBR work if the content of one is a minor modification of the content of the other (e.g., different drafts or versions).

FRBR Manifestation: A representation of a content object as a presentation of content. Two content objects present the same FRBR Manifestation if their content is identical and the presentation and storage and representational encoding of the content is identical—i.e., they are they same file, though the location and access metadata of the

objects may be different. Two content objects are distinct FRBR Manifestations of the same FRBR expression if their content is identical, but the presentation of the content differs (e.g., different file encoding, different file format).

FRBR Instance: A content object viewed in the context of the FRBR model, bound to a specific single storage location. All content objects are FRBR instances; all content objects can be mapped to a specific FRBR Manifestation, FRBR Expression, and FRBR Work, as increasingly abstract representations of the object content.

repository federation: A managed set of repositories, containing content objects (that are within collections) and a registry used to store metadata (metadata objects) used for discovery.

metadata object: Descriptive metadata, objective or subjective, for a content object.

registry: A repository containing metadata (metadata objects) for content objects from a set of repositories. In the federation model described, the registry also contains collection objects and repository objects. A registry is a type of resource.

metadata registry: A repository containing metadata about content objects, i.e., metadata objects.

collection registry: A repository containing metadata about collections, i.e., collection objects.

repository registry: A repository containing metadata about repositories, i.e., repository objects.

registry object: A metadata object describing a registry. A registry object is an object managed by a registry resource.

repository: A storage and management system for one or more sets of content objects or metadata objects. A repository is a type of resource.

repository object: A metadata object describing a repository. A repository object is an object managed by a repository resource.

user: Any application or client accessing a service. The client or application may be a proxy for a human user / end user.