

## Harvest Service Expression: *harvest resource—oai-pmh basic*

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. 2008."

### 1 Introduction

This service expression is a specialization of the harvest service genre. It exposes a web service that uses the OAI-PMH specification for harvesting metadata. All of the OAI-PMH operators (verbs) are supported. The resource (provider repository) being harvested exposes an interface for harvesting. Communications to the provider service implementation interface are represented in requests defined using HTTP. Requests and (dissemination) results are communicated between the client/requestor and the provider service implementation using HTTP, with results encoded in XML. The service expression supports disseminating unqualified Dublin Core metadata from the resource. The service implementation does not include a mechanism to authorize clients. How to discover service implementations that support harvesting is out of scope.

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].

The service expression description that follows uses e-Framework service expression description elements as of 2006-12-09. Other terms, e.g., client, provider, resource, are used as defined in the e-Framework. These definitions may conflict with those used in the OAI-PMH specification.

Items are tagged and identified using names assigned by the Fred project. Formal e-Framework names will be assigned by the e-Framework.

### 2 Service Expression Definition

#### 2.1 Name

- Fred Service Expression Name: harvest resource—oai-pmh basic
- e-Framework Service Expression Registry Name: TBD

#### 2.2 Classification

- Purpose: Harvest (Information Dissemination)
- End point: Requestor

Copyright © University of Southern Queensland and University of Memphis.



This work is licensed under the Creative Commons Attribution-Share Alike 2.5 Australia License. To view a copy of this license, visit <http://creativecommons.org/licenses/by-sa/2.5/au/>

This work is created as part of the Federated Repositories for Education (FRED) Project within the Australian ADL Partnership Laboratory. The FRED project is sponsored by the Australian Commonwealth Department of Education, Science and Training under the Framework for Open Learning Programme.

- Domain: Learning & Teaching
- Scope: Multiple
- Structure: Individual
- Visibility: Public
- Behaviour: Transactional
- Community: FRED
- Status: prototype

### 2.3 Service Genre

- Fred Service Genre Name: harvest [[link to service genre](#)]
- e-Framework Service Genre Registry Name: TBD [[link to e-Framework service genre](#)]

### 2.4 Version

- Fred Version: 1.0.0
- e-Framework Service Expression Version: TBD

#### Version History

Version	Date	Author	Description
0.1	2006-12-10	DR	Initial version hdl:FredNA/2EBE8F9695864E6C8C87C3A352526EAD
0.11	2006-12-19	NN	Taken out flags by Dan for work to be done in LOM WS version
0.12	2007-01-19	NN	Editorial; inserted full examples, attempted classification
1.0.0	2008-06-18	DR	Editorial to align with LSAL ADL work

### 2.5 Description

The harvest resource—oai-pmh basic service expression specializes the harvest service genre through binding to the OAI-PMH specification and further constrains the service implementations with additional constraints that are specific to the FRED community.

Repositories expose a “harvest” interface, defined by this service expression. Clients may send requests to this interface to gather the metadata stored in the repository. The OAI-PMH specification details some of the requirements on such an interface. This service expression follows OAI-PMH. Communications between the clients that harvest metadata from the repository and service implementation interface (repository interface) are through HTTP requests, not web services. Repositories are only required to expose Dublin Core metadata for objects in the repository. This service expression places additional requirements specific to the FRED community on all service implementations with the purpose of increasing interoperability.

Repository harvest interfaces are not access controlled, i.e., any client may attempt to harvest the metadata. There are no authentication controls. The repository is responsible

for determining what results it will return. How a repository and service implementation decide how to respond to requests (i.e., what they expose) is not specified. The service expression only details the format and encoding of requests and responses, and specifies constraints and requirements on service implementations. Repositories are not required to store metadata in any particular format. They are only required to disseminate Dublin Core metadata for objects upon request.

## **2.6 Functionality**

As specified by OAI-PMH, the harvest resource service expression supports six distinct functions.

- **Identify:** Get descriptive information about the repository (the resource behind the service implementation interface) and the capabilities of the service implementation. This information enables a client to successfully communicate with the repository.
- **ListMetadataFormats:** Get the set of metadata formats that the repository is capable of providing as results for a harvest request. The repository may be capable of returning (disseminating) the metadata in different formats. This information enables a client to request metadata in a particular format.
- **ListSets:** Get information about the structure of the repository. Objects in the repository may be grouped into "sets". This information enables a client to selectively harvest data by specifying the parts of the repository to be harvested.
- **GetRecord:** Get the metadata for a single specific object in the repository. The client may request the desired metadata format. This function enables the client to get the metadata for a single item.
- **ListRecords:** Get metadata records from the repository. This function enables the client to get the metadata for a selected collection of items. The client may request selective harvesting by specifying a date range or by specifying a part of the repository (via "sets"). Metadata records are returned for the objects that match the selection criteria. Mechanisms exist to provide flow control so the results may be returned in chunks.
- **ListIdentifiers:** Get the identifiers for objects in the repository. This function enables the client to get the identifiers for a selected collection of items. The client may request selective results by specifying a date range or by specifying a part of the repository (via "sets"). Only the identifiers are returned for the objects that match the selection criteria. Mechanisms exist to provide flow control so the results may be returned in chunks.

The service expression provides the capability to return unqualified Dublin Core metadata records for objects.

A service implementation MAY support other metadata formats.

Communications to the service implementation is via requests encoded as HTTP methods and transmitted using HTTP.

No other functionality is provided. The functionality defined SHALL NOT be extended.

## **2.7 Usage Scenarios**

Harvesting occurs to facilitate discovery of assets in the provider repository: discovery is transacted on the client repository instead, by means of the metadata harvested. The main reason to do this is when a single client repository gathers metadata from multiple provider repositories. In that case, discovery across the range of repositories involves only a single

query on the client, rather than a separate query for each provider. The providers form a federation of repositories by having their metadata harvested and centralised; search on the client repository is then termed *federated search*. The federation can be ad hoc, with metadata harvested as found on open access repositories; or it can be formal, with policy guarantees such as uniform metadata profiles and service level agreements underpinning it. The typical scenario is:

1. Client *harvests* metadata instance for an item from the provider repository
2. Client *validates* the submitted metadata against the registry schema
3. Metadata is *associated* with content item
4. Client *exposes* metadata to end users

The *associate* step is necessary only if the identifier used at the client repository to refer to content is distinct from that used at the provider repository. If both repositories use the same global identifier, it is skipped.

Once a metadata record is discovered on the client through a query, the user gains access to the content item referred to by the metadata record. This access occurs through a locator or identifier inside the metadata record harvested from the provider, possibly converted to the client's namespace through the *associate* step. The identifier for content is logically distinct from the identifier for the metadata record referring to it; the metadata record identifier is the parameter used by OAI-PMH. The content item need not reside on the provider repository.

The metadata on the client repository is expected to stay reasonably up to date. As a result, harvesting will usually be a periodically recurrent activity, and restricted only to those metadata records updated on the provider repository since the last harvesting session.

A client may not be interested in ingesting all the metadata records available from the provider, but only those relevant to the client repository subject matter. The provider offers a mechanism of restricting harvesting to those records identified as relevant; in OAI-PMH, this is done through "sets".

Harvesting proper is done through the ListRecords function. Of the other functions, Identify and ListMetadataFormats provide the client with the information necessary to set harvesting up, and choose the metadata profile relevant. ListSets allows the client to restrict harvesting to the subsets of the provider repository contents it finds relevant. The scenario outlined fetches metadata as a batch rather than as individual records (GetRecord). It gathers complete metadata records, including local identifiers, in order to allow discovery on the client repository. The alternative is that it fetches only the headers of a set of records, e.g. recently updated records (ListIdentifiers), and either fetches the full records piecemeal, or makes the user browse them on the provider repository.

So, integrating these in on overall workflow:

A client acting on behalf of a federation of vocational education repositories harvests metadata from a participating provider.

- What version of OAI-PMH does the provider support, and how does it handle date stamps and deleted records? (Identify)
- Given that: does the provider allow selective harvesting of only records describing vocational education modules? (ListSets; this presupposes an agreed vocabulary of sets, and that the provider may contain items other than vocational education)
- Alt 1: Request all headers of relevant metadata records. This is summary information about the metadata records, including when records have been updated and what

the identifiers for the records are. The client may retrieve the corresponding records later, or redirect the end user to those metadata records on the provider.  
(ListIdentifiers)

- Alt 2: Request all relevant metadata records. These will be ingested into the client repository (ListRecords)
- Alt 3: Given the identifier for a metadata record (which may have been retrieved through ListIdentifiers), request the full corresponding metadata record. (GetRecord)

## 2.8 Applicability

The service expression is applicable for harvesting Dublin Core metadata from repositories without access controls.

The service expression MAY be extended (by specifying the metadata format) if harvesting other metadata formats is required.

The service expression is not applicable when the repository requires authentication to permit harvest. The service expression SHALL NOT be extended to support authentication.

**NB:** Such extensions require a separate service expression.

The service expression is not applicable when the repository filters the results that are returned based on authorization policies or rules that need to be communicated to the repository through the harvest interface. The service expression SHALL NOT be extended to support authorization.

**NB:** Such extensions require a separate service expression.

**NB:** The repository may filter results by other means.

The service expression is not applicable if communications need to be secure. The service expression SHALL NOT be extended to support secure or encrypted communications.

**NB:** Such extensions require a separate service expression.

## 2.9 Requests & Behaviours

The format for request and responses are defined in the OAI-PMH specification. Six requests (verbs) SHALL be supported:

Identify (See OAI-PMH specification clause 4.2). The request and response SHALL be as specified. Additional requirements:

- The encoded repositoryName value SHALL NOT exceed 255 bytes in length.
- The protocolVersion value SHALL be 2.0.
- Any encoded adminEmail value SHALL NOT exceed 255 bytes in length.
- The granularity value (minimum granularity of datestamp) SHALL be the finest granularity supported by the service implementation.

ListMetadataFormats (See OAI-PMH specification clause 4.4). The request and response SHALL be as specified. Additional requirements:

- If the service implementation supports the dissemination of IEEE LOM metadata, the description of the metadataFormat SHALL be as follows:

```
<metadataFormat>
  <metadataPrefix>ieee-lom</metadataPrefix>
  <schema>http://ltsc.ieee.org/xsd/lomv1.0/lom.xsd</schema>
```

```
<metadataNamespace>http://ltsc.ieee.org/xsd/LOM</metadataNamespace>  
</metadataFormat>
```

ListSets (See OAI-PMH specification clause 4.6). The request and response SHALL be as specified. Additional requirements:

- The encoded resumptionToken value SHALL NOT exceed 255 bytes in length.

GetRecord (See OAI-PMH specification clause 4.1). The request and response SHALL be as specified. Additional requirements:

- The encoded identifier value SHALL NOT exceed 255 bytes in length.

ListRecords (See OAI-PMH specification clause 4.5). The request and response SHALL be as specified. Additional requirements:

- The encoded resumptionToken value SHALL NOT exceed 255 bytes in length.

ListIdentifiers (See OAI-PMH specification clause 4.3). The request and response SHALL be as specified. Additional requirements:

- The encoded identifier value SHALL NOT exceed 255 bytes in length.
- The encoded resumptionToken value SHALL NOT exceed 255 bytes in length.

## **2.10 Use & Interactions**

The model for a client to interact with a service implementation is defined in the OAI-PMH specification. Except as specified herein, the service implementation SHALL conform to the OAI-PMH specification.

The service implementation SHALL implement flow control (See OAI-PMH specification clause 3.5) with the following additional requirements:

- Flow control SHALL NOT be used if the number of records in the response is less than 1001. All records SHALL be returned in a single results set.
- The number of records in the results set SHALL NOT exceed 1000.
- The resumptionToken SHALL include an expirationDate. The resumptionToken SHALL be valid for at least 10 minutes from the time the service implementation transmits the response to the client.

Responses for all requests SHALL include all applicable error codes (See OAI-PMH specification clause 3.6).

OAI-PMH errors SHALL be communicated in the OAI-PMH response, as specified in the results schema, while SOAP errors SHALL be handled at the SOAP protocol level.

## **2.11 Structure**

A specific structure for an implementation and design are not specified. Implementation guidance may influence the design (see the *Implementation Guide & Dependencies* element for references). The OAI-PMH specification places requirements on the structure of a service implementation as listed below.

The data model for OAI-PMH harvest is given in the OAI-PMH specification, section 2. Key requirements:

- Repositories contain metadata describing content items (denoted *resources* in OAI-PMH), which may or may not be stored in the same repositories.

- These metadata descriptions are stored or generated by *items* which reside on the repository.
- These metadata descriptions are stored or generated in the form of *records*.
- There is a distinct record for each *metadata format*.
- Items are identified by *identifiers*. The combination of identifier and metadata format identifies a unique metadata record.
- Items can be assigned to *sets*, which enable selective harvesting.
- Records are returned as XML, containing:
  - A header containing:
    - The item identifier
    - The *timestamp* for the record
    - The set membership of the item
    - An optional *status* attribute, indicating that the item has been deleted from the repository
  - The metadata
  - An optional *about* container of data, which is metadata about the metadata—including, for example, a rights statement for the metadata record, or a description of the provenance of the metadata record.

All OAI-PMH requests are read-only, and do not change the state of the provider repository.

The List requests (ListRecords, ListIdentifiers, ListSets) support flow control when the requested data may be too large to return in a single response. Instead of a single request query, OAI-PMH allows a List transaction to consist of multiple request–response pairs; each incomplete response will contain a resumption token, which the client must use to obtain the next segment of the response. The provider must therefore treat List requests as stateful: it generates a listing of sets or items in response to the query (and extracts identifiers or records from the items in turn), but it must remember how many sets or items it has already presented to the client. Resumption tokens have optional expiration dates, so the provider **MUST** to commit to holding state information on List requests for a specified period of time.

Between queries, the set of records that the initial request specified may change, as new records are created, modified or deleted. If there is a change, the provider **MAY** return an error indicating that the request should be reinitiated. If the request is honoured, the provider **MUST** return all unchanged records. It **MAY** update its listing of sets or items to reflect the intervening changes, excluding or including changed records.

Providers are expected to notify clients if any items have been deleted; this will allow the client to delete the corresponding record from its own repository. Thus the provider must retain a log of the item, even if the item itself has been deleted. This will allow it to generate a record in response to a harvest request for the item, indicating that the item is no longer available. Providers have the option of not retaining information on deleted items, retaining it temporarily, or retaining it permanently; they inform clients of how they treat deletion through the Identify request.

Providers are also expected to notify clients if any individual metadata records have been deleted (or become unavailable), even if the item generating them remains in the provider's repository. If a provider discontinues

supporting a particular metadata format, the corresponding metadata records are deemed to be deleted, although other metadata records remain.

Selective harvesting of records may be specified by datestamp, so that the client limits its request to those records created, modified or deleted within the given date span. A provider MUST commit to keeping datestamps for all changes to the item resulting in an altered metadata record, and to exposing the mapping between datestamp and record (i.e., an index of items by most recent datestamp). This includes all creation and deletion of items, and all updates in metadata values, as well as all changes in metadata format (which will result in a different metadata record even if the content of the record is unchanged). Providers establish the date granularity.

Selective harvesting of records may be parameterized by set; this means that the provider must expose a mapping between set and record (i.e., an index of items by the set(s) they have been associated with).

## 2.12 Interface Definition

The format for OAI-PMH GET and POST HTTP methods are defined in the OAI-PMH specification. Both methods SHALL be supported. Other HTTP methods SHOULD NOT be supported.

Example:

GET Request

```
GET http://arrow.edu.au/oai?verb=GetRecord&identifier=hdl:1870/22FA672A787C4A0B82D8B0D7553ACE2B&metadataPrefix=oai_dc HTTP/1.1
```

POST Request

```
POST http://arrow.edu.au/oai? HTTP/1.1
Content-Length: 96
Content-Type: application/x-www-form-urlencoded
Date: Tue, 15 Nov 1994 08:12:31 GMT
Host: arrow.edu.au
User-Agent: Mozilla/5.0 (compatible; iCab 3.0.3; Macintosh; U; PPC Mac OS X)

verb=GetRecord&identifier=hdl:1870/22FA672A787C4A0B82D8B0D7553ACE2B&metadatanPrefix= oai_dc
```

Response

```
HTTP/1.1 200 OK
Content-Length: 96
Content-Type: application/text+html; charset=utf-8; action = http://www.fred.net/wsdl/RegistryInterface#Identity
Accept: application/soap+xml
Date: Tue, 15 Nov 1994 08:12:35 GMT
Last-Modified: Tue, 15 Nov 1994 08:12:33 GMT

<?xml version="1.0" encoding="UTF-8"?>
<OAI:OAI-PMH xmlns:OAI="http://www.openarchives/OAI/2.0/"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:schemaLocation="http://www.openarchives.org/OAI/2.0/
    http://www.openarchives.org/OAI/2.0/OAI-PMH.xsd">
  <responseDate>2002-05-01T19:20:30Z</responseDate>
  <request verb="GetRecord"
```

```
        identifier="hdl:1870/22FA672A787C4A0B82D8B0D7553ACE2B"
        metadataPrefix="oai_dc">arrow.edu.au/oai</request>
<GetRecord>
  <record>
    <header>
      <identifier>hdl:1870/22FA672A787C4A0B82D8B0D7553ACE2B</identifier>
      <datestamp>1993-12-14</datestamp>
    </header>
    <metadata>
      <oai_dc:dc
        xmlns:oai_dc="http://www.openarchives.org/OAI/2.0/oai_dc/"
        xmlns:dc="http://purl.org/dc/elements/1.1/"
        xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
        xsi:schemaLocation="http://www.openarchives.org/OAI/2.0/oai_dc/
        http://www.openarchives.org/OAI/2.0/oai_dc.xsd">
        <dc:title>Using Structural Metadata to Localize Experience of
          Digital Content</dc:title>
        <dc:creator>Dushay, Naomi</dc:creator>
        <dc:subject>Digital Libraries</dc:subject>
        <dc:description>With the increasing technical sophistication of
          both information consumers and providers, there is
          increasing demand for more meaningful experiences of digital
          information. We present a framework that separates digital
          object experience, or rendering, from digital object storage
          and manipulation, so the rendering can be tailored to
          particular communities of users.
        </dc:description>
        <dc:description>Comment: 23 pages including 2 appendices,
          8 figures</dc:description>
        <dc:date>2001-12-14</dc:date>
      </oai_dc:dc>
    </metadata>
  </record>
</GetRecord>
</OAI-PMH>
```

Except as specified herein, or in the OAI-PMH specification, the service implementation SHALL conform to the HTTP 1.1 specification.

- The GET and POST methods SHALL accept URIs of at least 4000 bytes in length. An HTTP status 414 Request-URI Too Long SHALL be returned if the length of the URI exceeds that which the service implementation can process.
- The POST method SHALL accept a message with Content-Length of at least 4000 bytes in length. An HTTP status 414 Request-URI Too Long SHALL be returned if the Content-Length exceeds that which the service implementation can process.
- Requests SHALL include a From request-header field with a well formed email address. The encoded email address SHALL NOT exceed 255 bytes. The service implementation SHALL return an HTTP status 400 Bad Request error if the field is missing or malformed.
- Requests SHALL include a User-Agent request-header field. The encoded User-Agent value SHALL NOT exceed 255 bytes. The service implementation SHALL return an HTTP status 400 Bad Request error if the field is missing.
- Responses SHALL include a no-cache pragma-directive.
- The service implementation MAY include load balancing. The service implementation may return an HTTP status 302 Found to redirect the client to a different service instance. The client SHOULD redirect the request to the indicated service instance.
- A service implementation SHALL return an HTTP status 503 Service Unavailable when it is too busy to respond to a request. The HTTP response SHALL include a Retry-After value. The client SHOULD obey the response and not retry the request until the specified time has expired. If the client retries the request too soon, the service implementation SHALL return an HTTP status 403 Forbidden.
- The service implementation SHALL support HTTP compression (See OAI-PMH specification clause 3.1.3). The service implementation SHALL support gzip compression.
- A service implementation SHALL return an HTTP status 400 BAD Request when the request contains code injection or other malicious elements.

### **Machine Processable Schemata**

The XML schema for results is described in the OAI-PMH specification. The schema is available at: <http://www.openarchives.org/OAI/2.0/OAI-PMH.xsd> Additional constraints on the sizes of XML elements in the result are defined in the *Requests & Behaviours* element within the service expression description.

The XML schema for Dublin Core metadata results is described in the OAI-PMH specification. The schema is available at: [http://www.openarchives.org/OAI/2.0/oai\\_dc.xsd](http://www.openarchives.org/OAI/2.0/oai_dc.xsd)

### **2.13 Applicable Standards**

Service Implementations of the service expression SHALL conform to the following standards:

*The Open Archives Initiative Protocol for Metadata Harvesting, Protocol Version 2.0*

<http://www.openarchives.org/OAI/openarchivesprotocol.html>

*Dublin Core Metadata Element Set, Version 1.1: Reference Description*

<http://dublincore.org/documents/dces/>

Hypertext Transfer Protocol -- HTTP/1.1

<http://www.ietf.org/rfc/rfc2616.txt>

### **2.14 Design Decisions & Tradeoffs**

Metadata Formats: The service expression extends the capabilities of the basic OAI-PMH service expression.

- A namespace for harvesting IEEE LOM metadata is specified.
- Supporting other metadata formats require that additional namespaces be specified for the ListMetadataFormats verb in *Requests & Behaviours* element. Adding a namespace insures interoperability, but does not require all service implementations to be capable of harvesting the specific type of metadata. There is no agreed upon name for the other metadata formats.
- If additional metadata disseminations were added, harvest for these metadata formats could be made mandatory. Requiring a dissemination of metadata in a specific format imposes an additional implementation burden on a service implementation. Requiring support for harvest of a particular metadata format should only added if the support is required across all service implementation.

### **2.15 Implementation Guide & Dependencies**

XML:

- A service implementation SHALL conform to all aspects of the *FRED Profile for Core Service Standards*.

Metadata Disseminations:

- A service implementation SHALL "disseminate" Dublin Core metadata. The service implementation must only be capable of disseminating the required metadata formats. The service implementation is not required to store the metadata in the required format (or in any format).

Consistency:

- The service implementation SHALL ensure that the time stamps for when content is added or modified are consistent with those returned to the client such that the metadata for all objects is harvestable.

Performance:

- A service implementation SHALL be capable of handling simultaneous requests from different clients.
- A service implementation SHOULD implement an indexing scheme or equivalent method to permit efficient harvesting by date ranges.
- Load balancing SHOULD be implemented for large repositories or those which are harvested frequently (continuously).

Interoperability:

- Clients should be aware that URIs longer than 255 bytes may not be supported by intermediaries (caches, proxies).

Security:

- A service implementation SHALL inspect all requests for possible code injection.
- A client SHOULD validate all XML results against appropriate schemas.
- Service implementation may be subject to denial-of-service attacks.

- Administrator email addresses are returned to the client. Requests include a From request-header field with an email address. Care should be taken to maintain privacy of email addresses.
- Service implementation may log request and results. Security of logs should be maintained.
- There are no authorization or authentication controls. Care should be taken to maintain data privacy.
- While a service implementation may return provenance data, clients are not obliged to process it.

Additional implementation guidance is available in:

*Implementation Guidelines for the Open Archives Initiative Protocol for Metadata Harvesting*

<http://www.openarchives.org/OAI/2.0/guidelines.htm>

with specific:

- *Guidelines for Repository Implementers*  
<http://www.openarchives.org/OAI/2.0/guidelines-repository.htm>
- *Guidelines for Harvester Implementers*  
<http://www.openarchives.org/OAI/2.0/guidelines-harvester.htm>
- *Guidelines for Aggregators, Caches and Proxies*  
<http://www.openarchives.org/OAI/2.0/guidelines.htm>

## **2.16 Known Uses**

Actual: None

Potential: The service expression could be used in a service usage model for a repository federation containing a central federated metadata registry. A federated metadata registry would be the client that sends harvest requests to the service implementations providing harvest interfaces to the repositories in the federation. The requests would be used to gather the metadata used to populate the federation registry. The client would periodically harvest the repositories in the federation to obtain updates to the objects held in the repositories. The federated metadata registry could also provide a service implementation interface to allow other clients to harvest the metadata in the federation registry, building a federation of federations.

## **2.17 Service Expression Dependencies**

None.

## **2.18 Related Service Expressions**

- Fred Service Expression Name: harvest resource—oai-pmh lom ws, [Vx.xx](#). The harvest resource—oai-pmh lom ws service expression provides the same core functionality as the harvest resource—oai-pmh basic service expression. The harvest resource—oai-pmh lom ws service expression is required to support LOM metadata, and communications are via web services, using SOAP over HTTP transport.

## **2.19 Related Service Usage Models**

- (FRED) repository federation: [V1.0.0](#).  
[[link to service usage model](#)]  
Harvest (genre) is a part of the repository federation service usage model (genre based)

and is used to gather metadata from the repositories and collections that participate in the federation to build the registry data used for discovery.

***2.20 Related Service Patterns***

None.