



e-Framework Service Expression

OpenURL Appropriate Location + HTTP + Inline KEV

1 Rationale

This service expression is a specification of the OpenURL protocol for use in the FRED project, which provides infrastructure for repository federation within the e-Learning space in Australia. One of the goals of FRED is to provide a toolkit on which FRED stakeholders can base an appropriate copy service, to deploy in their own learning content federations. This service expression is intended to support development of an appropriate copy service, whereby a FRED federation can select which instance of an object to deliver to a user, out of all the instances stored on the federation.

OpenURL service calls use parameters derived from a model of the context entities involved in an Obtain request. A toolkit for FRED stakeholders must support the models of context entities that are relevant to them. Based on FRED business analysis, particular parameters have been identified as required information for a FRED deployment of an appropriate copy service. This service expression describes the context entity model specific to FRED, and the interface specification for OpenURL which a FRED Appropriate Copy service needs to use in order to support that model.

FRED stakeholders are expected to customise any FRED Appropriate Copy toolkits (including this service expression), in order to address their specific needs when deploying an appropriate copy service in their own federations. This service expression does not concentrate on the requirements of any one stakeholder, but rather addresses the requirements common to FRED stakeholders.

To the end user attempting to obtain an object from a federation, the Obtain request is addressed to a unitary data source, and the differentiation between instances and repositories that this service expression makes are irrelevant. In other service genres and expressions written for FRED (e.g. SRW Search), and in CORDRA thinking about content delivery from a federation, the storage specifics of objects are considered irrelevant. Unlike other service expressions developed under FRED, this service expression is not used in isolation by an end user, and does not address an end user functional requirement: the end user only requests Obtain for an object, and this service expression is used within the delivery system to convert that request to a retrieval key used by Obtain for a particular instance. This functionality is nonetheless described in a service expression, given that the appropriate copy problem it addresses is a recurring non-functional requirement for repository federations.

Copyright © University of Southern Queensland.



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. The Australian ADL Partnership Laboratory is supported by the University of Southern Queensland.

The template structure and format of this document are based on e-Framework documentation templates and guidelines, which are governed by the e-Framework Intellectual Property Rights Statement <http://www.e-framework.org/Default.aspx?tabid=738>

2 Service Expression Definition

2.1 **Classification**^{1,2,3}

To be provided by the submitter:				
Domain(s)	<input checked="" type="checkbox"/> Learning & Teaching	<input type="checkbox"/> Research Libraries	<input type="checkbox"/> Administration IT Services	<input type="checkbox"/> Common
Development Status	<input type="checkbox"/> Proposed	<input checked="" type="checkbox"/> Developmental	<input type="checkbox"/> Prototype	<input type="checkbox"/> Production
Maturity	<input checked="" type="checkbox"/> Immature	<input type="checkbox"/> Mature		
State Behaviour	<input type="checkbox"/> Stateful		<input checked="" type="checkbox"/> Stateless	
Transactional Behaviour	<input checked="" type="checkbox"/> Transactional and ACID	<input type="checkbox"/> Transactional but Non ACID	<input type="checkbox"/> Non-Transactional	
Batch Behaviour(s)	<input checked="" type="checkbox"/> Individual	<input type="checkbox"/> Batch		
Time-Constraint Behaviour	<input checked="" type="checkbox"/> Hard Real Time	<input type="checkbox"/> Soft Real Time	<input type="checkbox"/> None	
Service End Point	<input checked="" type="checkbox"/> Provider	<input type="checkbox"/> Requestor	<input type="checkbox"/> Transcoder (both requests and provides)	
Authentication/ Authorization Dependency	<input type="checkbox"/> Auth-Dependent		<input checked="" type="checkbox"/> Auth-Independent	
Protocol Binding(s)	<input type="checkbox"/> Web Service <input type="checkbox"/> SOAP	<input checked="" type="checkbox"/> REST <input type="checkbox"/> HTTP	<input type="checkbox"/> Other	
Service Genre Coverage	<input checked="" type="checkbox"/> Full	<input type="checkbox"/> Extended	<input type="checkbox"/> Subset	<input type="checkbox"/> Overlapping
Deployment Scale	<input type="checkbox"/> Isolated		<input checked="" type="checkbox"/> Ubiquitous	
To be determined by the e-Framework:				
Status	<input type="checkbox"/> Approved	<input type="checkbox"/> Placeholder <input type="checkbox"/> Unapproved	<input type="checkbox"/> Superseded <input type="checkbox"/> Withdrawn	
Confidence Level	<input type="checkbox"/> High	<input type="checkbox"/> Medium	<input type="checkbox"/> Low	

2.2 **Service Genre**

Appropriate Copy [To Be Written]

2.3 **Version**

- e-Framework Service Expression Version: v1.0

2.4 **Version History**

Version	Date	Author	Description	Organization / Project
V0.1	2007-06-04	Nick Nicholas	Initial Draft	FRED Project

¹ Classification categories shown in **Bold** are required.

² Optional classification category "Deployment Scale" can be deleted from the table if not used.

³ See definitions of the Service Expression Classification Scheme categories and their allowable choices at : <http://www.e-framework.org/Services/ServiceClassificationScheme/ClassificationSchemeForServiceExpression/tabid/816/Default.aspx>

--	--	--	--	--

2.5 Description

This service expression is a specification of the OpenURL protocol used with an HTTP transport. The OpenURL protocol is used to address the appropriate copy problem: a delivery system has access to several instances of an object, each stored in a way that affects the user experience of delivery (e.g. different physical location, different access regime). Given a request for the object (but not for any specific instance of the object), the delivery system must select the instance to deliver so that the user experience of delivery is optimal. Therefore OpenURL is used through this service expression to support a content delivery application, as the OpenURL standard defines it: OpenURL Applications transport packages "with the intent of obtaining context-sensitive services pertaining to the referenced resource" (OpenURL Standard, p. 1), and the context-sensitive service which this service expression supports is Obtain. This does not preclude other context-sensitive services, but the service expression must support context-sensitive Obtain at a minimum.

The service expression is a specialisation of the service genre *Appropriate Copy*, in which a set of parameters (including at a minimum attribute(s) of a requester and attribute(s) of a referent) are mapped to a retrieval key of a referent—that is, an encoding of the pair (Data Source, Label), where the Label may be used to retrieve the referent object from the Data Source through an Obtain service. The retrieval key returned by the service expression is encoded as a URL, and the service expression is intended expressly for orchestration with an Obtain service expression which takes the same retrieval key as its parameter.

The parameters of an OpenURL call specify the contextual entities involved in a request to obtain an object from a data source; they include but are not limited to attributes identifying the requested object. The call returns a URL to the instance of that object in the data source which is most appropriate for the specified context. The URL is then expected to be used by an Obtain expression, to access that instance of the requested object. The service which will be used to access the object can optionally be specified as a contextual entity.

Communications under this OpenURL service expression are through HTTP. There is no provision in this service expression for access control or authentication control of the resolution. This does not imply that there is no access or authentication control for the object instances that the service expression provides resolution to.

This service expression profiles OpenURL to the requirements of the FRED project. As a result, only certain OpenURL parameters are to be used. Moreover, those parameters need to be specified explicitly in OpenURL configuration files as outlined here, as they do not occur in the commonly available configurations (community profiles) of the protocol.

The service expression as described here does not fulfill all the requirements of an appropriate version service. Different service expressions cover other functional requirements with different parameter configurations (return all available retrieval keys in the federation; return the most appropriate version of the object, not just the most appropriate instance). The service expression is isolated from the other similar service expression for simplicity, and because they represent distinct service genres. We expect that implementations will realise more than one relevant service expressions.

2.6 Functionality

There is one request type in this service expression:

- Resolve to URL. The request takes parameters specifying the context of a request to Obtain: these parameters describe context entities. In this service expression, the context entities identified by the parameters are the *referent* (object to be obtained), the *requester* (end user requesting obtain), the *referrer* (the federation whose delivery system generates the request hyperlinks), and the *service* (the service used to obtain a particular presentation of the object).
 - The parameter specifying the *referent* is mandatory. The referent is specified through an identifier. The identifier must be unique, but needs not be global. The identifier is actioned through the OpenURL service expression, and needs not be resolvable independently of that service.
 - At least one parameter specifying a *requester* is required. The requester is either uniquely identified through an identifier, or else is partly specified through descriptive intrinsic attributes: an institutional affiliation or a geographical location.
 - The parameter specifying the *referrer* is mandatory. The referrer is specified through an identifier. The identifier must be unique. The referrer is specified in order to maximise interoperability with other OpenURL implementations which may operate on the same service call (identifying the service implementation as specific to the FRED community), and does not affect the functionality of the service expression within the FRED federation.
 - The parameter specifying the *service* is optional. The service is specified through descriptive intrinsic attributes: whether the presentation of the referent required is a presentation of metadata, a presentation of the object through the browser, or a packaging and downloading of the object. If the parameter specifying the service is absent, the service is assumed by default to be a presentation of metadata.
- Given a specification of the requester and referent, the service expression returns a URL. The URL is the locator of the instance of the referent in the repository federation which is most appropriate for a requester with the given attributes.
 - If no instance of the referent is appropriate for a requester with the given attributes, the service expression fails to resolve the request, and returns an error.

Communications to the service implementation are through HTTP. Service requests are made as an inline HTTP GET request. No other functionality is provided. The functionality defined SHALL NOT be extended.

2.7 Usage Scenarios [optional]

Usage Scenarios for an Appropriate Copy service are available separately: [<http://fred.usq.edu.au/appropriatecopyscenarios.html>] . This service expression only applies to a subset of the usage scenarios enumerated:

- Scenario C: Resolve given referent identifier and requester affiliation: Search
- Scenario D: Resolve given referent identifier and requester affiliation: Link from Object
- Scenario E: Resolve given referent identifier and requester affiliation: External Link
- Scenario H: Resolve given referent identifier and requester identity: DRM
- Scenario I: Resolve given referent identifier, requester affiliation, and requester physical location
- Scenario J: Resolve directly to object vs. to metadata

Of the remaining scenarios,

- Scenario A: Resolve given only referent identifier: Search (single instance) is addressed by *OpenURL + HTTP + Inline KEV Multiple Location Resolver*
- Scenario B: Resolve given only referent identifier: Search (multiple instances) is addressed by *OpenURL + HTTP + Inline KEV Multiple Location Resolver*
- Scenario F: Resolve given referent identifier and requester accessibility profile is addressed by an Appropriate Version service expression
- Scenario G: Resolve given referent identifier and requester accessibility profile: Choice of Manifestation is addressed by an Appropriate Version service expression

2.8 Applicability [optional]

This service expression assumes at least an information model in which digital objects that are identical in content and presentation are identified as belonging to the same Object Manifestation; the digital objects so related are considered to be Object Instances. The service expression specifies the object to be obtained (the referent), through referent attributes, at the Manifestation level. That is to say, the given referent attributes may match several objects stored in the federation, but those objects may only differ in their storage location, and not in their content or presentation. The end user must not be able to detect any difference in the object obtained, whichever selection of object instance the service expression makes. That way, the user requirement to interact with a given object manifestation is not affected by the functionality of this service expression.

Since the objects selected amongst must all be instances of the same Manifestation, the FRBR model can be further used to limit the scope of the service expression (although implementing the model does not presuppose adoption of the FRBR model). Objects selected through the service expression:

- Shall not differ in authorship, including authority/corporate ownership (the objects belong to the same FRBR Work)
- Shall not have major difference of content (the objects belong to the same FRBR Work)
- Shall not have minor difference of content (the objects belong to the same FRBR Expression)
- Shall not differ in file format (the objects belong to the same FRBR Manifestation)
- Shall not have major difference of content presentation (the objects belong to the same FRBR Manifestation)

As a result, the service expression is not applicable if the object is specified at the Object Instance level, as a specific instance at a specific location—i.e. if the end user specifies a retrieval key specific to a unique location to obtain the object. The service expression shall not be extended to allow object specification at the Object Instance level: A location-specific retrieval key is the result of a Resolve call, so invoking Resolve given a retrieval key becomes redundant.

The service expression is not applicable if the user specifies the object at the Object Manifestation level, and a unique location to obtain the object: the service expression maps these parameters to a unique retrieval key specific to the location, so an Object Instance still ends up being specified.

The service expression is applicable if the user specifies the object at the Object Manifestation level, and a range of locations from which to obtain the object (which may be a proper subset of the locations in the federation): the service expression selects the instance to be obtained from the range of locations proposed. However, there is no provision for specifying a range of candidate locations as parameters in the present service expression: this requires an extension to the proposed referent metadata profile.

The service expression is not applicable if the user specifies the object at the FRBR Expression level—that is, the object is specified in such a way that a candidate object may match the request regardless of its digital presentation. Any request to obtain an object is understood by this service expression to require a particular file format and encoding of the object. The service expression shall not be extended to select an appropriate manifestation as well as instance (copy) of an object specified at the FRBR Expression level (e.g. for accessibility purposes; cf. Scenario F). That functionality is instead covered by an Appropriate Version service expression.

The service expression is not applicable if the user specifies the object at the FRBR Work level—that is, the object is specified in such a way that a candidate object may match the request regardless of any discrepancies in its content. Any request to obtain an object is understood by this service expression to require a particular version of the object content. The service expression may be extended to select an appropriate expression and manifestation as well as instance (copy) of an object specified at the FRBR Expression level (e.g. customisations of content for particular audiences).

OpenURL allows intrinsic (meaningful) attributes of the referent to be used to identify the referent. Such attributes make OpenURL resemble Search. However, OpenURL is not applicable in a search context, i.e. when more than one referent differing at the FRBR Work level matches the referent attributes. The service expression shall not be extended to select between objects representing different FRBR Works. Moreover, the FRED profile of OpenURL requires that intrinsic attributes of the referent not be used in service calls. (Referents are identified only through semantically opaque identifiers.)

The service expression returns a unique locator, and is not applicable to contexts where the user is presented with a choice of locations. That functionality is instead covered by *OpenURL + HTTP + Inline KEV Multiple Location Resolver*.

2.9 Requests & Behaviours

The format for requests and responses shall follow the OpenURL v.1.0 specification. As noted under *Functionality*, the only descriptions of context entities supported by this service expression as parameters are those of referent and requester. The following parameters describing context entities shall be supported by this service expression:

- *rft_id*: a single identifier for the referent. The identifier shall be a Handle. The identifier shall be encoded for OpenURL calls using the `info:hdl` namespace, as determined by the OpenURL Standard.
- *req.id*: a single identifier for the requester. The identifier shall be an institutional email address. The identifier shall use the `mailto:` namespace.
- *req.affiliation*: the unique institutional affiliation for the requester. The affiliation shall be a string drawn from a limited vocabulary of institutions decided by the community.
- *req.location*: the unique location of the requester. The location shall be a string drawn from a limited vocabulary of locations decided by the community.
- *rfr_id*: the identifier of the referrer. The referrer shall be deemed to be the institution hosting the repository federation, which is responsible for generating OpenURL hyperlinks to access content within the federation.
- *svc.metadata*: should the service to be resolved to by the resolver access the referent metadata. Legal values are "yes" and "no".
- *svc.view*: should the service to be resolved to by the resolver display the object content through the browser. Legal values are "yes" and "no".

- *svc.download*: should the service to be resolved to by the resolver package the object for downloading. Legal values are "yes" and "no".

A single referent shall be identified through the referent descriptors (parameters). The Community Profile schema (see *Interface Definition*) does not allow the count of referent descriptors to be reduced to one; nonetheless only one referent descriptor shall be used for any call of the service expression.

A single requester shall be identified through the requester descriptors (parameters). The requester may be specified through one or more of affiliation and location, or through identifier. The Community Profile schema allows requester identifier to be combined with the other requester parameters.

A single referrer shall be identified through the referent descriptors (parameters). The referrer is given to ensure interoperability with OpenURL resolvers not specific to FRED, and does not affect functionality within the FRED federation. To quote the OpenURL implementation guidelines:

OpenURL Referrers are strongly encouraged to provide a genuine Referrer Identifier within the ContextObject as specified in Section 4.6. This will assist general OpenURL interoperability enabling the interpretation of local Identifiers, as well as Resolver functionality such the provision of usage statistics, and the prevention of circular links. Inclusion of this provenance within an OpenURL is an indication of its quality.

Only zero or one of the *svc* parameters shall have the value "yes"; that is to say, each service type parameter specifies a distinct service through a boolean value, and at most one service type shall be specified by the *svc* context object. If no service type parameter is given, the metadata service (i.e. view metadata as a front page presentation of the object, which is commonplace in repositories) shall be the default service invoked in the resolution. The service type context object specifies service at the service expression level (hence "service type"): there is an expected behaviour and output format distinct for each service specified, but particular implementations and instances of services are not specified.

The service expression may be extended to support other parameters and other context entities, within the constraints put forward elsewhere in the service expression.

- For example, the San Antonio profile [KEV Metadata Format for Service Types for the Scholarly Community](http://www.openurl.info/registry/docs/mtx/info:ofi/fmt:kev:mtx:sch_svc) (http://www.openurl.info/registry/docs/mtx/info:ofi/fmt:kev:mtx:sch_svc) allows the distinct services of: Abstract; Citation; Fulltext; Holdings; ILL (= request Interlibrary Loan); or Any. A FRED service expression might be extended to add an Abstract service type. The service context object is represented by FRED in a way consistent with this metadata format.
- The San Antonio profile, and the Dublin Core profile [Dublin Core Community Profile \(DCCP\) for Simple Dublin Core in KEV Format](http://alcme.oclc.org/openurl/servlet/OAIHandler?verb=GetRecord&metadataPrefix=oai_dc&identifier=info:ofi/pro:dccp) (http://alcme.oclc.org/openurl/servlet/OAIHandler?verb=GetRecord&metadataPrefix=oai_dc&identifier=info:ofi/pro:dccp), allow intrinsic (meaningful) descriptors to be used to identify referents. Meaningful referent descriptors from either profile (or from a novel profile), such as author and title, may be used instead of the referent identifier proposed here. This allows access to OpenURL services from different communities using those profiles. However such extension shall be deemed to constitute a distinct service expression.

The service expression response is either a single URL or an error. The URL is that of one of the instances of the object matching the referent descriptor in the system. In a repository federation, these are the URLs of all Object Instances of the Object Manifestation stored on a repository in the federation; the ContextObjects are used to select among those instances. In FRED, the URLs are

made available to the OpenURL service expression by an Instances Registry, which maintains the listing of all object instances in the federation, their locators, and the Object Manifestations (= files) they correspond to. As discussed under *Design Decisions & Tradeoffs*, an Instances Registry may be implemented as a combination of ruleset (selecting repository) and lookup table (containing the instances for each repository).

A single URL is returned if there is sufficient information in the ContextObject to decide on the most appropriate instance to deliver to the end user, given the context. So long as there is any requester descriptor present, the service expression is expected to return a single URL. If multiple URLs are equally appropriate for the requester, the service may use other criteria, unrelated to the requester, to select an instance (e.g. load balancing). An error is returned if none of the available instances is appropriate for the user.

2.10 Use & Interactions

This service expression is assumed to be used by an end user in conjunction with an Obtain delivery mechanism: the URL returned through OpenURL (as a resolution of its input parameters) then invokes a service to obtain the object instance so identified. This service expression is orchestrated specifically with the service expression *Obtain HTTP/URL/Browser*: the input is a URL activated in a web browser, and the result is an object delivered across HTTP and rendered through the browser. In order to make the integration of OpenURL and *Obtain HTTP/URL/Browser* seamless, the call to OpenURL is also presented as a URL activated in a web browser. That way, the user experience is one of *Obtain HTTP/URL/Browser* alone (since Obtain, not Resolve, is the end user requirement); the OpenURL resolution to the required URL of the appropriate copy is hidden from the user, and treated by the browser as an HTTP redirect.

Any content delivery systems using the OpenURL service expression to deal with appropriate copy delivery should substitute all hyperlinks they would provide to federation content (i.e. all service calls to *Obtain HTTP/URL/Browser*) with OpenURL hyperlinks (i.e. service calls to OpenURL + *Obtain HTTP/URL/Browser*). These hyperlinks may be presented in response to requests for discovery; or in rendering either content objects or object metadata containing references to other objects. Under FRED, all content objects are identified through identifiers (Handles); the OpenURL hyperlinks therefore provide access to content objects based on those identifiers.

The service expression may be extended as necessary in order to be used in conjunction with other context-sensitive services, except where such extension is prohibited in this document.

The service expression is expected to be realised in tandem with an implementation of *OpenURL + HTTP + Inline KEV Multiple Location Resolver* or of an Appropriate Version service expression. It is expected that the one implementation will implement more than one service expressions, and will choose which expression's behaviour to execute based on the parameters passed to the service implementation.

2.11 Structure [optional]

This service expression presupposes the data model for OpenURL defined in the OpenURL v.1.0 standard:

- Any request for a service involves six *context entities*.
- These context entities provide the context for the service call.
- The descriptions of these context entities are aggregated into a single *ContextObject*.

- Therefore specifying the ContextObject specifies the context for the service call.
- Context entities may be described through attributes (given in-line or by reference to an external file), or specified through identifiers. These attributes and identifiers are *Descriptors*, and constitute the parameters of an OpenURL service call.
- The context entities constituting a ContextObject are:
 - *Referent*: an object to be operated on by the service.
 - *Service Type*: the service to be applied, specified at the genre or expression level.
 - *Requester*: the end user requesting the service.
 - *Resolver*: the resource processing the service request (i.e. the resource hosting the OpenURL service expression).
 - *Referring Entity*: the resource containing the reference to the referent which triggers the service call (for this service expression, the resource containing the hyperlink which triggers the service expression call).
 - *Referrer*: the resource generating the call to the OpenURL service expression (typically for this service expression, the federation as an institutional entity).

This service expression presupposes a federation of repositories hosting content, with each participating repository a candidate for the Data Source (Location) identified by the service as hosting the most appropriate copy—i.e. the Data Source in the retrieval key pair (Data Source, Label). The service expression presupposes a further data source: an Instances Registry, containing all the instances of objects in the federation.

2.12 Interface Definition [recommended]

The service expression shall conform to v.1.0 of the OpenURL standard. The service expression shall use the KEV ContextObject Format; this includes the KEV ContextObject serialisation (key-value pairs, i.e. service call expressed as an HTTP GET query), and the Z39.88-2004 constraint language (native to OpenURL v.1.0). The service expression shall use the UTF-8 character encoding. The service expression shall use the Inline OpenURL HTTP transport. Alternate formats and transports are allowed (e.g. XML/SOAP), but a choice of different format or transport shall result in a new service expression.

These choices listed above an OpenURL Community Profile. As required by OpenURL, the FRED Community Profile is described in machine readable form at <http://fred.usq.edu.au/files/fredprofile.xml>, following the [Community Profile Schema](#) (<http://alcme.oclc.org/openurl/servlet/OAIHandler/extension?verb=GetMetadata&metadataPrefix=xsd&identifier=info:ofi/fmt:xml:xsd:pro-2004>). This profile is to be submitted for registration.

As required by the Inline OpenURL transport, all descriptors (parameter values) shall be URL-encoded.

As required by the Inline OpenURL transport, all calls to the service expression shall include the parameter `url_ver=Z39.88-2004`, giving the version of OpenURL. All calls shall include the parameter `url_ctx_fmt=info%3Aofi%2Ffmt%3Akev%3Amtx%3Actx`, nominating the KEV ContextObject format. Calls shall not include the `url_tim` parameter, giving the timestamp of the OpenURL creation, as there is no compelling business need from FRED to do so.

The service expression shall not use as parameters administrative data keys describing the ContextObject. (Of these, `ctx_ver` is identical to `url_ver`; `ctx_tim`

is identical to `url_tim`; the identifier of the OpenURL ContextObject `ctx_id` is not applicable to the expected use of OpenURLs as dynamic hyperlinks; the character encoding `ctx_id` is implicit in the FRED Community Profile.)

The requester parameters constitute a requester metadata profile. The profile is defined through the OpenURL Constraint language in the file http://fred.usq.edu.au/files/fred_req_mtx.html. The file location shall be nominated as the `req_val_fmt` parameter for all service calls to this service expression.

The service type parameters constitute a service type metadata profile. The profile is defined through the OpenURL Constraint language in the file http://fred.usq.edu.au/files/fred_svc_mtx.html. The file location shall be nominated as the `svc_val_fmt` parameter for all service calls to this service expression. The FRED service type metadata profile shares its format with the San Antonio community profile [KEV Metadata Format for Service Types for the Scholarly Community](#) (http://www.openurl.info/registry/docs/mtx/info:ofi/fmt:kev:mtx:sch_svc), but does not have any services in common with it.

The metadata profiles have not been registered with the OpenURL registry as of this writing. Until they are, the file location shall be identified in the `req_val_fmt` and `svc_val_fmt` parameter through a URL. The URL shall be a Handle resolution call to resolve the Handle YYYY; e.g. `req_val_fmt=http://hdl.handle.net/YYYY` **[to be supplied]**. The file shall not be identified through a Handle by itself: the OpenURL standard bars "info" URIs from use in identifying unregistered profiles.

As recommended by OpenURL, the referrer identifier `rfr_id` shall have a value of the format `info:sid/DNS` or `info:sid/DNS:COLLECTION`. In accordance with the [info:sid specification](#) (<http://alcme.oclc.org/openurl/docs/pdf/info-sid.pdf>), `DNS` shall be the DNS-resolvable internet domain or subdomain associated with the referrer institution; `COLLECTION` is an optional description of the federation. (e.g. `info:sid/thelearningfederation.gov.au`, `info:sid/www.learningplace.com.au:qld-school-repository-federation` .) The identifier shall be URL-encoded when it appears as an OpenURL parameter.

Service Implementations must support HTTP GET OpenURL hyperlinks of at least 2048 bytes length.

Foreign keys are service call parameters of a form not defined by the standard, i.e. not prefixed by a prefix matching the regular expression: `m/url_|ctx_|(rft|rfe|req|rfr|res|svc)[_./]`. Foreign keys shall be ignored by service implementations.

2.13 Applicable Standards

- American National Standards Institute 2005. *The OpenURL Framework for Context-Sensitive Services*. Bethesda (Md.): NISO Press. ANSI/NISO Z39.88-2004. ISSN: 1041-5653. http://www.niso.org/standards/standard_detail.cfm?std_id=783
- International Federation of Library Associations and Institutions. 1998. *Functional Requirements for Bibliographical Records*. Munich: K.G. Saur. ISBN 3-598-11382-X. <http://www.ifla.org/VII/s13/frbr/frbr.pdf>
- info:sid specification. <http://alcme.oclc.org/openurl/docs/pdf/info-sid.pdf>

2.14 Design Decisions & Tradeoffs [optional]

2.14.1 Institutional Service Component vs. Centralised Provision

Difference in affiliation of the requester can be specified in several ways in an OpenURL. The traditional way in the library community has been to use a

different OpenURL link server address for each institution. So a Cairns High School-appropriate copy of `hdl:102.100/378` is retrieved through a call like http://cairnshs.qld.gov.au/openurl?rft_id=hdl%3A102.100%2F378 , and a Tweed Heads-appropriate copy through http://tweedheadshs.nsw.gov.au/openurl?rft_id=hdl%3A102.100%2F378 . This solution is appropriate if:

- Each institution has its own repository
- Each institution is able to deploy and populate its own OpenURL service
- There is no business need to centralise or coordinate the deployment of OpenURL across institutions

Since the link server is the value which gets populated differently for each institution, it is described as the *Institutional Service Component*; the ContextObject, being the parameters of the service call, do not change between instances of the link.

Under FRED, institutions do not necessarily have their own repository or OpenURL services, and there is incentive to coordinate the distribution of resources across the federation. A centralised OpenURL service is the preferred option; the service is accessed through links embedded in content rendered at the registry, as well as or instead of the source repositories. Given a centralised OpenURL service, the link server is no longer the Institutional Service Component: institutional affiliation needs to be included among the service parameters instead, explicitly or implicitly (via the user identifier). Scenario B illustrates an explicit affiliation parameter. Scenario A illustrates an institution-specific OpenURL server, where `openurl.example.com` would be replaced by the institution's particular server.

2.14.2 Restriction of Selection to Locations

In order to expedite the process of selecting an instance, the process of selection can be restricted to the source repository of the appropriate copy, rather than a direct selection of instances. The Instances Registry may accordingly be implemented as two further data sources:

- A rule set, according to which the input requester parameters are used to select the most appropriate Location.
- A lookup table, mapping the shared identifier for the object, given as the referent descriptor, to the object Label which is local to the selected Data Source.

In this service expression, the retrieval key is encoded as a URL: the Data Source is the repository hostname (base URL), and the Label the URL suffix. The service expression composes the locator for the most appropriate copy from these two components. The suffix of the URL may be determined one of three ways:

- If the Label is parameterised on the shared identifier of the object, then a URL suffix template is determined by the ruleset, and the identifier is inserted into the suffix template, providing the actual URL suffix to append to the base URL. e.g. `http://www.example.com?id=hdl:/721.3829`.
- If the Label is parameterised on a local identifier, then a URL suffix template is determined by the ruleset. The shared identifier is mapped to the local identifier through the lookup table, which has been populated by the local repository. The identifier is inserted into the suffix template, providing the actual URL suffix to append to the base URL. This is the most common expected outcome.
 - For instance, the referent is specified in the request by the Handle `hdl:/721.3829` . The retrieval label for a Fedora repository involves a local identifier (PID). The ruleset determines that the most appropriate repository is <http://www.example.com>, and the URL suffix template for that repository is of the form `?id=___`. The

lookup table specific to `www.example.com` maps the Handle `hdl:/721.3829` to the `www.example.com` PID 512 . That PID value is inserted into the suffix template, giving the URL suffix `?id=542`. The URL suffix is appended to the base URL, giving the appropriate copy locator `http://www.example.com?id=542`.

- If the suffix of the URL is not parameterised but is idiosyncratic for each resource, then the entire URL suffix for the global identifier is looked up in through a lookup table supplied by the local repository holder. The URL suffix is appended to the base URL.

2.14.3 Access-Appropriateness

Under Scenario H, the OpenURL service needs to anticipate whether the user will gain access to the object through the candidate hyperlink. If a user with the given user profile cannot obtain the object because of access restrictions, then the hyperlink will not yield an appropriate copy for the user. In principle, this means the OpenURL service needs access to the user database of all target data sources, and the data permissions of all Object Instances on all target data sources. As a practical constraint, it is sufficient for the OpenURL service implementation to anticipate access restriction only as a function of requester's institutional affiliation mapping to data source. This map can be incorporated readily into the ruleset determining the data source to connect to. Finer granularity of requester and of referent (i.e. permissions specific to individual users, or to individual objects) can be policed at the target repository: the OpenURL service can use affiliation to narrow down the number of data sources which the requester can usefully be directed to, without fully anticipating all access restrictions possible.

This is consistent with the design outlined above: the requester determines the choice of most appropriate Data Source, and a unique object in the data source is assumed to match the requested referent. There is no provision in this model for selection of the most appropriate Object Instance *independent* of Data Source. So business logic, as a ruleset, determines the choice of data source given requester context; but requester context does not determine the choice of label, which is left to a simple lookup table. In the same way, access appropriateness is determined for the Data Source by the service implementation; the choice of label is not informed in the same way.

2.15 Implementation Guidance & Dependencies [recommended]

Implementation guidelines for OpenURL applications using the KEV ContextObject format are available from http://alcme.oclc.org/openurl/docs/implementation_guidelines/index.html .

Given the lack of security in the HTTP transport, the Requester profile should not be used to transport authentication data and thus provide an authenticated version of the service expression.

By-reference parameters (for which the values of a ContextEntity descriptor are retrieved from an external on-line file, instead of being passed into the service request as in-line parameters) are a security risk, as discussed in the Implementation Guidelines, App. C.1. By-reference parameters are not used in this service expression.

As recommended in the Implementation Guidelines, App. C.2, service implementations shall not fetch metadata from a network location specified in an OpenURL with an invalid version string.

2.16 Known Uses [optional]

Actual: None

Potential: The service expression may be used in a repository federation wherever a hyperlink would be used to provide access to an object stored in the federation. The hyperlink may be generated anywhere inside or outside the federation, including:

- links to objects from discovery interfaces generated within the federation;
- links to objects from other objects within the federation (both from content and metadata datastreams);
- links to objects in the federation from objects outside the federation;
- links to particular views of an object from anywhere inside or outside a federation, by invoking specific services.

2.17 Service Expression Dependencies [recommended]

<type text here>

2.18 Related Service Expressions [optional]

- Obtain HTTP/URL/Browser

2.19 Related Service Usage Models (SUMs) [optional]

- Manage Resolver SUM: service usage model describing application through which the possible mappings of ContextObjects to URLs carried out through this service expression are populated, updated, maintained. Includes initial population, and orchestration of updating OpenURL Data Source with updating location of object accessed through OpenURL.

2.20 Related CoRe SUMs [optional]

<type text here>