Extended-Linking Services: towards a Quality Web[1]

Eric F. Van de Velde

California Institute of Technology

A URL takes requesters from a citation to a destination… provided, of course, the URL is still valid. The current chaotic web is wonderful in its way. However, within this chaotic web, we believe there is a need for a high-quality web of vetted information. The emerging OpenURL standard is the cornerstone of a worldwide web with high-quality links that feature properties such as:

- Persistence: Increase the probable lifetime of citations.
- Multiplicity: Produce a menu of targeted services for each citation.
- Context-Sensitivity: Resolve a citation in a manner appropriate to the user and to the context.

To encourage the development of extended-linking services, NISO formed a committee to develop a standard OpenURL syntax. Our immediate goal is to serve the scholarly-information community immediately. However, the OpenURL technique is widely applicable, and we expect to serve many other information communities.

Historical Perspective

The ideas behind OpenURL were developed at the University of Ghent by Van de Sompel and Hochstenbach [Van de Sompel and Hochstenbach 1999a, 1999b, 1999c]. These ideas received wide exposure at the Santa Fe meeting, which was organized by Paul Ginsparg, Rick Luce, and Herbert Van de Sompel in October 1999. It is no exaggeration to call this meeting historic. Software developers and technology managers present at this meeting committed to a far-reaching collaboration on digital-library initiatives. This meeting established the Open Archives initiative and contributed to the meteoric rise of OpenURL.

What is an OpenURL?

A bibliographic citation describes a **referent**, which may be a journal article, a book, a technical report, or some other work. When the citation to the referent occurs in a database or in an electronic document, it makes sense to embed the URL of the referent so that the referent is only a mouse click away from its citation. In OpenURL terminology, the person who clicks on that link is called the **requester**.

Conventional links have some disadvantages, however. For example:

- The URL becomes effectively useless when the location of the referent changes.
- Not all referents have URLs. For example, non-digital works do not have a URL.

[1] This is an adaptation of [Van de Velde and Coles 2002].

- Some referents are associated with more than one URL, only one of which may be appropriate for the requester. For example, if the referent is part of a licensed resource, requesters need the URL of that version to which they have access.
- Requesters may be interested in more than just the referent: they may want services related to the referent. For example, they may want to check whether a book is available in their institute libraries.

In the OpenURL approach, an indirect link replaces the direct link from citation to referent. The appropriately formatted citation is transported to a **resolver,** which transforms the citation into one or more URLs and/or into a menu of services. Because the resolver performs this transformation only when the requester clicks on the link, it is able to use current context-sensitive information such as identity of the requester, current URLs of referents, and information produced since the document was written. To implement this basic idea, we must

1. Cast the citation into a **format** that can be parsed by an automated service. This machine-readable format organizes the **metadata** obtained from the citation and from the context in which this citation occurs.
2. **Transport** the metadata via the web to the resolver.
3. Build a resolver that transforms metadata into services and/or URLs.

An OpenURL is a web-transportable metadata format. It is only concerned with steps 1 and 2 of the above process. While OpenURL is an enabling technology for linking services to citations, it is not concerned with the nature of these services or with the methods by which the metadata contained in the OpenURL are transformed into services. That belongs in the realm of resolvers. Whereas resolvers can be proprietary and closed systems, the OpenURL format will be an open standard.

Draft guidelines for constructing OpenURLs are already freely available [Van de Sompel, Hochstenbach, and Beit-Arie 2000], and a formal standardization process has started under the aegis of NISO [NISO Committee AX].

The number of available OpenURL resolvers is growing rapidly. Currently, they include:

- SFX [Ex Libris (USA), Inc. SFX] was the first OpenURL resolver. In fact, the SFX resolver predates OpenURL. Van de Sompel and Hochstenbach developed the SFX resolver and the OpenURL concepts as part of their research on context-sensitive linking [Van de Sompel and Hochstenbach 1999a, 1999b, and 1999c].
- 1Cate, jake.openly.com, and link.openly.com [Openly Informatics, Inc.]
- LinkFinderPlus [Endeavor Information Systems, Inc.]
- Open Linking Technology [Fretwell-Downing, Inc.]
- Powell's OpenResolver [Powell] is available under the GNU open-source license.
- KDB Linking System [KINS, Inc.]

Demonstration and Technical Details

Consider the citation displayed in Example 1. Using the draft OpenURL specifications [Van de Sompel, Hochstenbach, and Beit-Arie 2000], the OpenURL version of this citation could take the form displayed in Example 2.

Van de Sompel, Herbert and Beit-Arie, Oren. 2001. Open Linking in the Scholarly Information Environment Using the OpenURL Framework. D-Lib Magazine. 7(3).

Example 1: A typical conventional citation to a journal article

http://sfx.caltech.edu:8088/caltech?genre=article&atitle=Open%20Linking%20in%20the%20Scholarly%20Information%20Environment%20Using%20the%20OpenURL%20Framework&title=D-Lib%20Magazine&issn=1082-9873&date=2001-03&volume=7&issue=3&aulast=Van%20de%20Sompel

Example 2: The citation of Example 1 formatted as an HTTP-encoded OpenURL

Figure 1: List of services produced by Caltech's SFX resolver with the metadata of Examples 1 or 2 as input

From Example 2, we see that an OpenURL may take the form of a familiar HTTP GET request. The part before the question mark is the URL of Caltech's SFX resolver [Ex Libris (USA), Inc. SFX]. The part following the question mark is metadata describing the referent (the citation in machine-readable form). When the requester clicks on the above link, the browser jumps to the URL of the resolver, and the metadata is transported to the resolver. What the resolver does with this information is not standardized. In our example, the resolver produces a list of services appropriate to this particular citation. Figure 1 displays the list of services produced by the Caltech SFX resolver.

The mechanism as explained thus far is inadequate. For example:

- Non-Caltech requesters are referred to Caltech resources, such as the document-delivery system Ibid or the catalog of the Caltech Library System.
- Documents or databases containing links like those in Example 2 must be updated every time the URL of the resolver changes.

The resolver must be determined at the time when the requester clicks on the link. This can be achieved in several ways. Unfortunately, elegant solutions require web-browser modification, and we cannot wait for that to happen. For now, we must settle for pragmatic approaches, each of which has some drawbacks. For example, the URL of the resolver could be stored in a user profile. This works well for systems that require users to log in. In other cases, one may have to resort (somewhat reluctantly) to web-browser cookies. This is not the proper forum to examine all possible approaches to resolver selection. However, it is instructive to examine the Cookie Pusher mechanism, which was first proposed by Van de Sompel and Hochstenbach [Van de Sompel and Hochstenbach 2000].

Before they can use the resolver, requesters must browse to a particular web page in order to set a cookie that contains the URL of the resolver. This one visit activates their access to the resolver until the cookie is deleted. If this cookie is not set, the data provider assumes the requester does not have access to an OpenURL resolver and either does not provide resolver functionality or (if available) uses a free resolver that may be used by anyone.

For simplicity, we assume that the cookie has been set. Since we have no prior knowledge of the URL of the resolver, it is impossible to embed an HTTP link like the one of Example 2 in our database web page or electronic document. Instead, we have to retrieve the cookie, construct the HTTP request, and process the HTTP request. In an HTML-formatted document, this activity is "hidden" behind a button placed next to the citation as in Example 3.

Van de Sompel, Herbert and Beit-Arie, Oren. 2001. Open Linking in the Scholarly Information Environment Using the OpenURL Framework. D-Lib Magazine. 7(3). S·F·X

Example 3: An "OpenURL Aware" citation to a journal article

Typically, a program written in a browser-compatible language such as Java or JavaScript performs all of the actions required: read the user's cookie, determine the

appropriate resolver, and define an SFXButton function that can be invoked in an HTML page. Ex Libris (USA), Inc. [Ex Libris (USA), Inc. Script] provides such a script. If the HTML document includes this JavaScript, then the OpenURL-Aware citation of Example 3 can be encoded in HTML as shown in Example 4.

```
<P>Van de Sompel, Herbert and Beit-Arie, Oren. 2001. Open Linking in the
Scholarly Information Environment Using the OpenURL Framework. D-Lib
Magazine. 7(3).
<SCRIPT TYPE="text/javascript" >
SFXButton(
       "genre=article&
       atitle=Open%20Linking%20in%20the%20Scholarly%20Information%20Envir
onment%20Using%20the%20OpenURL%20Framework&
       title=D-Lib%20Magazine&issn=1082-9873&
       date=2001&
       volume=7&
       issue=3&
       aulast=Van%20de%20Sompel&
       auinit=H")
</SCRIPT></P>
```

Example 4: HTML Representation of an OpenURL-Aware citation.

OpenURL Standardization

In February 2001, NISO formed NISO Committee AX and started the OpenURL standardization process. At the time of writing this, no NISO membership votes had been taken. What follows is an outline of the status of discussions within NISO Committee AX around June 2002 and is not endorsed by NISO or the NISO membership. For recent updates on the OpenURL standardization process, please check the NISO web site [NISO] or the NISO Committee AX web site [NISO Committee AX].

The Committee adopted both a short-term and a long-term approach. In the short time, it wanted to encourage early adoption of OpenURL by assuring reasonable stability to early adopters. In addition, the committee recognized that the OpenURL guidelines [Van de Sompel, Hochstenbach, and Beit-Arie 2000] are a great success both in number of early adopters and the quality of the applications. The committee, therefore, recommended this draft without amendments or modifications as Version 0.1 of the OpenURL standard. OpenURLs without a version number will be interpreted according to these draft specifications. This should assure early adopters that the standardization process would not undermine their current efforts.

In the long term, only an evolving OpenURL standard can be successful: it must continually adapt to new technologies. It is easy to be caught up in the minutiae of encoding issues. However, encoding is intimately tied to current technology and is, therefore, not the proper foundation for a long-term evolutionary process. For Version 1.0, the committee intends to put in place the theoretical and fundamental concepts that are independent of technology.

At the core, the fundamental issue is which metadata of which possible entities need to be described. In turn, this depends on what an OpenURL is supposed to be. The initial discussions led to the following definition of an OpenURL:

An OpenURL is a transportation mechanism for metadata that describe

- *one or more referents and*
- *zero or more other entities that define the context in which the reference to the referents occurs or in which the transportation of metadata takes place.*

This framed the discussion and led to the following result:

In an OpenURL, we must be able to describe the following entities:

- *Referent*
- *Resolver*
- *Requester*
- *Referrer*
- *Referring-entity*
- *Service-type*

Each of these entities can potentially be described in several different ways. The fundamental metadata-description mechanisms (or descriptors) include:

- *Id*
- *Metadata-description or by value*
- *Metadata-description-pointer or by reference*
- *Private-zone*

Not all descriptors apply to all entities. For example, the resolver must be described by means of the id descriptor. That would be overly restrictive for the referent, however, which may be described by any of the four available descriptors. For details, please consult NISO committee AX documents [NISO Committee AX].

Conclusion

OpenURL is the beginning of an evolution that will increase the power of web links. With OpenURL, web links

- Are context sensitive.
- Deliver narrowly targeted and appropriate services.
- Have a longer useful life.
- Provide connections to services and information that did not yet exist at the time of writing the documents.

Right now, OpenURL improves the functionality of bibliographic citations in various databases and electronic documents. In the future, OpenURLs may describe not only bibliographic citations, but also subject headings, chemical formulas, genomes, products, patents, etc. Once the latter can be described, it will be possible to provide extended linking services for them.

Related Web Sites

Endeavor Information Systems, Inc. LinkFinderPlus.
http://www.endinfosys.com/prods/lfwhatis.htm

Ex Libris (USA), Inc. SFX. http://www.sfxit.com/

Ex Libris (USA), Inc. OpenURL JavaScript.
http://demo.exlibrisgroup.com:8888/OpenURL/javascript.html

Fretwell-Downing, Inc. Open Linking Technology.
http://www.fdusa.com/products/olt.html

NISO. The Web site of NISO. http://www.niso.org

NISO Committee AX. The web site of NISO Committee AX on OpenURL Standardization. http://library.caltech.edu/openurl

KINS, Inc. KDB Linking System.
http://www.kdblink.com/

Openly Informatics, Inc. 1Cate, jake.openly.com, and link.openly.com.
http://www.openly.com

Van de Sompel, Herbert; Hochstenbach, Patrick and Beit-Arie, Oren. May 2000. OpenURL syntax description. http://www.sfxit.com/openurl/openurl.html or http://library.caltech.edu/openurl/Documents/OpenURL_Version_0.1.mht

Van de Sompel, Herbert and Hochstenbach, Patrick. 2000. Cookiepusher document.
http://www.sfxit.com/openurl/cookiepusher.html

Bibliography

Powell, Andy. OpenResolver: A simple OpenURL Resolver. Ariadne. 22-June-2001. Issue 28. http://www.ariadne.ac.uk/issue28/resolver/intro.html

Beit-Arie, Oren et al. 2001. Linking to the Appropriate Copy. D-Lib Magazine. 7(9).
http://www.dlib.org/dlib/september01/caplan/09caplan.html

Van de Sompel, Herbert and Hochstenbach, Patrick. 1999a. Reference Linking in a Hybrid Library Environment. Part 1: Frameworks for Linking. D-Lib Magazine. 5(4).
http://www.dlib.org/dlib/april99/van_de_sompel/04van_de_sompel-pt1.html

Van de Sompel, Herbert and Hochstenbach, Patrick. 1999b. Reference Linking in a Hybrid Library Environment. Part 2:SFX, a Generic Linking Solution. D-Lib Magazine. 5(4). http://www.dlib.org/dlib/april99/van_de_sompel/04van_de_sompel-pt2.html

Van de Sompel, Herbert and Hochstenbach, Patrick. 1999c. Reference Linking in a Hybrid Library Environment. Part 3: Generalizing the SFX solution in the "SFX@Ghent & SFX@LANL" experiment. D-Lib Magazine. 5(10). http://www.dlib.org/dlib/october99/van_de_sompel/10van_de_sompel.html

Van de Sompel, Herbert and Beit-Arie, Oren. 2001. Generalizing the OpenURL Framework beyond References to Scholarly Works, The Bison-Futé Model. D-Lib Magazine. 7(7). http://www.dlib.org/dlib/july01/vandesompel/07vandesompel.html

Van de Velde, Eric F. and Coles, Betsy. OpenURL-Aware ETDs. Caltech Library System Papers and Publications. January 2002. http://resolver.library.caltech.edu/caltechLIB:2002.002

Walker, Jenny. 2001. Open linking for libraries: the OpenURL framework. New Library World. 102(1163/1164). pp. 127-133. http://alidoro.emeraldinsight.com/vl=53339837/cl=20/nw=1/rpsv/~1181/v102n4/s1/p127