Date 2nd version : 18/07/2006

A Look at New Information Retrieval Protocols: SRU, OpenSearch/A9, CQL, and XQuery

Sally H. McCallum (1)
Library of Congress, USA

Meeting:	**102 IFLA-CDNL Alliance for Bibliographic Standards ICABS**
Simultaneous Interpretation:	**Yes**

WORLD LIBRARY AND INFORMATION CONGRESS: 72ND IFLA GENERAL CONFERENCE AND COUNCIL
20-24 August 2006, Seoul, Korea
http://www.ifla.org/IV/ifla72/index.htm

Abstract

Libraries have a large stake in search protocols because library systems are diverse yet library users need to access multiple sites without learning the search syntax of each site. This paper reviews and compares the relative advantages of several of the newest search protocols and query languages: Search via URL (SRU), OpenSearch, Contextual Query Language (CQL), and XQuery. The models for SRU and OpenSearch operations are described in order to explain differences in functionality - keyword search and simple data record return for OpenSearch and richer search with multiple format data return for SRU. The advantages of CQL are described along with possible complementary uses of the highly detailed and complex XQuery being developed for XML.

Introduction

Information retrieval, while seemingly simple and obvious when accessing Google, is surprisingly complex. Up until computing was developed half way through the last century, retrieving documents was highly dependent on metadata - then called cataloging - that could be installed on devices such as card or book catalogs and provided to the searcher to peruse. This "technology" resulted in the development and refinement of a number of conventions to enhance

retrieval - consistent sets of metadata to be recorded, rules for formulation of access points, lists of names, cross reference conventions to assure collocation of like names, extensive development of general and specialized subject vocabularies, and development of strings of subject terms with meaning inherent in the order. All of these components provided highly controlled metadata that would collocate in files that could be searched and that often did well on tests for recall and relevancy.

With the advent of computers, information specialists began to use the metadata in new ways for enhanced user retrieval, exploiting the consistency of the metadata. By the 1980s, libraries and library systems vendors had been quite creative in using the standard metadata in the MARC record to offer to users a variety of searching methods that enhanced retrieval - and their products. Since these systems had different architectures, in the late 1980s libraries developed a standard search protocol that enabled searching across them, with widespread implementation coming in the 1990s. This was possible because of the extensive standardization of the metadata in the community so the protocol could focus on other differences that were the reality of this historic, pre-web effort - different basic architectures and different approaches to design. Today that protocol, Z39.50[(2)], is an essential component of library catalogs. For example the web interface of the Library of Congress Bibliographic catalog currently receives an average of approximately 150,000 searches each work day and more than half of them come via the Z39.50 protocol.

The development and extensive use of Z39.50 has highlighted the fact that indexing by different systems is a critical element in searching remote catalogs and the mismatch of indexing at different targets can be a significant confusion for end users. At different times efforts have been made to "standardize indexes", but there has never been enthusiasm from systems vendors, rather there has sometimes been hostility. Whereas standardizing the metadata itself enables metadata sharing, it can be argued that the freedom to index the rich MARC metadata has been beneficial to libraries as it encourages system developers to experiment with different approaches to searching.

If we scroll forward 20 years, to 2006, the technology offers many more options for cross-platform retrieval. The web was invented in the 1990s and the development of XML provided an opportunity to take the understanding of searching built through the development and use of Z39.50 to this new platform. Since 2000 a number of opportunities and approaches have appeared that take advantage of the new tools, new standards, and the plethora of digital full text material available. Thus the components of retrieval - metadata, controlled vocabulary, collocation, free text keyword, indexing, recall and relevance - are being discussed in a new light.

Search is a broad topic and this paper will focus on some current models and protocols that are being propagated for cross system searching. In particular it examines two search protocols, Search/Retrieve via URL (SRU) and OpenSearch/A9, and two query languages, Contextual Query Language (CQL) and Xquery. Metasearch or multi-site search is not treated per se as most client/server protocols can be configured to provide the end user with metasearch results -

albeit with more and less sophistication and transparency.

Terminology used. This paper differentiates a *query language* from a *search protocol* in the following way. A search protocol is a series of messages between a client (called *user* in this paper) and a server (called a *target* in the paper). User in this context is not really the end user but the end user=s system. One component of those messages is the actual query itself, but the messages also provide the context for the query, information on user and target preferences, information about and handling of the result set that is the result of the query, and the vehicle to convey the records retrieved (the *retrieval* component). *Protocol* will generally be used for the management of the messaging and the retrieval and *query* for the argument component that expresses the search criteria. The *query language* is the syntax that specifies in a structured form the parameters for a query, such as "search for the specific word 'fish' that appears in titles or subject terms". An important distinction will be made between a query in the local and/or user syntax and one using an abstract or standard syntax definition.

SRU - Search/Retrieve via URL

SRU is a search and retrieval protocol that uses Internet and web facilities to carry the messages between user and target. SRU has an important antecedent, Z39.50, which is very widely implemented globally. Much of the functionality of SRU is derived from the older protocol, however, only the most useful was brought over, and in a simplified form. At the time that SRU development began, similar search use of the URL was under investigation in several institutions, notably by staff at the Royal Library in the Netherlands. An international group of experts from the Z39.50 community collaborated on a draft of this new protocol for the Internet/web/XML environment. The SRU specifications were first published in 2002 and have been popular for use in new applications because of the ease of implementation.[(3)]

SRU is very flexible. It is XML-based and the most common implementation is SRU via URL, which uses the HTTP GET for message transfer. Other versions, however, can be run over the web's SOAP protocol (SRU via SOAP), which supports more web service features, and over HTTP POST (SRU via POST), which avoids some length and character set restrictions that are currently present with HTTP GET. The records returned in response to a search may be in any well-defined XML format.

The Z39.50 and SRU Model

The functional model for searching different sites with the SRU protocol is similar to the model used for the Z39.50 protocol. The end user creates a search request on the user (home) system, which employs a specific local query syntax. For searching a target system, with its different specific query syntax, database design(s), and indexing conventions, the end user's local query is turned into a standard format. The target site receives that standard search message composed of protocol and query, and translates it into the syntax that its databases understand. If the query is very detailed or the search parameters are not supported at the target, the target may either reject the search as unsupported or in some cases ignore/change attributes it does not like and go on

with the altered search.

The protocol part of the search that accompanies the query specifies to the target several preferences: format for retrieved records, number of records to return, and other result set information, etc. The protocol then supports replies to the query which may include retrieved records, indication of errors, specification of the actual format of the returned records, etc.

Since a key issue in intersystem search is the difference in indexing at different sites, to improve the quality of intersystem search the Z39.50 and SRU developers modeled a facility that enables the user to first ask the target what it supports, expecting the target to be able to express that in a standard way as specified in the protocol. Using the information obtained about the target's indexes, it was expected that a more successful search could be formulated by the user system.

SRU Operations

SRU has three basic operations that enable the above model: Explain, Search/Retrieve, and Scan.

SRU Explain. Based on considerable experience with Z39.50, SRU developers treated the Explain operation as a priority. Explain is the name of the facility described above that allows the target to tell the user what it can do. The target has a description of itself in a standard XML format that can be easily retrieved by the user system via an Explain request, and this description can help the user to formulate queries. The Explain record has a number of other descriptive fields such as the XML formats that can be provided for retrieved records or the default number of records that can be returned. SRU targets do not have to maintain an Explain record, but are strongly urged to do so.

SRU Search/Retrieve. The Search/Retrieve operation is the primary functionality of SRU. It manages the sending of a query to the target along with the protocol preference information and manages the response to that query such as the return of records that match the query and related information. The query syntax used is the Contextual Query Language described below.

SRU Scan. SRU Scan enables the user to request to see the terms around the one(s) specified in a parameter, and the number of items that include those terms - if, of course, the target supports this type of browse capability. While the Search/Retrieve operation enables searching indexes of terms for matching records, the Scan operation allows the user to request a range of the available terms at a given point within that index. This enables users to see an ordered list of terms and, if supported, how many hits there would be for a search on each term. Scan is often used to select terms for subsequent searching or to verify visually a negative search result.

Example of use. SRU is being implemented initially by Z39.50 sites as an additional avenue to catalogs - and by non-Z39.50 sites as an easier avenue to external search. The Library of Congress implemented SRU as a target site in 2004, employing a gateway to the Z39.50 that is part of its catalog vendor software, rather than going directly to the databases. This software that includes SRU also enabled LC to improve its straight Z39.50 responses and offer to return

records in MARC 21, MARCXML, MODS, and even DC. The Library is in the process of implementing SRU to a number of other databases that are not natively MARC oriented, but it is anticipated that MARCXML and MODS will be offered in responses.

CQL - Contextual Query Language

A key component of the Search/Retrieve operation is the query. SRU creators developed a query syntax that is both rich and simple - and well suited to getting the most out of library metadata. That query language is the Contextual Query Language, or CQL as it is usually called.[4]

CQL is a formal language for representing queries. It was designed to accommodate information retrieval systems such as web indexes, bibliographic catalogs, and museum collection information. Unlike the query syntax commonly used with Z39.50, CQL is intended to be human readable and writable, and reasonably intuitive, while maintaining important components of the highly expressive (and complex) Z39.50 query language. Thus it is more powerful than a simple Google-like language. As the web site for SRU notes "CQL tries to combine simplicity and intuitiveness of expression for simple, every day queries, with the richness of more expressive languages to accommodate complex concepts when necessary."[5] While CQL strives to be human readable, it is still expected that end users will have a friendly interface which may simply be their local catalog interface and syntax.

CQL is based on the definition of a set of abstract access points, such as title, author, subject and refinements of those such as personal author, uniform title, geographical subject. While large data bases generally have some form of indexing structure associated with them, and the abstract access points of the CQL are often called abstract "indexes", CQL does not actually mandate the existence of "physical" indexes at the target but the ability to retrieve as if there were. CQL does not make presumptions about the database design - whether it is relational, object, hierarchical, network, etc. - but it is biased toward searching metadata that is identified (i.e., records as data rather than as documents) to enable "smart" searching. Thus while it can support simple searches, that use does not take advantage of its strength.

A Different Model, OpenSearch

In March 2005, Amazon demonstrated a new service that they called OpenSearch which is an implementation of a search protocol developed by the software company A9.[6] While Opensearch/A9 may be related to a number of different services, such as RSS feeds, the focus here is on the search protocol component - its functional model, how it differs from SRU, and the comparative advantages and possible combination of the two approaches.

Similar to the Library community models, Amazon recognized that "traditional search engines often cannot do a good job of indexing content from specialized web sites and that local search engines may understand local content far better: 'Different types of content require different types of search engines. And most of the time the best search engines for a site are the ones written by those that know the content the best.'"[7] This is a recognition that, for example, a

medical information catalog would probably provide highly specialized and precise search and retrieval and that even in general material databases, searching may be oriented to a special clientele, such as schoolchildren or researchers or art students. Thus there may be functional diversity in the implementations of search against files. This is essentially the indexing diversity problem that other external search protocols such as Z39.50 and SRU recognized and for which they attempted to provide solutions.

In its quest to tap into specialized search implementations, Amazon and A9 developed the following model. The user system sends a request to the target for information about the target's local system. The target sends back the parameter names used locally for search activities. The user then sends a query to the target using the target's "language" and receives retrieved records from the target in RSS format. While the basic concept is interesting and the interactions are mostly compatible with SRU, by avoiding as much complexity as possible, OpenSearch makes it very easy to do simple keyword search, but significantly harder to do anything with more precision.
The Opensearch client first requests some minimal information about the target - notably the names of the parameters to use for the target's local search form. The particulars of access points supported by the target are not addressed. There is no need for a standard search syntax like CQL to convey refined queries as the queries can only be keywords. There is no choice on the syntax of the records returned either.

Open Search is useful for providing a very low threshold search protocol that primarily supports keyword searching. It accepts the fact that different databases may have differently defined searches and simply uses a keyword search that will be treated as the target sees fit. There is a presumption that the end user may not need to know how the term is being treated by the target.

So OpenSearch and SRU may be contrasted in that OpenSearch does not try to understand the target but simply to enable the target to treat a search term as a keyword. SRU can support such simple searching, but it has the capability to attempt an understanding of the target and to interpret specifics of its user's request to obtain the more refined results that the user wants - through the Explain and the development of abstract access points. SRU also supports request of a specific XML syntax for records returned, whereas OpenSearch simplified its protocol by restricting retrieved information to RSS syntax.

The OpenSearch approach is especially valuable for searching across the many unstructured servers and document files. Although the results may be heterogeneous, they are adequate for some purposes - or as trial investigations of sites that might then be searched directly. SRU is best used against reasonably structured catalogs where a user desires a more structured searching capability and more control over the results. Searchers accustomed to a local catalog in libraries would find SRU more useful for its similarity to the local searching; searchers coming from Google searching might find the OpenSearch results satisfactory.

The two protocols are compatible and discussions have been initiated between the A9 and SRU developers.[8] The goal would be enabling OpenSearch to expand to richer searching and bridge

the gap between environments that don't require the somewhat more strict SRU but do require more functionality than OpenSearch currently provides.

XQuery

XQuery is a query syntax for XML data under development by the World Wide Web Consortium (W3C).[9] Work has progressed slowly for about 7 years, with a fundamental disagreement between those who prefer to view XML as document markup (identifying paragraphs, chapter starts, etc.) and those who prefer XML as data markup (identifying names, subjects, dates, etc.). It is dependent on knowledge of the XML tagging of the documents it is searching and is oriented to working with the primary material rather than indexes. But XQuery, while a highly complex (and complete, if anything in this area can be complete) query language, may find its major use for managing queries within systems rather than in client/server applications. The SRU and OpenSearch, which are client/server protocols, may interact with XQuery applications when they reach the server site as they do today with SQL (Structured Query Language) and other database search software.

In Conclusion

This paper has been an exploration of some leading "acronyms" that are currently popular in the client/server information search environment - search protocols such as SRU and OpenSearch/A9 and query languages such as CQL and XQuery. Libraries have the need various approaches -"smart" searching of structured data in catalogs and unstructured search of digitized documents and catalog files. SRU and OpenSearch are innovative and thoughtful search solutions for the diversity of search systems that server targets offer. Collaboration by their developers is welcome news and will hopefully result in enhancement to both approaches. Meanwhile, CQL offers a well vetted and reasonably structured search syntax for client/server applications - today - that is adaptable to a variety of server idiosyncrasies; whereas the comprehensive and highly complex XQuery may provide a foundation for future developments in search.

Notes

(1) The author thanks Ray Denenberg, Library of Congress, and Robert Sanderson, University of Liverpool, for reviewing and supplying information for this paper; however, any remaining errors are mine.
(2) The Z39.50 protocol web site is <http://www.loc.gov/z3950/agency/>. There are links to the ANSI/NISO version of the standard at that site and a print of the ISO version of the standard is available from ISO (ISO 23950).
(3) The SRU web site is <http://www.loc.gov/standards/sru/>. The SRU specification, development descriptions, and supporting documents are available from that site.
(4) The CQL web site is <http://www.loc.gov/standards/sru/cql/>
(5) <http://www.loc.gov/standards/sru/cql/>. (Accessed May 30, 2006)
(6) Wiggins, Richard W., "Amazon's New OpenSearch Enables Search Syndication", March 28,

2005 <www.infotoday.com/> (Accessed April 22, 2006)
(7) From the OpenSearch FAQ located at <http://opensearch.a9.com/docs/faq.jsp> (Accessed April 22, 2006)
(8) “Rob Sanderson visits A9.com”
<http://blog.a9.com/blog/2006/04/14/rob-sanderson-visits-a9com/> (Accessed June 29, 2006)
(9) “XML Query (XQuery) Requirements”, W3C Working Draft 3 June 2005
<http://www.w3.org/TR/xquery-requirements> (Accessed May 30, 2006)