Open Search Environments: The Free Alternative to Commercial Search Services.

Adrian O'Riordan

ABSTRACT

Open search systems present a free and less restricted alternative to commercial search services. This paper explores the space of open search technology, looking in particular at lightweight search protocols and the issue of interoperability. A description of current protocols and formats for engineering open search applications is presented. The suitability of these technologies and issues around their adoption and operation are discussed. This open search approach is especially useful in applications involving the harvesting of resources and information integration. Principal among the technological solutions are OpenSearch, SRU, and OAI-PMH. OpenSearch and SRU realize a federated model to enable content providers and search clients communicate. Applications that use OpenSearch and SRU are presented. Connections are made with other pertinent technologies such as open-source search software and linking and syndication protocols. The deployment of these freely licensed open standards in web and digital library applications is now a genuine alternative to commercial and proprietary systems.

INTRODUCTION

Web search has become a prominent part of the Internet experience for millions of users. Companies such as Google and Microsoft offer comprehensive search services to users free with advertisements and sponsored links, the only reminder that these are commercial enterprises. Businesses and developers on the other hand are restricted in how they can use these search services to add search capabilities to their own websites or for developing applications with a search feature. The closed nature of the leading web search technology places barriers in the way of developers who want to incorporate search functionality into applications. For example, Google's programmatic search API is a RESTful method called Google Custom Search API that offers only 100 search queries per day for free.[1] The limited usage restrictions of these APIs mean that organizations are now frequently looking elsewhere for the provision of search functionality.

Free software libraries for information retrieval and search engines have been available for some time allowing developers to build their own search solutions. These libraries enable search and retrieval of document collections on the Web or offline. Web crawlers can harvest content from multiple sources. A problem is how to meet users' expectations of search efficacy while not having

Adrian O'Riordan (a.oriordan@cs.ucc.ie) is Lecturer, School of Computer Science and Information Technology, University College, Cork, Ireland.

the resources of the large search providers. Reservations about the business case for free open search include that large-scale search is too resource-hungry and the operational costs are too high; but these suppositions have been challenged.[2]

Open search technology enables you to harvest resources and combine searchers in innovative ways outside the commercial search platforms. Further prospects for open search systems and open-source search lie in areas such as peer-to-peer, information extraction, and subject-specific technology.[3] Many search systems unfortunately use their own formats and protocols for indexing, search, and results lists. This makes it difficult to extend, alter, or combine services.

Distributed search is the main alternative to building a "single" giant index (on mirrored clusters) and searching at one site, *a la* Google Search and Microsoft Bing. Callan describes the distributed search model in an information retrieval context.[4] In his model, information retrieval consists of four steps: discovering databases, ranking databases by their expected ability to satisfy the query, searching the most relevant databases, and merging results returned by the different databases. Distributed search has become a popular approach in the digital libraries field. Note that in the digital libraries literature distributed search is often called federated search.[5] The federated model has clear advantages in some application areas. It is very hard for a single index to do justice to all the possible schemas in use on the web today. Other potential benefits can come from standardization. The leading search engine providers utilize their own proprietary technologies for crawling, indexing, and the presentation of results. The standardization of result lists would be useful for developers combining information from multiple sources or pipelining search to other functions. A common protocol for declaring and finding searchable information is another desirable feature.

Standardized formats and metadata are key aspects of search interoperability, but the focus of this article is on protocols for exchanging, searching, and harvesting information. In particular, this article focuses on lightweight protocols, often REST (Representational state transfer)–style applications. Lightweight protocols place less onerous overheads on development in terms of adapting existing systems and additional metadata. They are also simpler. The alternative is heavyweight approaches to federated search, such as using web services or service-oriented architectures.

There have been significant efforts at developing lightweight protocols for search, primary among which is the OpenSearch protocol developed by an Amazon subsidiary. Other protocols and services of relevance are SRU, MXG, and the OAI-OMH interoperability framework. We describe these technologies and give examples of their use. Technologies for the exchange and syndication of content are often used as part of the search process or in addition. We highlight key differences between protocols and give instances where technologies can be used in tandem.

This paper is structured as follows. The next section describes the open search environment and the technologies contained therein. The following section describes open search protocols in detail, giving examples. Finally, summary conclusions are presented.

AN OPEN SEARCH ENVIRONMENT

A search environment (or ecosystem) consists of a software infrastructure and participants. The users of search services and the content providers and publishers are the main participants. The systems infrastructure consists of the websites and applications that both publish resources and present a search interface for the user, and the technologies that enable search. Technologies include publishing standards for archiving and syndicating content, the search engines and web crawlers, the search interface (and query languages), and the protocols or glue for interoperability. Baeza-Yates and Raghavan present their vision of next-generation web search highlighting how developments in the web environment and search technology are shaping the next generation search environment.[6]

Open-source libraries for the indexing and retrieval of document collections and the creation of search engines include the Lemur project (and the companion Indri search engine),[7] Xapian,[8] Sphinx,[9] and Lucene (and the associated Nutch Web crawler).[10] All of these systems support web information retrieval and common formats. From a developer perspective they are all cross-platform; Lemur/Indri, Xapian, and Sphinx are in C and C++ whereas Lucene/Nutch is in Java. Xapian has language bindings for other programming languages such as Python. The robustness and scalability of these libraries support large-scale deployment, for example the following large websites use Xapian: Citebase, *Die Zeit* (German newspaper), and Debian (Linux distribution).[11] Middleton and Baeza-Yates present a more detailed comparison of open-source search engines.[12] They compare twelve open-source search engines including Indri and Lucene mentioned above across thirteen dimensions. Features include license, storage, indexing, query preprocessing (stemming, stop-word removal), results format, and ranking.

Apache Solr is a popular open-source search platform that additionally supports features such as database integration, real-time indexing, faceted search, and clustering.[13] Solr uses the Lucene search library for the core information retrieval. Solr's rich functionality and the provision of RESTful HTTP/XML and JSON APIs makes it an attractive option for open information integration projects. In a libraries context Singer cites Solr as an open-source alternative for next-generation OPAC replacements.[14] Solr is employed, for example, in the large-scale Europeana project.[15]

The focus of much of this paper is on the lightweight open protocols and interoperability solutions that allow application developers to harvest and search content across a range of locations and formats. In contrast, the DelosDLMS digital library framework exemplifies a heavyweight approach to integation.[16] In DelosDLMS, services are either loosely or tightly coupled in a service-oriented architecture using web service middleware.

Particular issues for open search are the location and collection of metadata for searchable resources and the creation of applications offering search functionality. Principal among these technological solutions are the OpenSearch and SRU protocols. Both implement what Alipour-Hafezi et al. term a federated model in the context of search interoperability, wherein providers agree that their services will conform to certain standard specifications.[17] The costs and adoption risk of this approach are low. Technologies such as OpenSearch occupy an abstraction layer above existing search infrastructure such as Solr.

Interoperability

Interoperability is an active area of work in both the search and digital library fields. Interoperability is "the ability of two or more systems or components to exchange information and to use the information that has been exchanged."[18] Interoperability in web search applies to resource harvesting, meta-search, and to allow search functions interact with other system elements. Interoperability in digital libraries is a well-established research agenda,[19] and it has been described by Paepcke et al. as "one of the most important problems to solve [in DLs]."[20] Issues that are common to both web search and digital-library search include metadata incompatibilities, protocol incompatibilities, and record duplication and near-duplication. In this paper, the focus is on search protocols; for a comprehensive survey of technology for semantic interoperability in digital library systems, see the DELOS report on same.[21] A comprehensive survey of methods and technology for digital library interoperability is provided in a DL.org report.[22]

Formats and Metadata

Free and open standard formats are extensive in web technology. Standard or de facto formats for archiving content include Plain text, Rich Text Format (RTF), HTML, PDF, and various XML formats. Document or resource identification is another area where there has been much agreement. Resource identification schemes need to be globally unique, persistent, efficient, and extensible. Popular schemes include URLs, Persistent URLs (PURLs), the Handle system (handle.net), and DOIs (Digital Object Identifiers). Linking technologies include OpenURL and COinS. OpenURL links sources to targets using a knowledge base of electronic resources such as digital libraries.[23] ContextObjects in Spans (COinS), as used in Wikipedia for example, is another popular linking technology.[24]

Applications can use various formats for transporting and syndicating content. Syndication and transport technologies include XML, JSON, RSS/Atom, and heavyweight web service-based approaches. Much of the metadata employed in digital libraries is in XML formats, for example in MARCXML and the Metadata Encoding and Transmission Standard (METS).

The World Wide Web Consortium defined RDF (Resource Description Framework) to provide among other goals "a mechanism for integrating multiple metadata schemes."[25] RDF records are defined in an XML namespace. In RDF, Subject-predicate-object expressions represent web resources and typically identified by means of an URI (Universal Resource Identifier). JSON (JavaScript Object Notation) is a lightweight data-interchange format that has become popular in Web-based applications and is seeing increasing support in digital library systems.[26]

Harvesting

Web content is harvested using software called web crawlers (or web spiders). A crawler is an instance of a software application that runs automated tasks on the web. Specifically the crawler follows web links to index websites. There was been little standardization in this area except the Robot Exclusion Standard and use of various XHTML meta-elements and HTTP header fields. There is a lot of variability in terms of the policy for the selection of content sources, policy for following links, URL normalization, politeness, depth of crawl, and revisit policy. Consequently, there are many web crawling systems in operation; open-source crawlers include DatapartSearch, Grub, Heritrix, and the aforementioned Nutch.

Harvesting and syndication of metadata from open repositories is the goal of the Open Archives Initiative Protocol for Metadata Harvesting (OAI-PMH), originally developed by Los Alamos National Laboratory, Cornel, and NASA in 2000 and 2001.[27] Resource harvesting challenges include scale, keeping information up-to-date, robustness, and security. OAI-PMH has been adopted by many digital libraries, museums, archives, and publishers. The latest version, OAI-PMH 2.0, was released in 2012.

OAI-PMH specifies a general application-independent model of network-accessible repository and client harvester that issues requests using HTTP (either GET or POST). Metadata is expressed as a record in XML format. An OAI-PHM implementation must support Dublin Core, with other vocabularies as additions. OAI-PMH is the key technology in the harvesting model of digital library interoperability described by Van de Sompel et al.[28] An OAI-PMH-compliant system consists of harvester (client), repository (network accessible server), and items (constituents of a repository).

Portal sites such as Europeana and OAIster use OAI-PMH to harvest from large numbers of collections.[29] There are online registries of OAI-compliant repositories. The European Commission's Europeana allow users to search across multiple image collections including the British Library and the Louvre online. Another portal site that uses OAI-PMH is CultureGrid, operated by the UK Collections Trust. CultureGrid provides access to hundreds of museum, galleries, libraries, and archives in the UK. The Apache Software Foundation has developed a module, mod_oai, for Apache Webservers that helps crawlers to discover content.

Syndication and Exchange

Here we outline lightweight options for syndication and information exchange. Heavyweight web services-based approaches are outside the scope of this article. Web syndication commonly uses RSS (Really Simple Syndication) or its main alternative, Atom. Atom is a proposed IETF standard.[30] RSS 2.0 is the latest version in the RSS family of specifications, a simple yet highly extensible format where content items contain plain text or escaped HTML.[31] Atom, developed to counter perceived deficiencies in RSS, has a richer content model than RSS and is more reusable in other XML vocabularies.[32] Both RSS and Atom use HTTP for transport. RSS organizes information into channels and items, Atom into feeds and entries. Extension as modules allows RSS to carry multimedia payload (RSS enclosures) and geographical information (GeoRSS). Atom has an associated publishing protocol called AtomPub. Syndication middleware, which supports multiple formats, can serve as an intermediary in application architectures.

Information and Content Exchange (ICE) is a protocol that aims to "automate the scheduled, reliable, secure redistribution of any content."[33] TwICE is a Java implementation of ICE. ICE automates the establishment of syndication relationships and handles data transfer and results formatting. This gives content providers more control over delivery, schedule, and reliability than simple web syndication without deploying a full-scale web services solution.

The Open Archives Initiative—Object Reuse and Exchange (OAI-ORE) protocol provides standards for the description and exchange of aggregations of web resources.[34] This specification standardizes how compound digital objects can combine distributed resources of multiple media types. ORE introduces the concepts of aggregation, resource map, and proxy resource. Resource providers or curators can express objects in RDF or Atom format and assign HTTP URIs for identification. ORE supports resource discovery so crawlers or harvesters can find these resource maps and aggregates. ORE can work in partnership with OAI-PMH.

We outline some additional lightweight technologies for information exchange to conclude this section. OPML (Outline Processor Markup Language) is a format that represents lists of web feeds for services such as aggregators.[35] It is a simple XML format. Feedsync and ROME support format-neutral feed formats that abstract from wire formats such as RSS 2.0 and Atom 1.0 for aggregator or syndication middleware. These technologies are described in the literature.[36] LOCKSS (Lots of Copies Keep Stuff Safe) is a novel project that users a peer-to-peer network to preserve and provide access to web resources. For example, the MetaArchive Cooperative uses LOCKSS for digital preservation.[37]

Meta-search

Meta-search is where multiple search services are combined. Such services have a very small share of the total search market owing to the dominance of the big players. MetaCrawler, developed in the 1990s, was one of the first meta-search engines and serves as a model for how such systems operate.[38] A meta-search engine utilizes multiple search engines by sending a user request to multiple sources or engines aiming to improve recall in the process. A key issue with meta-search is how to weight search engines and how to integrate sets of results into a single results list. Figure 1 shows a general model of meta-search where the meta-search service chooses which search engines and content providers to employ. Active meta-search engines on the web include dogpile, Yippy, ixquick, and info.com. Note that these types of website appear, change names, and disappear frequently. Currently meta-search services use various implementation methods such as proprietary protocols and screen scraping.

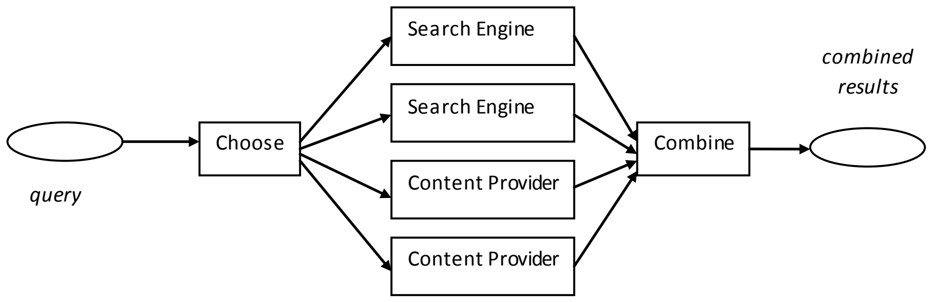

Figure 1. General model of meta-search.

Metasearch XML Gateway (MXG) is a meta-search protocol developed by the NISO MetaSearch Initiative, a consortium of meta-search developers and interested parties.[39] MXG is a message and response protocol, which enables meta-search service providers and content providers communicate. A goal of the design of MXG was that content providers should not have to expend substantial development resources. MXG, based on SRU, specifies both the query and search results formats. Combining results, aggregation and presentation are not part of the protocol and handled by the meta-search service. The standard defines three levels of compliance allowing varying degrees of commitment and interoperability.

SEARCH PROTOCOLS

We describe OpenSearch and SRU, along with applications, in the following subsections. After that, we detail related technologies.

OpenSearch

OpenSearch is a protocol that defines simple formats to help search engines and content providers communicate. It was developed by A9, a subsidiary of Amazon, in 2005.[40] It defines common formats for describing a search service, query results, and operation control. It does not specify content formats for documents or queries. The current specification, version 1.1, is available with a Creative Commons license. It is an extensible specification with extensions published on the website. Both free open systems and proprietary systems use OpenSearch. In particular, many open-source search engines and content management systems support OpenSearch, including YaCy, Drupal and Plone CMS.

OpenSearch consists of a description file for search source and a response format for query results. Descriptors include elements Url, Query, SyndicationRight, and Language. Resource identification can be by URLs, DOIs, or a linking technology such as OpenURL. Responses describe a list of results and can be in RSS, Atom, or HTML formats. Additionally there is an auto-discovery feature to signal that a HTML page is searchable, implemented using a HTML 4.0 <link> element.

OpenSearch makes very few assumptions about the types of sources, the type of query or the search operation. It is ideal for combining content from multiple disparate sources, which may be data from repositories, webpages, or syndicated feeds.

For illustrative purposes, listing 1 gives an example OpenSearch description for harvesting book information from an example digital library called DigLib. The root node <openseachdescription> includes an XML namespace attribute, which gives the URL for the standard version. The *url* element specifies the content type (a MIME type), the query (book in this case), and the index offset where to begin. The *rel* attribute states that the result is a collection of resources.

```
<openseachdescription xmlns=http://a9.com/~/spec/opensearch/1.1/>
<shortname>DigLib</shortname>
<description>Harvests book items</description>
<url type="application/rdf+xml" indexoffset="0" rel="collection"
template="http://diglib.com/?q={book}&amp;start={startIndex?}&amp;format=rdf" />
<language>en-us</language>
<outputencoding>UTF-8</outputencoding>
</opensearchdescription>
```

Listing 1. XML OpenSearch Description Record.

Next, we describe some deployed applications that use OpenSearch. OJAX uses qeb technologies such as Ajax (Asynchronous Javascript) to provide a federated search service for OAI-PMH compatible open repositories.[41, 42] OJAX also supports the Discovery feature of OpenSearch, as described in the OpenSearch 1.1 specification, for auto-detecting that a repository is searchable. Stored searches are in Atom format.

Open-source meta-search engines can combine the results of OpenSearch enabled search engines.[43] A system built as a proof-of-concept uses four search sources: A9.com, yaCy, mozDex, and alpha. A user can issue a text query (word or phrase) with Boolean operators and several modifiers. Users can prefer or bias particular engines by setting weights. The system ranks results, combined using a voting algorithm and implemented using the Lucene library.

OpenSearch can be employed to specify the search sources and as a common format when results are combined. As LeVan points out "the job of the meta-search engine is made much simpler if the local search engine supports a standard search interface."[44] LeVan also mentions MXG in this context. Nguyen et al. describe an application where over one hundred search engines are used in experiments in federated search.[45] The search sources were mostly OpenSearch-compliant search engines. An additional tool scrapes results from noncompliant systems.

InterSynd uses OpenSearch to help provide a common protocol for harvesting web feeds. InterSynd is a syndication system that harvests, stores, and provides feed recommendations.[36] It uses Java.net's ROME (RSS and atOM utilitiEs) library to represent feeds in a format-neutral way.

InterSynd is syndication middleware that allows sources to post and services to fetch information in all major syndication formats (see figure 2). Its feed-discovery module, Disco, uses the Nutch crawler and the OpenSearch protocol to harvest feeds. Nutch is an open-source library for building search engines that supports OpenSearch. Nutch builds on the Lucene information retrieval library, adding web-specifics, such as a crawler, a link-graph database, and parsers for HTML.

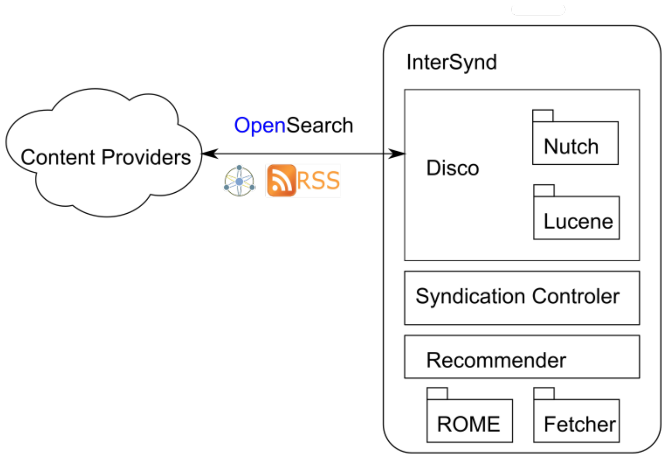

Figure 2. OpenSeach in InterSynd.

OpenSearch 1.1 allows returned results in either RSS 2.0 or Atom 1.0 format or an OpenSearch format, the "bare minimum of additional functionality required to provide search results over RSS channels" (quoted from A9 Website). Listing 2 below shows a Disco results list in RSS 2.0 format. OpenSearch fields appear in the channel description. The Nutch fields appear within each item (not shown). An OpenSearch namespace is specified in the opening <rss> XML element. The following additional OpenSearch elements appear in the example: *totalResults*, *itemsPerPage* and *startIndex*.

```
<?xml version="1.0" encoding="UTF-8"?>
<rss version="2.0"
 xmlns:nutch="http://www.nutch.org/opensearchrss/1.0/"
 xmlns:opensearch="http://a9.com/-/spec/opensearch/1.1/">
 <channel>
   <title>Nutch: metasearch</title>
   <description>Nutch search results for query: metasearch</description>
 <link>http://localhost/nutch-1.6-
dev/opensearch?query=metasearch&amp;start=0&amp;hitsPerSite=2&amp;hitsPerPage=10</link>
   <opensearch:totalResults>282</opensearch:totalResults>
   <opensearch:startIndex>0</opensearch:startIndex>
   <opensearch:itemsPerPage>10</opensearch:itemsPerPage>
   <nutch:query>metasearch</nutch:query>
<item> cut... </item>
more items cut...
</channel>
</rss>
```

Listing 2. Results Produced using Nutch with OpenSearch.

We mention one more application of OpenSearch here. A series of NASA projects to develop a set of interoperable standards for sharing information employs various open technologies for sharing and disseminating datasets including OpenSearch for its discovery capability.[46] Discovery of document and data collections is by keyword search, using the OpenSearch protocol.

There are various extensions to OpenSearch. For example, an extension to handle SRU allows SRU (Search and Retrieval via URL) queries within OpenSearch contexts. Other proposed extensions include support for mobility, e-commerce, and geo-location.

SRU (Search/Retrieval via URL)

A technology with some similarities to OpenSearch but more comprehensive is SRU (Search/Retrieval via URL).[47] SRU is an open RESTful technology for web search. The current version is SRU 2.0, standardized by the Organization for the Advancement of Structured Information Standards (OASIS) as searchRetrieve. Version 1.0. SRU was developed to provide functionality similar to the widely deployed Z39.50 standard for library information retrieval updated for the web age.[48] SRU addresses aspect of search and retrieval by defining models: a data model, a query model, and processing model, a result set model, a diagnostics model and a description-and-discovery model. SRU is extensible and can support various underlying low-level technologies. Both Lucene and DSpace implementations are available. The OCLC implementation of SRU supports both RSS and Atom feed formats and the Atom Publishing Protocol.

SRU uses HTTP as the application transport and XML formats for messages. Requests can be in the form of either GET or POST HTTP methods. SRU supports a high-level query language called Contextual Query Language (CQL). CQL is a human-readable query language consisting of search clauses. SRU operation involves three parts: Explain, Search/Retrieve and Scan. Explain is a way to publish resource descriptions. Search/Retrieve entails the sending of requests (formulated in CQL) and the receiving of responses over HTTP. The optional SRU Scan enables software to query the index. The result list is in XML Schema format. The meta-search service MXG uses SRU, but relaxes the requirement to use CQL.[39] SRW (Search/Retrieve Web Service) is a web services implementation of SRU that uses SOAP as the transfer mechanism.

Hammond combines OpenSearch with SRU technology in an application for nature publishers.[49] He also points out the main differences between the protocols such as SRU's use of a query specification language and differences in the results records. As well as supporting OpenSearch data formats (RSS and Atom), the nature application also supports JSON (Javascript Object Notation). OpenSearch is used for formatting the result sets whereas SRU/CQL is used for querying. This search application launched as a public service in 2009.

Listing 3 below is an example from the nature application showing CQL search queries (<Query> tags) used in an OpenSearch description document. Note how both the SRU *queryType* and the

OpenSearch *searchTerms* attributes appear in the query. Further details on how to use SRU and OpenSearch together are on the OpenSearch Website.

```
<?xml version="1.0" encoding="UTF-8"?>
<OpenSearchDescription xmlns="http://a9.com/-/spec/opensearch/1.1/"
               xmlns:sru="http://a9.com/-/opensearch/extensions/sru/2.0/">
 <ShortName>nature.com</ShortName>
 <LongName>OpenSearch interface for nature.com</LongName>
 <Description>The nature.com OpenSearch service </Description>
 <Tags>nature.com opensearch sru</Tags>
 <!-- URL Template for SRU -->
 <Url type="application/sru+xml" indexOffset="1"
template="http://www.nature.com/opensearch/request?version=1.1&operation=searchRetrieve&quer
y={searchTerms}&queryType={sru:queryType?}&httpAccept=application/sru%2Bxml&startRecord={st
artIndex?}&maximumRecords={count?}&sortKeys={sru:sortKeys?}&stylesheet={sru:stylesheet?}"/>
 <!-- Example Queries -->
 <Query role="example" sru:queryType="searchTerms" searchTerms="metasearch" />
 <Query role="example" sru:queryType="cql" searchTerms="metasearch" />
 <Query role="example" sru:queryType="cql" searchTerms="cql.keywords=metasearch" />
 <Query role="example" sru:queryType="cql" sru:sortKeys="title,pam,1"
searchTerms="cql.keywords=metasearch" />
 <Query role="example" sru:queryType="cql" sru:stylesheet="http://example.org/example.xsl"
searchTerms="cql.keywords=metasearch" />
</OpenSearchDescription>
```

Listing 3. Example using SRU and OpenSearch.

Other Technologies

Here we more briefly survey some additional technologies of relevance to open-search interoperability. XML-based approaches to information integration, such as the use of XQuery, are an option but do not present a loose integration.

Chudnov et al. describes a simple API for a copy function for web applications to enable syndication, searching, and linking of web resources.[50] Called unAPI, it requires small changes for publishers to add the functionality to web resources such as repositories and feeds. Developers can layer unAPI over SRU, OpenSearch, or OpenURL.[51]

Announced in 2008, Yahoo!'s SearchMonkey technology, also called Yahoo!'s Open Search Platform, allowed publishers to add structured metadata to Yahoo! Search results. SearchMonkey divided the problem into two parts: metadata extraction and result presentation. In is not clear how much of this technology survived Yahoo! and Microsoft's new Search Alliance, signed in 2010.[52] Mika described a search interface technology called Microsearch that is similar in nature.[53] In Microsearch, semantic fields are added a search and search result presentation enriched with metadata extracted from retrieved content. Govaerts et al. described a federated search and recommender system that operates as a browser add-on. The system is OpenSearch-compliant and all results are in the Atom format.[54]

The Corporation for National Research Initiatives (CNRI) Digital Object Architecture (DOA) provides a framework for managing digital objects in a networked environment. It consists of three parts: a digital object repository, a resolution mechanism (Handle system), and a digital object registry. The Repository Access Protocol (RAP) proves a means of networked access to digital objects, which supports authentication and encryption.[55]

SUMMARY AND CONCLUSIONS

A rich set of formats and protocols and working implementations show that open search technology is an alternative to the dominant commercial search services. In particular, we discussed the lightweight OpenSearch and SRU protocols as suitable glue to create loosely coupled search-based applications. These can complement other developments in resource discovery and description, open repositories, and open-source information retrieval. The flexibility and extensibility offers exciting opportunities to develop new applications and new types of applications. The successful deployment of open search technology shows that this technology has matured to support many uses. A fruitful area of further development would be to make working with these standards easier for developers and even accessible to the nonprogrammer.

REFERENCES

1. Google Custom Search API, https://developers.google.com/custom-search/v1/overview.
2. Mike Cafarella and Doug Cutting, "Building Nutch: Open Source Search: A Case Study in Writing an Open Source Search Engine," *ACM Queue* 2, no. 2 (2004), http://0-dl.acm.org.library.ucc.ie/citation.cfm?doid=988392.988408.
3. Wray Buntine et al., "Opportunities from Open Source Search," in *Proceedings, the 2005 IEEE/WIC/ACM International Conference on Web Intelligence*, 2–8 (2005), http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=1517807.
4. Jamie Callan, "Distributed Information Retrieval," *Advances in Information Retrieval* 5 (2000): 127–50.
5. Péter Jacsó, "Internet Insights—Thoughts About Federated Searching," *Information Today* 21, no. 9 (2004): 17–27.
6. Ricardo Baeza and Prabhakar Raghavan, "Next Generation Web Search," in *Search Computing* (Berlin Heidelberg: Springer, 2010): 11–23, http://link.springer.com/chapter/10.1007/978-3-642-12310-8_2.

7. Trevor Strohman et al., "Indri: A Language Model-Based Search Engine for Complex Queries," in *Proceedings of the International Conference on Intelligent Analysis* 2, no. 6, (2005): 2–6.

8. Xapian project website, http://xapian.org/.

9. Andrew Aksyonoff, *Introduction to Search with Sphinx: From Installation to Relevance Tuning* (Sebastopol, CA: O'Reilly, 2011).

10. Rohit Khare, "Nutch: A Flexible and Scalable Open-Source Web Search Engine," Oregon State University, 2004, p. 32, http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.105.5978

11. "Xapian Users," http://xapian.org/users.

12. Christian Middleton and Ricardo Baeza-Yates, "A Comparison of Open Source Search Engines," 2007, http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.119.6955.

13. Apache Solr, http://lucene.apache.org/solr/.

14. Ross Singer, "In Search of a Really 'Next Generation' Catalog," *Journal of Electronic Resources Librarianship* 20, no. 3 (2008): 139–42, http://www.tandfonline.com/doi/pdf/10.1080/19411260802412752.

15. Europeana portal, http://www.europeana.eu/portal/.

16. Maristella Agosti et al., *DelosDLMS—The Integrated DELOS Digital Library Management System* Berlin Heidelberg: Springer, 2007).

17. Mehdi Alipour-Hafezi et al., "Interoperability Models in Digital Libraries: An Overview," *Electronic Library* 28, no. 3 (2010): 438–52, http://www.emeraldinsight.com/journals.htm?articleid=1864156.

18. Institute of Electrical and Electronics Engineers, *IEEE Standard Computer Dictionary: A Compilation of IEEE Standard Computer Glossaries* (New York: IEEE, 1990).

19. Clifford Lynch and Hector García-Molina, "Interoperability, Scaling, and the Digital Libraries Research Agenda," in *IITA Digital Libraries Workshop*, 1995.

20. Andreas Paepcke et al., "Interoperability for Digital Libraries Worldwide," *Communications of the ACM* 41, no. 4 (1998): 33–42.

21. Manjula, Patel et al., ""Semantic Interoperability in Digital Library Systems," 2005, http://delos-wp5.ukoln.ac.uk/project-outcomes/SI-in-DLs/SI-in-DLs.pdf.

22. Georgios Athanasopoulos et al., "Digital Library Technology and Methodology Cookbook," Deliverable D3.4, 2011, http://www.dlorg.eu/index.php/outcomes/dl-org-cookbook.

23. Herbert Van de Sompel and Oren Beit-Arie, "Open Linking in the Scholarly Information Environment using the OpenURL Framework," *New Review of Information Networking* 7, no. 1 (2001): 59–76, http://www.tandfonline.com/doi/abs/10.1080/13614570109516969.

24. Daniel Chudnov, "COinS for the Link Trail," *Library Journal,* 131 (2006): 8-10.25. Lois Mai Chan and Marcia Lei Zeng, "Metadata Interoperability and Standardization—A Study of Methodology, Part II," *D-Lib Magazine* 12, no. 6 (2006), http://www.dlib.org/dlib/june06/zeng/06zeng.html.

26. JSON (JavaScript Object Notation), http://www.json.org/.

27. The Open Archives Initiative Protocol for Metadata Harvesting, http://www.openarchives.org/OAI/openarchivesprotocol.html.

28. Herbert Van De Sompel et al., "The UPS Prototype: An Experimental End-User Service across E-print Archives," *D-Lib Magazine* 6, no. 2 (2000), http://www.dlib.org/dlib/february00/vandesompel-ups/02vandesompel-ups.html.

29. OAIster, http://oaister.worldcat.org/.

30. Mark Nottingham, ed., "The Atom Syndication Format. RfC 4287," memorandum, IETF Network Working Group, 2005, http://www.ietf.org/rfc/rfc4287.

31. RSS 2.0 Specification, Berkman Center for Internet & Society at Harvard Law School, July 15, 2003, http://cyber.law.harvard.edu/rss/rss.html.

32. "RSS 2.0 And Atom 1.0 Compared," http://www.intertwingly.net/wiki/pie/Rss20AndAtom10Compared

33. Jay Brodsky et al., eds., "The Information and Content Exchange (ICE) protocol," Working Draft, Version 2.0, 2003, http://xml.coverpages.org/ICEv20-WorkingDraft.pdf.

34. Open Archives Initiative Object Reuse and Exchange, http://www.openarchives.org/ore/.

35. OPML (Outline Processor Markup Language), http://dev.opml.org/.

36. Adrian P. O'Riordan, and M. Oliver O'Mahoney, "Engineering an Open Web Syndication Interchange with Discovery and Recommender Capabilities," *Journal of Digital Information*, 12, no. 1 (2011), http://journals.tdl.org/jodi/index.php/jodi/article/viewArticle/962.

37. Vicky Reich and David S. H. Rosenthal, "LOCKSS: A Permanent Web Publishing and Access System," *D-Lib Magazine* 7, no. 6 (2001): 14, http://mirror.dlib.org/dlib/june01/reich/06reich.html.

38. Erik Selberg and Oren Etzioni, "Multi-service Search and Comparison Using the MetaCrawler," in Proceedings of the Fourth Int'l WWW Conference, Boston, 1995. **[pub info?]**

39. NISO Metasearch Initiative, Metasearch XML Gateway Implementers Guide, Version 1.0, NISO RP-2006-02, 2006, http://www.niso.org/publications/rp/RP-2006-02.pdf.

40. DeWitt Clinton, "OpenSearch 1.1 Specification, draft 5," http://opensearch.org/Specifications/OpenSearch/1.1.

41. Judith Wusteman, "OJAX: A Case Study in Agile Web 2.0 Open Source Development," in *Aslib Proceedings* 61, no. 3 (2009): 212–31, http://dx.doi.org/10.1108/00012530910959781.

42. Judith Wusteman and Padraig O'hlceadha, "Using Ajax to Empower Dynamic Searching," *Information Technology & Libraries* 25, no. 2 (2013): 57–64, http://0-www.ala.org.sapl.sat.lib.tx.us/lita/ital/sites/ala.org.lita.ital/files/content/25/2/wusteman.pdf.

43. Adrian P. O–Riordan, "Open Meta-Search with OpenSearch: A Case Study," technical report hosted at cora.ucc.ie repository, 2007, http://dx.doi.org/10468/982.

44. Ralph LeVan, "OpenSearch and SRU: A Continuum of Searching," *Information Technology & Libraries* 25, no. 3 (2013): 151–53, https://napoleon.bc.edu/ojs/index.php/ital/article/view/3346.

45. Dong Nguyen et al., "Federated Search in the Wild: The Combined Power of Over a Hundred Search Engines," in *Proceedings of the 21st ACM International Conference on Information and Knowledge Management* (Maui, Hawaii): ACM Press, 2012): 1874–78, http://dl.acm.org/citation.cfm?id=2398535.

46. B. D. Wilson et al., "Interoperability Using Lightweight Metadata Standards: Service & Data Casting, OpenSearch, OPM Provenance, and Shared SciFlo Workflows," in *AGU Fall Meeting Abstracts* 1 (2011): 1593, http://adsabs.harvard.edu/abs/2011AGUFMIN51C1593W.

47. Library of Congress, "SRU—Search/Retrieve via URL," www.loc.gov/standards/sru.

48. The Library of Congress Network Development and MARC Standards Office, "Z39.50 Maintenance Agency Page," www.loc.gov/z3950/agency.

49. Tony Hammond, "nature.com OpenSearch: A Case Study in OpenSearch and SRU Integration," *D-Lib Magazine* 16, no. 7/8, (2010), http://mirror.dlib.org/dlib/july10/hammond/07hammond.print.html.

50. Daniel Chudnov et al., "Introducing unapi," 2006, http://ir.library.oregonstate.edu/xmlui/handle/1957/2359.

51. Daniel Chudnov and Deborah England, "A New Approach to Library Service Discovery and Resource Delivery," *Serials Librarian* 54, no. 1–2 (2008): 63–69, http://www.tandfonline.com/doi/abs/10.1080/03615260801973448.

52. "News About Our SearchMonkey Program," Yahoo! Search Blog, 2010, http://www.ysearchblog.com/2010/08/17/news-about-our-searchmonkey-program/.

53. Peter Mika, "Microsearch: An Interface for Semantic Search," in *Semantic Search, International Workshop located at the 5th European Semamntic Web Conference (ESWC 2008)* 334 (2008): 79–88, http://CEUR-WS.org/Vol-334/.

54. Sten Govaerts et al., "A Federated Search and Social Recommendation Widget," in *Proceedings of the 2nd International Workshop on Social Recommender Systems* (**[pub info?]**, 2011): 1–8.

55. S. **[first name?]**Reilly, "Digital Object Protocol Specification, Version 1.0," November 12, 2009, http://dorepository.org/documentation/Protocol_Specification.pdf.