

WEB SERVICES: WHAT ARE THEY; HOW DO THEY DO IT; AND WHERE CAN I GET ONE?

Mac Horn

Insight Informatics Pty Ltd.

ABSTRACT

Web Services is a broad-ranging term used for a variety of application to application communications using the World Wide Web. There are a number of different technologies used in the provision of Web Services, primarily based on XML and related technologies.

This paper reviews some current Web Services applications in use in Libraries in Australia, and the underlying technologies. This includes consideration of the emerging standards used in Web Services and an attempt to de-mystify the acronym soup which surrounds these technologies. The paper also considers some of the potential for future developments in Web Services which could provide new and interesting possibilities in the provision of library services and especially in support of direct end user access to information.

INTRODUCTION

The term Web Services has been broadly defined as

“... a new breed of Web application. They are self-contained, self-describing, modular applications that can be published, located, and invoked across the Web. Web services perform functions, which can be anything from simple requests to complicated business processes.” (Gardner, 2001)

Web Services applications are a step onwards from “web-enabled” applications. In the initial phases of providing services over the web, the emphasis has been on designing applications for a human user to browse, search and interact with. It is assumed that the decision-making will be done by a human selecting relevant choices from those on offer. So there is always a human at one end of the transactions. Web Services applications, on the other hand, are designed for direct computer-to-computer interaction, without any direct human involvement in the transactions. There is then a completely different set of requirements for defining and implementing such applications.

The key features of Web Services applications are that they are self-contained and self-describing. It is not necessary to know in advance exactly how a particular Web Services application is going to operate because the application itself provides the required definitions.

In a comprehensive paper from Microsoft, the Web Services architecture is characterised as having five core principles:

“The core principles that have driven the design and implementation of the Web service architecture protocols are as follows:

- Message orientation—using only messages to communicate between services and realizing that messages often have a life beyond a given transmission event.
- Protocol composability—avoiding monoliths through the use of infrastructure protocol building blocks that may be used in nearly any combination.
- Autonomous services—allowing endpoints to be independently built, deployed, managed, versioned, and secured.
- Managed transparency—controlling which aspects of an endpoint are (and are not) visible to external services.
- Protocol-based integration—restricting cross-application coupling to wire artifacts only.” (Cabrera, Kurt & Box, 2004)

The *Message orientation* of Web Services applications means that the applications are based on messages being passed from one system to another. The structure of these messages is designed so that the content is not specific to any particular operating system or process requirements, allowing true platform independence.

The *Protocol Composability* means that each level of the protocols can be used independently, allowing the reuse of components in multiple applications. This is of obvious assistance in simplifying the development and implementation of particular Web Services applications.

Autonomous Services means that Web Services may be developed and deployed independently of the service's consumers. If a new version of a service is developed, it can announce its presence through the use of specific header elements, and only those consumers capable of using the new version can recognise this fact. It also means that the levels of trust can be managed between applications, typically using security tokens of some sort that have been previously agreed between the service and the consumer.

On a similar note, *Managed Transparency* means that within a particular XML message, varying levels of security may be applied, so that the consumer application only has access to those parts of the message that are relevant or authorised. This is explicitly managed by the application to separate the details of the actual implementation from the messages being sent and received through the use of machine-readable descriptions of the messages being sent and received as well as the services offered.

Protocol-based Integration means that it is only the messages being sent and received by the service that are held in common between applications. By using a self-contained system for description and messaging that is devoid of programming language or operating system details, Web Services applications can run in disparate environments and still communicate securely and reliably.

Overall it is the high degree of self-containment that characterises Web Services applications. The message structures are self-describing, the application publishes its own capabilities and requirements, and (at least theoretically) any other application can become a consumer of the services offered. Of course in the real world, a great degree of co-operation is required between the organisations offering and using Web Services applications, especially in the library context where access to proprietary data is a common use of Web Services applications.

TECHNOLOGIES

As with all new computing technologies, the field of Web Services is liberally strewn with acronyms. These include XML, SOAP, UDDI and WSDL.

To better understand these acronyms and what they mean, it may be useful to consider the analogy between Web Services and good old-fashioned postal mail. Both of these are methods for the transmission of information from one party to another, with a known level of reliability and security and with a method of handling communications in both directions.

In the postal context, for information to be sent from one party to another there are some basic components involved. These are the information to be sent, which is written on a piece of paper – the message. Then there is the container in which the message is conveyed – the envelope. On this envelope is the place that the message must be delivered – the address. And finally there is the delivery mechanism – the postal system. All of these have analogous components in the Web Services context.

In this simplistic analogy, in Web Services, the message is written in XML – the eXtensible Markup Language. The envelope is provided by SOAP – the Simple Object Access Protocol. The address on the envelope is also encoded in the SOAP information, but to find the address, the application must look it up in a directory using UDDI – the Universal Description, Discovery, and Integration protocol. Finally the message is sent across the internet using TCP/IP or UDP protocols. The entire process is described for interested parties using WSDL – the Web Services Definition Language.

XML – eXtensible Markup Language

XML, the eXtensible Markup Language, was developed from the SGML general markup language as a means of describing documents in the web environment. XML has formed the basis for a number of

document description protocols, including EAD. More recently, the XML Information Set [XML-Infoset] has been defined through work at the W3C organisation to provide a data model compatible with both text-based data as well as MIME-encoded binary data. This means that an almost infinite variety of data forms may be encoded in XML for transmission across the web. This is an important characteristic for Web Services applications in that the message structure is independent of the encoding method used.

An XML document may consist of a number of elements, but every XML document consists of exactly one document information item and at least one element information item. The entire structure of an XML document can be a combination of different information items such as elements, attributes, namespaces, and comments. Each information item has an associated set of properties that provide a more complete description of the item. Thus an XML document is essentially self-describing in that both the content and the structure are described within the document.

SOAP – Simple Object Access Protocol

SOAP is a lightweight, XML-based protocol for transfer of structured data and type information across a network in a stateless manner. A SOAP message consists of an Envelope, a Header and a Body. The fact that both the message payload and the protocol headers employ the same model can be used to ensure the integrity of headers as well as bodies. Applications may route messages based on the content of both the headers and the data inside the message.

The messaging flexibility provided by SOAP allows services to communicate using a variety of message exchange patterns, satisfying the requirements of distributed applications.

One major advantage of SOAP is that it is defined independently of the underlying transport mechanism in use. It allows the use of many alternative transports for message exchange, and allows both synchronous and asynchronous message transfer and processing. Most current Web Services applications use HTTP over TCP as the main communication protocol. In some applications, to avoid the overhead of TCP, UDP is used instead, but this also reduces the reliability of the transport process.

UDDI – Universal Description, Discovery, and Integration protocol

The Universal Description, Discovery, and Integration protocol, or UDDI, specifies a protocol for querying and updating a common directory of Web Services information. The directory includes information about service providers, the services they host, and the protocols those services implement. The directory also provides mechanisms to add metadata to any registered information.

The UDDI directory approach can be used when Web Services information is stored in well-known locations. Once the directory is located, a series of query requests can be sent to obtain the desired information.

These directories of Web Services applications may be either public or private. Public directories are hosted by large organisations including Microsoft and IBM with their UDDI Business Registry. In addition, many organisations maintain private directories of Web Service applications.

As well as static directories of known Web Services applications stored in well known locations, there is also another possibility in the use of dynamic discovery. In this instance, a Web Services application broadcasts a message to announce its availability and any interested party can respond to this message and gain access to the service.

To stretch the postal analogy a bit further, public directories are similar to the Yellow Pages or other directories of services, while private directories are like your personal address book. In both cases, you can identify the desired service provider by using the structure of the directory (and some serendipity). The dynamic discovery model is more like an advertisement seen in the paper which triggers your interest and so you can, if you choose, respond to the blandishments of the advertiser.

WSDL - Web Services Definition Language

The Web Services Description Language, or WSDL, was the first widely adopted mechanism for describing the basic characteristics of a Web service. WSDL provides an unambiguous description of what both the request and response messages must look like. It is thus the first stage of the automatic definition of the Web Services application.

Using WSDL, applications can be developed to read the structure and content required for communicating with a Web Service, supporting the self-defining characteristics of Web Services applications.

WSDL is the language used to provide the information in Web Services registries as described above. It is thus a standardised format for the directory listing, containing all the information required for a client to address an existing Web Services application.

STANDARDS

One of the key features of Web Services applications is that they are self-defining. This means that once the basic standards required for successful communication are met, Web Services applications can bypass the need for data standards such as MARC21 because as long as the data is defined in the application, it can be in whatever format is most convenient to the application.

This may prove to be both the major benefit and a stumbling block in the development of Web Services applications. For example, it is a relatively simple matter for one library system vendor and one book supplier to agree on message formats for selection and ordering materials, and then develop appropriate applications using these formats. However, other libraries using systems from other vendors may be shut out of this process unless the format is made widely available. On the other hand, defining and approving universal standards is a massively complex and time-consuming process, potentially delaying the implementation of fully functional systems for many years.

It is likely that as with the evolution of other recent library standards like the NISO Circulation Interchange Protocol (NCIP), library system vendors will indeed negotiate private standards that may later become available for public use. This process will almost certainly include a number of protocols that do not reach public acceptance, but the evolutionary principle of survival of the fittest applies equally in the library systems industry as in the primordial swamps. Like NCIP, the public standard will be synthesised from the successful proprietary standards.

To support the transition to Web Services, the Library of Congress has defined MARCXML which provides an XML based structure for the transmission of MARC bibliographic data. This provides a link between an old standard, that of the MARC communications format, and the new technologies of Web Services. Whether this proves to be of long term value will depend on both the evolution of Web Services applications in libraries and also the future of the MARC record structures themselves.

APPLICATIONS

In identifying current Web Services applications in libraries, it is interesting to note how different library system vendors describe their products. I think it is fair to say that in some cases, any application using XML is described as a “Web Services application” in order to satisfy the commercial requirement to have some product that is a “Web Service”. To the purist, it is only a Web Services application if it satisfies the full definitions given above – simply using XML as part of an application does not make it a Web Services application.

There are a number of Web Services applications currently available to Australian libraries from library systems vendors. These include:

Vendor	Application	Description
Insight Informatics	MARCXML export	Provides records from WebOPAC searches in MARCXML format
	Federated search	Supports searching across multiple Libero systems
	Content Enrichment	Provides a link from the Libero WebOPAC to Amazon.com for enriched content of titles in the Library catalogue
Innovative Interfaces	XML Server	Provides bibliographic and other data in XML format from Innovative Millennium systems
	E-Checkin	Supports automatic transmission and receipt of information on e-journal and print issues from serial vendors direct to the library system
	Inventory Express	Supports direct ordering and transmission of receipt information from services such as Baker & Taylor, Amazon and other suppliers
Endeavor	XML Gateway	Supports federated searching of Elsevier Engineering Village 2 databases
EBSCO		Provides a Web Services gateway to EBSCOhost reference databases

This is not meant to be a comprehensive list, but rather a sample of what is currently available. New products are being announced continuously. It is of interest to note that two main categories have emerged – interfaces to library materials suppliers, both book and serial, to improve the timeliness and accuracy of order and receipt data transfer, and federated searching of multiple databases. It is this second area which provides one of the most useful directions for Web Services applications in libraries –overtaking the Z39.50 standard for multiple database searching by a Web Services based federated search model. This is discussed in more detail below.

Among the providers of services to libraries, Amazon.com has been an early adopter of Web Services, offering a free Web Services interface for content enrichment, and actively participating in a number of other Web Services projects to support services to libraries.

In addition to these purely library process-oriented Web Services applications, there is also work being done on Web Services applications to support digital repositories, such as the Australian Research Repositories Online to the World (ARROW) project using the Fedora system. The ARROW project will identify and test software or solutions to support best practice institutional digital repositories comprising e-prints, digital theses and electronic publishing.

Links between library systems and e-learning systems is another area of current development.

Because Web Services applications require both a service provider and a consumer in order to be useful, there is some interest in co-operative development of various Web Services applications. An

informal consortium including many (but not all) of the major library systems and services vendors has recently been created in the United States to this end. VIEWS, which stands for Vendor Initiative for Enabling Web Services, describes itself as “an initiative by vendors and library service organisations to promote interconnectivity among automated systems and services.” The VIEWS consortium has conducted a survey of Web Services applications in libraries, and is planning a co-ordinated development approach. The top three areas identified were:

1. Authentication/Authorisation
2. Routine transactions with financial and e-learning systems
3. Enabling gateway searching of library databases

Other areas of interest included; electronic resource management transactions, patron profile sharing, circulation functions including ILL and NCIP, and EDI and other types of record sharing activities (Grant, 2004)

FUTURE POSSIBILITIES

In looking where Web Services applications may prove useful in the library context, it is clear from existing applications that there are two main areas of development. The first of these starts in the area of federated searching, by which I mean the simultaneous searching of multiple databases. The second main area is in the interactions between libraries and suppliers of library materials to manage and improve the efficiency of the acquisitions process. This includes both the normal purchasing of library materials, and the resource sharing processes of Inter Library Loans.

Federated searching

One of the promises of Web Services applications is that they offer the possibility of searching multiple databases, whether or not they are in standard bibliographic formats, to identify and locate materials of interest to the library user. I am deliberately using the term “materials” to describe the items being sought, since as indicated earlier, the structures of Web Services applications make the physical format of the target material irrelevant to the processes required to identify and locate it.

A typical library user is not especially interested in the format of the materials, containing the desired information, the processes involved in acquiring the desired information or the ultimate source of the information. Basically, a library user has a desire for a specific piece of information, and would like the library staff to get that information. “Now” would be good, “soon” is usually acceptable. The library user may have some interest in the fees that may be charged for access to the required information, but is certainly not at all interested in any indirect costs that they do not have to pay themselves. Even though the user may not be bothered with this level of detail, it is in the best interests of the library to get the required information at the least cost and in the least time. And this is where the promise of Web Services applications comes to the fore.

In the first phase, identifying the desired information, there are a number of parameters that may need to be included:

- Question - what information is required?
- Identification/authentication – who is the user and what sources may be used?
- Location – where is the required information housed/owned/hosted?
- Costs – how much do particular sources cost to use, and how much is the user prepared to pay?

Once the definition of the parameters have been established, it is a relatively simple process to build a Web Services application to process this sort of query. The issue of the *question* is straightforward – it is the information being sought, and can be expressed in terms of keywords, controlled vocabulary phrases or identification numbers such as ISBNs, product codes or whatever is relevant. The *identification and authentication* issues can be easily handled through the layered security inherent in Web Services applications. The *location* of the information is more or less irrelevant, unless it is a particular physical item being sought. The Web Services model would allow at least the identification

of an item irrespective of location. And finally the associated costs can be retrieved as part of the query result.

Behind the scenes, the process is actually more complex. Having received the information and user identification, the initial Web Services application needs to identify possible sources, based on a librarian's input as to the content and access rules for particular databases. It would then use the UDDI protocol to interrogate a Web Services registry to identify appropriate Web Services applications that might provide an answer. Then it is a matter of sending off the XML document containing the query and the authorisation level and awaiting a reply, or, more likely, a set of replies. The level of detail in these replies can be specified as part of the request, so unlike a Z39.50 search, the requesting service can specify a particular level of detail required in the response. Once received, the responses can then be ordered in a useful manner for the user, and presented with appropriate annotation as to costs, relevance and accessibility.

This federated searching model is likely to render simple query-response processes like Z39.50 searching a thing of the past. There are many limitations to Z39.50 searching, including the relative inflexibility of the query structures; the unpredictability of the search translation process; and the limited level of information in the standard response formats. On the other hand, the flexibility of a Web Services based federated search process is only limited by the functionality that the developers choose to include in their products. Of course, in the initial phases of development of federated searching it is likely that vendors will focus on like-systems searching – that is, a federated search across databases of the same structure, such as the vendor's own systems. However, the self-defining nature of a Web Services application means that once the application has been made public, any other Web Services application can readily be developed to match the same searching capabilities. In this way, an informal standard is likely to emerge, to be adopted by other Web Services applications as they are developed. What this standard will contain cannot be predicted now, but this evolutionary approach is the one most likely to succeed.

One potential barrier to the development of federated searching applications across disparate databases, especially those from different database publishers or aggregators, is the commercial requirement of the various publishers to retain differentiators in the services provided. A Web Services application that masks these differences by providing a single search process and an integrated result set would run counter to this requirement. As a consequence, the willingness of database publishers to participate in federated search developments may be very limited. It will be necessary for libraries to encourage their participation in open federated search applications so that the best possible service will be achieved for the libraries and their users.

Acquisitions processes

Once the desired materials have been located and the availability terms discovered, a second phase can be initiated. This can be regarded as the *acquisition* phase, in which a second level of Web Services applications can either retrieve the desired information if it is in digital form, or trigger a request for purchase or loan of the information in physical form. In this phase, the services being sought would include those that manage the supply of materials in either electronic or physical format.

Inter Library Loans

One aspect of acquiring library materials is the ILL process. Once the desired materials have been identified at a particular library, a Web Services application could automatically initiate the ILL request process. This could be as simple as a plain ILL request "form" sent to the supplying library, or, more efficiently, a series of messages to negotiate the ILL, including the request, delivery, payment and return processes. Since the existing ISO ILL protocols are essentially message-based, it would be possible to incorporate a subset of these messages in a Web Services application, encapsulating them as XML documents. Alternatively, a simpler structure could be used, especially in the case of ILL requests between like systems.

By combining federated searching and Web Services ILL processes, it should be possible to create a seamless, patron-initiated ILL system between regional or interest-based library consortia. The key requirement will be the use of compatible systems, but the self-defining nature of Web Services applications should assist in this.

Purchasing processes

As noted earlier, there are a number of existing Web Services applications designed to improve the efficiency of the purchase and delivery of library materials, both electronic and physical. It is worth noting that in these areas, private protocols have been developed for communicating between specific library systems and specific suppliers. This is a reflection on the general failure of older interface standards such as EDIFACT which created too great an overhead for efficient development and deployment of these protocols. By using proprietary protocols, requiring only two or three organisations to agree on the data requirements, the vendors working in this area have been able to create new interfaces quite rapidly. It is likely that a number of these protocols will eventually converge, as different libraries push their system vendors to develop similar interfaces for a wider range of material suppliers.

Other applications

The development areas identified by the VIEWS consortium also offer some promise. It will be interesting to see how the authentication/authorisation Web Services applications develop, since this is one area of increasing importance to many libraries, and one where there are a number of parallel systems being developed for remote authentication. Since managing the trust and security between applications is a key element of Web Services applications, new protocols that support the authentication of library users will be important building blocks for other Web Services applications for libraries. This will be especially important in developing the electronic rights management aspects of digital document delivery.

On the other side of the ledger, it is difficult to see how a Web Services application would be able to work for the retrieval of bibliographic data for copy cataloguing. Since there is a higher degree of intellectual effort involved in accurately identifying the required information, this process can probably best be left in current web-enabled technologies, where a human is required to make decisions based on the data available. Although the MARCXML standard does provide a means of wrapping MARC data in XML documents, this is probably only suitable for delivery of bibliographic data incidental to other Web Services applications such as those in the acquisitions processes described earlier.

CONCLUSIONS

It is quite clear that traditional integrated library systems are pretty much complete. Most library systems available today are very capable at handling normal library operations such as acquisitions, cataloguing, circulation, serials, public access functions and the like. But as libraries look to leverage this investment in technology to provide efficient access to a wider range of library materials, it is in the interactions between the internal library system and the myriad services available outside the library that the most benefits can be obtained. Web Services applications offer a useful model for developing and managing these interactions. However, the range of services that could be used, and the variety of relationships involved in these interactions mean that applications will probably develop faster than the traditional standards processes can move. So for a time there can be expected to be a range of proprietary standards in use, which over time should evolve or converge into a smaller number of more widely used protocols. The range of applications for Web Services is only limited by the imagination and resources of the libraries, library system vendors, and library service providers who get involved in this exciting technology.

BIBLIOGRAPHY

Amazon Web Services <http://www.amazon.com/gp/browse.html/102-4524098-9345733?node=3435361>

ARROW - Australian Research Repositories Online to the World <http://www.arrow.edu.au>

BOSS, Richard (2004) VIEWS Press Release arouses curiosity about Web Services *Information Systems Report*, Vol 3 No. 2 (August 30, 2004).. Excerpt available at http://www.views-consortia.org/views/documents/isr_aug_04_excerpt.pdf

CABRERA, Luis Felipe, KURT, Christopher and BOX, Don. (2004) *An Introduction to the Web Services Architecture and Its Specifications. Version 2.0, October 2004.* <http://msdn.microsoft.com/library/default.asp?url=/library/en-us/dnwebsrv/html/introwsa.asp>

GARDNER, Tracey (2001) An Introduction to Web Services *Ariadne* Issue 29 (2 October 2001) <http://www.ariadne.ac.uk/issue29/gardner/intro.html>

GRANT, Carl (2004) VIEWS Vendor Initiative for Enabling Web Services. Introduction <http://www.views-consortia.org/views/index.shtml>

McDONALD, Diane (2003) *Web Services Technologies Report for the JISC Technology Watch Service* Joint Information Systems Committee, 2003. http://www.jisc.ac.uk/index.cfm?name=techwatch_report_0304