Web Services: Distributed Computing for the New Millennium?

Gareth Cronin
University of Auckland
g.cronin@auckland.ac.nz

The World Wide Web (WWW) has become more than just a source for media-rich information; it has begun to provide "services": self-describing, machine discoverable published resources for processing business information. Web Services are being hailed as a solution to many of the problems of distributed computing and as the future of electronic business. This paper examines the issues involved with the development of Web Services, and the progress of the technologies offered for their implementation so far.

1. Introduction

The World Wide Web (WWW) was originally conceived as a method of publishing text and images, with the level of interaction restricted to retrieval through requests made by a user with a web browser application [1]. The growth of Electronic Commerce (e-commerce), in particular Business-to-Business (B2B) e-commerce has seen the rise of non-user interactions with the WWW, with the Hyper Text Transfer Protocol (HTTP) being used as a vehicle for electronic business transactions without user involvement.

Although it originally had a more generic meaning, the term Web Services has come to mean an internet accessible component that is self-contained, self-describing, queryable, universally interoperable, configurable at run-time and published and located through registries that are themselves Web Services [2].

2. Background

Most pre-Web communications between distributed applications were synchronous. The technologies for providing such distributed communications are complex, and while they function well over a reliable infrastructure such as a local area network, they do not fit well with the dynamically routed and unreliable structure of the Internet. If an application is open to communication from anywhere on the Internet, loads are unpredictable, so there must be some system of queuing and prioritising when loads are heavy [6].

3. The Vision

A Web Service exists at a higher level than a software component. In fact it can be defined as an operating agent paired with software components [3]. In this way a Web Service offers itself to people or machines through an interface that is not static, but can evolve without having to notify its users directly. The self-describing properties of a Web Service allow this loose coupling to take place. This vision for a WWW that allows machine interoperability is often referred to as the "Semantic Web". That is, a web where the data is defined in such a way that machines, not just people are able to interpret data. The World Wide Web Consortium (W3C) is currently working on standards to support this [4].

A vital part of the vision is *discovery*. Discovery is implemented by means of public registries where a Web Service can announce itself. These registries can then be searched by machine to identify an appropriate service for a given task.

As an example, consider booking an airline ticket for a conference. To do this currently, human intervention is required to first find an appropriate online agency from which to purchase the ticket, and then to fill out electronic forms to complete the transaction. With discovery, a user could specify the necessary details, e.g. when and where they wished to travel, and a registry search engine can discover a suitable service and execute the purchase there [1]. Business partners could be discovered and selected automatically based on price and location by entirely machine-driven processes, new partners could be established and trading could begin without any human intervention.

Of course this remote execution requires some sort of remote procedure invocation protocol. The idea of Web Services is to use messaging over existing Internet

protocols such as HTTP and SMTP (Simple Mail Transfer Protocol) [5].

It should be noted that this reliance on existing Internet protocols means that all Web Services will be asynchronous. In general, the Web is not sufficiently robust to cope with synchronous applications and so a messaging-based infrastructure must be used [6].

Personal computer based web browsers are not the only envisaged client devices. Both Sun Microsystems and Microsoft intend to develop Web Services APIs for wireless devices such as Personal Digital Assistants (PDAs) and cellular phones [18][11].

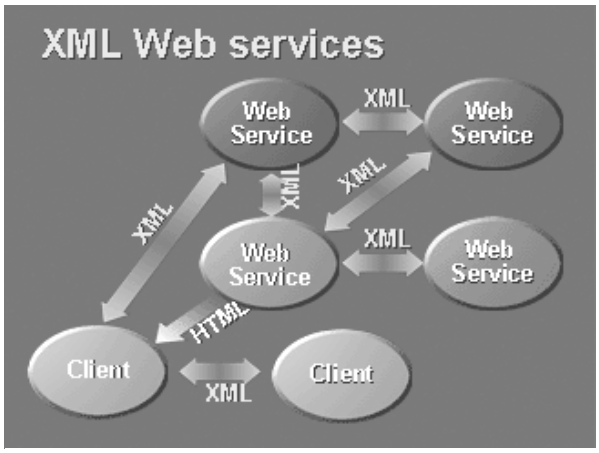

Figure 1 - Microsoft's vision of XML Web Services [18]

4. The Motivation

There are many ways that businesses can take advantage of the Web Services framework. Existing infrastructure can be exposed to the Internet to further take advantage of investment in internal software, e.g. allowing customers to use a parcel tracking service. Existing e-commerce systems can be registered to take advantage of discovery. Web Services can also be used to enable inter-enterprise (B2B) operations, such as integrated supply and inventory between businesses [6].

As mentioned earlier, the automatic and dynamic discovery of business partners opens up many possibilities. A business trading in a commodity could use an agent to continuously search registries to discover the current supplier with the best price and lowest shipping cost combination and immediately being sending that supplier orders. This of course reduces costs by very quickly moving trading to the most appropriate supplier without the need for human intervention.

Although most current writing concentrates on Web Services as an answer to inter-enterprise business, it is easy to see how it could also be used within the enterprise. The increasingly popular portal model can be combined with Web Services to produce enterprise-wide application entry points, glued together as Web Services.

5. Enabling Technologies

To achieve the inter-operable loose coupling that Web Services requires, there must be a set of standards that govern how communication between systems takes place. The likely direction for a language to describe services and to encapsulate messages is the Extensible Markup Language (XML) [17]. Developers using XML can take advantage of a common parser that drastically reduces development time [7]. With the current hype surrounding XML it is hard to believe that there are alternative languages, but one such alternative is the DAML family of languages built around the RDF specifications of the W3C [4]. DAML and RDF deliver advantages over XML such as description logic and inheritance [1].

The likely technology for the transport of XML messages over the Internet is the Simple Object Access Protocol (SOAP) [8]. SOAP is a better way to manage remote invocation over the Internet than distributed technologies such as DCOM, RMI and IIOP because it packages XML messages into “envelopes” and sends them over standard internet protocols such as HTTP and SMTP, taking advantage of well defined data formats and making it easier to operate over firewalls, which are normally already set up to cope with the standard HTTP port 80.

To implement discovery, there needs to be a way to define and query registries of Web Services. Universal Description, Discovery and Integration (UDDI) is such a mechanism. It uses SOAP messaging to publish, edit, browse and search a registry. The Web Services Description Language (WDSL) is an XML standard for describing a Web Service [9].

6. Progress

Two of the world’s leading software companies, Sun Microsystems and Microsoft Corporation already offer implementations of the aforementioned technologies. Microsoft have taken the ideas even further by re-engineering their flagship development platform Visual Studio around Web Services and launching a heavyweight marketing campaign to support it.

Microsoft’s offering is known as Microsoft.NET [18] and Sun’s offering is an extension of its Java 2 Enterprise Edition (J2EE) framework, involving a new set of APIs

and specifications [11]. The idea behind both systems is to reduce the complexity involved in processing XML, load-balancing and handling transactions by providing "containers" that take care of the details, allowing developers to write components that conform to specifications and run inside these containers [5]. It should be emphasised that Sun's approach is a set of standards, whereas Microsoft's approach is a product.

Other large companies such as iPlanet, BEA and Oracle also offer complete Web Services solutions. With this level of support from industry, the future of Web Services seems assured.

However, the availability of development and deployment architectures is not sufficient to realise the complete vision of universal discovery and interoperability, there must also be progress with registries to publish Web Services. The OASIS project is one such initiative that is attempting to gather industry-wide cooperation to develop registry standards [16]. An example of an actual registry that is now operational can be found in Korea [15].

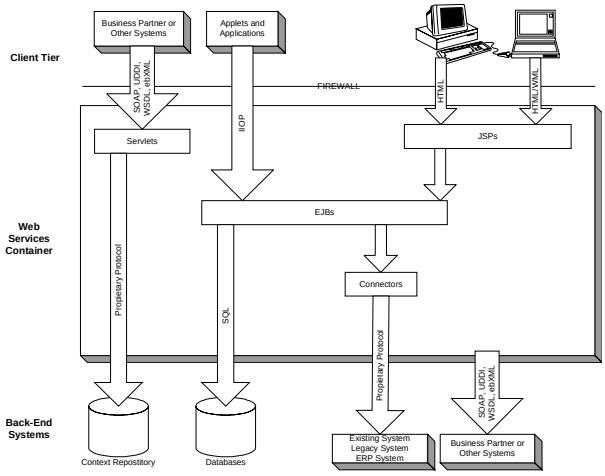

Figure 2 – Sun Microsystems's proposed architecture [5]

7. Problems

Developing messaging-based infrastructures for the Internet can be very complex. The systems must be scalable, and to improve efficiency will probably use caching schemes that must be carefully designed to account for the dynamic nature of Web Services [6]. For Web Services to be taken up by many companies, large numbers of skilled developers are required to cope with this complexity. In the current business environment where there is a substantial shortage of skilled IT workers, this presents a serious difficulty [12].

Cooperation between the industry giants is necessary for the vision of universal interoperability to succeed. The business strategy that has been employed in the past of patenting technologies and not licensing them to other companies to allow competition and further development is contrary to this goal. Although XML, UDDI and WSDL provide standards already, higher-level standards such as those for particular types of business are being developed, and these standards must remain open if they are to provide true interoperability. The United Nations are supporting a suite of XML specifications and behaviours known as ebXML (Electronic Business XML)[14] to address this problem by providing standards for typical business transactions [9][13].

As with software component re-use, there are potential risks with a Web Service falling short of meeting all requirements when used in particular systems. Since it is not really possible to extend the behaviour of a service with inheritance, perhaps more work needs to be done into methods of drawing together multiple services to implement new behaviour, or wrapping existing remote services in locally developed services.

Maintaining Web Services could require all new processes and strategies. If a service ceases to provide functionality that was being used, how will the user of that service be notified? The loss of a particular service could cause loss of quality of service to customers of the business using the service. Protocols for notifying users of updates to Web Services and methods to allow machines to discover and use alternative services in the event of failure must be developed for the business model to succeed.

There is also the human aspect to consider. Making machines responsible for previously human responsibilities raises questions as to whether it is really worth the machine doing the job if a human could already do it adequately, given the lost employment opportunities that this redundancy creates.

Market research in the form of a survey of companies suggests that the grand vision of Web Services, complete with dynamic discovery and interaction will not happen for at least another two years, in the meantime the technologies will more than likely be used solely for more traditional integration purposes [10].

8. Conclusion

Web Services are promising because they draw on best-of-breed technologies and open standards to provide a universally interoperable and ultimately automated system for electronic commerce and other forms of inter-organisational communication.

The promises can only be delivered on if the major software vendors do not take actions that prevent the Web Services technologies remaining as open, unlicensed standards.

We must be aware that despite the seductive notion that Web Services will provide the "ultimate solution" to distributed computing, history has taught us there are no ultimate solutions in Software Engineering [19]. Web Services will no doubt bring with them as yet unforeseen complexities and issues as their popularity grows.

However, it is safe to say that Web Services are here to stay.

References

[1] McIlraith SA, Son TC, Honglei Zeng, "Semantic Web services", *IEEE Intelligent Systems*, vol.16, no.2, pp.46-53. IEEE, USA, March-April 2001.

[2] Portland Patterns Repository Wiki, "WebServices", http://www.c2.com/cgi/wiki?WebServices, 2001.

[3] Clemens Szyperski, "Components and Web Services", *Software Development Magazine*, vol. 9 no.8, USA, August 2001.

[4] W3C, "Semantic Web Activity Statement", http://www.w3.org/2001/sw/Activity, 2001.

[5] Chad Vawter and Ed Roman, "J2EE vs. Microsoft.NET", http://www.theserverside.com/resources/articles/J2EE-vs-DOTNET/article.html, Sun Microsystems, 2001.

[6] Adam Bosworth, "Developing Web Services", *Proceedings 17th International Conference on Data Engineering*. IEEE Comput. Soc, pp.477-81. Los Alamitos, CA, USA, 2001.

[7] Edward Vielmetti, "The (R)evolution of Useful Web Services", *IEEE Communications Magazine*, IEEE, USA, September 1999.

[8] W3C, "Simple Object Access Protocol", http://www.w3.org/TR/SOAP, 2001.

[9] James Kao, "Developer's Guide to Building XML-based Web Services", http://www.theserverside.com/resources/articles/J2EE-s-DOTNET/article.html, Sun Microsystems, 2001.

[10] Dick Kelsey, "Web Services Won't Catch On For Two Years – Jupiter", http://www.newsbytes.com, August 30 2001.

[11] Sun Microsystems. Java 2 Enterprise Edition, http://java.sun.com/j2ee. http://www.sun.com, 2001.

[12] Lawrence A. West and Walter A. Bogumil, "Immigration and the Global IT Work Force", *Communications of the ACM*, Vol. 44, No. 7, New York, USA, July 2001.

[13] Elizabeth Montalbano, "CRN Interview with Simon Phipps – Sun Microsystems", http://www.crn.com/Components/Search/Article.asp?ArticleID=25300, http://www.crn.com, 2001.

[14] ebXML, http://www.ebxml.org, 2001.

[15] ebXML Registry, Korea, http://www.ebxml.or.kr/registry/index.html, 2001.

[16] OASIS, http://www.oasis-open.org, 2001.

[17] XML. http://www.w3.org/XML. 2001.

[18] .NET. http://www.microsoft.com/net/default.asp. 2001.

[19] K.H. Bennett and V.T. Rajlich, "Software Maintenance and Evolution: a Roadmap", *Proceedings of Conference on the Future of Software Engineering*, Limerick, Ireland, June, 2000.