# Web Services and Virtual Communities

**Qingyu Zhang**
*Arkansas State University, USA*

## INTRODUCTION

With the escalating need for global and flexible interactions in an increasingly networked society, virtual communities as a new model are becoming important for virtual communication and collaboration. Virtual community refers to a group of people who interact socially for mutual benefits in a shared cyberspace, and a social network of relationships that provide information and a sense of belongings (de Moor & van den Heuvel, 2004; Schubert & Ginsburg, 2000). Schubert and Ginsburg (2000) describe virtual communities as the union between individuals or organizations who share common values and interests using electronic media to regularly communicate within a shared semantic space.

From a technical perspective, the communicative and collaborative functionalities of virtual communities mostly consist of standard components, modules and tools (Ludwid & Kluber, 2003). These software components and functionalities increasingly come in a form of Web services discovered and invoked via the Internet (de Moor & van den Heuvel, 2004). Virtual community can be realized by using Web services to take advantage of the convergence of software technologies (e.g., distributed computing, object orientation, WWW) and network technologies (e.g., voice/data service, IP, Internet) (Muschamp, 2004). Web services are interesting for virtual communities, since they allow community members, even non-technical ones, to combine Web services in new value-adding applications (de Moor & van den Heuvel, 2004). In the following sections, we are going to discuss what Web services are, why Web services are important and where they are going. In addition, we look at Web service architecture, standards, advantages, development platforms and semantic Web.

## WHAT ARE WEB SERVICES?

The World Wide Web Consortium (W3C) developed the following definition for a Web service:

*A Web service is a software system designed to support interoperable machine-to-machine interaction over a network. It has an interface described in a machine-processable format (specifically WSDL). Other systems interact with the Web service in a manner prescribed by its description using SOAP messages, typically conveyed using HTTP with an XML serialization in conjunction with other Web-related standards.*

Web service is a self-describing, self-contained, modular application accessible over the Web. It provides a modular capability to combine, decouple and recombine software components to create virtual applications in an ad hoc, real-time manner (Iyer, Freedman, Gaynor, & Wyner, 2004; Lim & Wen, 2003). The advantage of Web service is the ease of integration, and thus, Web services will be fundamental to flexible systems integration over the Internet. Instead of building an application as a collection of tightly coupled components, the Web services approach is much more dynamic. Like the WWW and a browser, Web services use simple, open standards to combine the loosely coupled components to achieve ubiquitously deployed infrastructure capabilities.

## WHY ARE WEB SERVICES IMPORTANT?

For decades, the IT industry has been trying to address the fundamental challenges of distributed computing: locating and accessing remote components (Sleeper, 2001; Wang, Huang, Qu, & Xiem, 2004; Graham et al., 2005). The big issue is the interoperability, the ability to communicate and share data and information with software from different vendors and platforms. As shown in Table 1, major shifts have occurred toward flexibility and interoperability through open and widely accepted standards (Graham et al., 2005).

Web services were designed to tackle the problem of heterogeneous sources integration and make heterogeneous systems interoperable in distributed computing (Wang et al., 2004). In the 1990s, many companies and organizations—including the Object Management Group (OMG), Microsoft, Sun Microsystems and IBM—began developing their own technologies to enable communications among distributed components. OMG's Common Object Request Broker Architecture (CORBA), Microsoft's Distributed Component Object Model (DCOM), Sun Microsystem's Remote Method

*Table 1. Major shifts/evolutions of interoperability*

| Age | Major shifts/evolutions of interoperability | Platforms/Major technologies |
|---|---|---|
| 1970 | Networking | TCP/IP (Client-server computing) |
| 1990 | Open portable user interface | HTML/HTTP (World Wide Web) |
| 1995 | Open portable programming | Java |
| 1996 | Open portable data exchange | XML |
| 2000 | Open dynamic integration | Web services |

Invocation over Internet Inter-Orb Protocol (RMI/IIOP) and IBM's Distributed System Object Model (DSOM) allow programs written in various languages and executing in different locations to communicate.

However, the interoperability is limited among these technologies. For example, DCOM and CORBA cannot communicate easily. This has impeded distributed computing from facilitating system integration and business process automation. In 2000, Microsoft coined the term "Web services" when the company introduced Web services as a key component of its .NET framework. Web services, an emerging technology, adopt open standards (e.g., XML and HTTP). Web services are loosely coupled, reusable software components that semantically encapsulate discrete functionality and are distributed and programmatically accessible over standard Internet protocols (Sleeper, 2001).

Web services technologies represent the next stage in distributed computing. Previous generations of distributed computation environments did not display the flexibility as Web services do. CORBA, DCOM and RMI are based on the RPC paradigm, with tight coupling between what the client sends and what the server expects. Web services enable application-to-application integration over the Internet to address the challenges of distributed computing and business-to-business (B2B) integration.

## WEB SERVICES ARCHITECTURE

Web service is based on service-oriented architecture (SOA). In an SOA, all software components are modeled as services to be consumed over a network (Elfatatry & Layzell, 2004; Gottschalk, Graham, Kreger, & Snell, 2002). The focus of design is the service's interface. Components with well-defined interfaces allow applications to be more flexible, mix-and-match, loosely coupled and dynamically composed.

Figure 1 depicts the Web services architecture with three roles (service provider, service requestor and service broker) and three operations (publish, find and bind). Web services are based on generally accepted open standards. These primary standards and technologies include:

- Extensible markup language (XML), with a platform-independent format for data encoding and exchange
- Simple object access protocol (SOAP) for describing the message structure and embodying the client-server relationship between the service requestor and the service provider.
- Web services description language (WSDL) to describe the application programming interface (API) detailing the interface and explaining how to use the service for service registration or service advertisement.
- Universal description, discovery and integration (UDDI) to register a service so others can discover and locate it.

XML is a platform-independent language capable of describing data with customized elements and tags. Since XML documents are text based and use a standard character encoding, they can be processed on any platform (Muschamp, 2003; Ferris & Farrell, 2003). SOAP defines the structure of messages. A SOAP message is an XML document that describes particular actions to invoke on a remote application and enables programs on separate computers to interact across a network. SOAP relies on underlying transport protocols such as HTTP and SMTP to transfer messages between computers (Hundling & Weske, 2003).

WSDL is an XML document that provides the information, in a standardized format, about the functionalities of the Web service; what data is required for requesting the

*Figure 1. Web services architecture*