An OPEN

BY STANS KLEIJNEN and SRIKANTH RAJU
SUN MICROSYSTEMS, INC.

Web Services architecture

The name of the game is web services—sophisticated network software designed to bring us what we need, when we need it, through any device we choose. We are getting closer to this ideal, as in recent years the client/server model has evolved into web-based computing, which is now evolving into the web services model (see Figure 1). In this article, I will discuss Sun Microsystems' take on web services, specifically Sun ONE: an open, standards-based web services framework. I'll share with you Sun's decision-making rationales regarding web services, and discuss directions we are moving in.

One of the basic questions in developer's minds when starting to write Web Services is—what language to use? While it is possible to write web services in any language including Perl, Fortran etc., we strongly believe that the Java™ Programming Language is the most appropriate because of its ability to scale from the tiniest of JavaCards all the way up to the mighty Enterprise Edition. [More comparisons of Java Platform vs. other platforms are

discussed later]. With the Java Platform, programmers can readily define operations and place that logic on virtually any computer system, regardless of the underlying hardware architecture or operating system—a distinct advantage in the diverse world of web services.

A sophisticated Java-based distributed enterprise application will typically use the Java 2 Enterprise Edition (J2EE), have all data secured in a directory server, business logic implemented in the application server, and presentation logic implemented and exposed through the web server (see Figure 2). On the client side, a desktop-based Java client will use the Java 2 Standard Edition (J2SE), and a wireless client will use the Java 2 Micro Edition (J2ME).

Complex systems worldwide have been built on the Java 2 platform. The Brazilian National Healthcare system has deployed a Java-based solution to 20,000 clinics serving over 12 million patients. These applications run E10K servers using sophisticated Java-based security implementation. Information is fed through a hierarchy of regional

An OPEN Web Services architecture

servers from hospitals, clinics, and pharmacies. Every patient-clinician interaction is recorded in the system, including X-rays, EKGs, endoscopic video exams, and prescription information. Some data is entered manually, but much is streamed directly from instruments. This cost-effective system has paid for itself several times over. Drug fraud has been virtually eliminated, as physicians have access to all patient information via the patient medical

ID card. The existing infrastructure is being upgraded to replace this ID card with JavaCards. Another current initiative is the replacement of non-desktop materials with J2ME-enabled cell phones and wireless PDAs. Because the application was architected from the ground up using the Java platform, it lent itself to improvisation using web services, and many other client side technology advancements using J2ME and JavaCard based APIs.

An important new capability is the Application Verification Toolkit, which verifies applications using the J2EE standard are not hampered by proprietary extensions that lock them out of platform portability. Anyone can download the application programming interfaces and source code of the Java platform; to date there have been almost five million downloads by interested companies and developers. This openness fosters innovation and increased productivity, as developers learn the Java platform from the inside out. Figure 3 illustrates how J2EE in the containers empowers web service development. As shown here, the J2EE platform now comes standard with interfaces to the delivery and integration layers. The standards shown are named "Java API for XML <something>." Following is a more complete list than shown in the figure:

JAX-RPC	RPC programming with SOAP
JAXM	Message orient middleware, using JMS
JAXB	Bind the contents of an XML message to the methods of a Java Bean
JAXP	Process the low-level contents of an XML

Web Application

JAXR	message using SAX or DOM Access the UDDI or ebXML registry to advertise or discover a service.
JMS	Java messaging service.
JDBC	Java database access APIs

Figure 4 illustrates how these technologies can be used in web services. In this diagram, a developer tool is accessing a UDDI registry to see what services are available.

The registry could be public or a private enterprise registry. The tool browses the registry by sending XML messages in the SOAP protocol. Within the registry is the description of the available services, where an entry written in WSDL describes the services and interfaces. The web service is usually built from components, such as Enterprise Java Bean (EJB) components wrapped in XML/SOAP, or it could be a legacy application with an XML/SOAP wrapper. The legacy case will probably be the most prevalent web service in the near future.

SUN ONE: AN OPEN, STANDARDS-BASED WEB SERVICES FRAMEWORK

In order to meet the needs of secure, always-on, mission-critical, and task-specific services, known as Services on Demand, services must offer more than simple, specific functions such as weather or traffic information. The real benefits emerge when services combine and recombine to satisfy user needs spontaneously. Services on Demand represents a continuum of technology, as illustrated in Figure

5. Today, we have the emerging infrastructure for basic web services, which provide loosely coupled integration between applications. In the future we will enhance web services with ubiquitous identity and contextual awareness, and introduce additional service-based technologies, such as Jini for dynamic configuration systems, and JXTA for peer-to-peer and grid computing.

Sun ONE is an open framework that supports today's web services and lays the foundation for the future. It

provides a comprehensive, standards-based, integratable computing model that lets organizations create and deploy Services on Demand. Figure 6 illustrates the Sun ONE architecture. The lower portion of the diagram depicts the OS, hardware, storage, and networking platform, which includes the directory technology to define users, subscribers, organizations, and policy. The upper portion depicts tools for creating, assembling, deploying, and testing services. In the center are the three layers familiar

An **OPEN** Web Services architecture

to web application programmers: presentation, business, and back-end data access logic. Presentation is what a portal server focuses on, delivering services to any device, aggregating content, and providing security, personalization, and knowledge management. We also included in the service delivery box the ability to deliver Java applications to mobile devices, as well as an application framework for rapid development of web applications and services.

The Service Container, where web services run, is typically a J2EE application server. Inside the Service Container are pre-built web services an enterprise may build or buy, often hosted by an existing commerce or communications application. These services and the Service Container collectively deal with the business logic.

Service Integration is about integrating to other enterprises, legacy applications, databases, and message-oriented middleware.

The generic services stack maps into an integrated, products-based stack, or Integrated Stack (see Figure 7), which completes the overall framework for the creation, deployment, and access of next generation Services on Demand. The Sun ONE platform is comprised of many industry standards, as Figure 8 illustrates. Please note the standards shown in italics are not yet complete.

Sun has been a leader in the open standards community for over 20 years. Many of the popular Internet standards, such as TCP/IP, Java, XML, and LDAP were invented or co-invented by Sun, or we were early adopters and evangelizers. Twenty years ago, Sun was the first vendor to include an integrated TCP/IP stack in every box we shipped. That was radical in 1982, but seems pretty obvious now.

BUILDING WEB SERVICES WITH SUN ONE

In October of 2002, Sun introduced the Sun ONE Ap-

Services on Demand

FIG 5

Services on Demand represent a continuum of technology

Service Creation, Assembly and Deployment

FIG 6

The Sun ONE architecture.

plication Server version 7.0. This key ingredient of the Sun ONE platform is tightly integrated with the new Sun ONE Studio for Java 2 Enterprise Edition 4.1 set of web services tools. Sun also introduced a new business model that offers the core version (Platform Edition) free to enterprises and independent software vendors (ISVs) on all leading platforms. With its new modular architecture, the application server increases options, as well as return on investment (ROI), in rapidly building and deploying Sun ONE Java web services.

The Solaris 9 platform has added hundreds of features, making it one of the best platforms to run Sun ONE. The Sun ONE directory server is embedded into Solaris 9 (di-

rectory optionally installs during OS installation), and the application server is in the process of being embedded. Solaris can now take advantage of LDAP services, and will eventually migrate users away from the NIS and NIS+ systems. In addition, the first fully integrated Sun ONE platform, called Sun ONE Developer Platform, is now available in a technology preview edition. It is bundled with the Sun ONE Application Server 7.0, and also includes the directory server, identity server, a UDDI registry product, and a new release of the Portal Server 6.0. The Sun ONE platform designs and builds a web service in 10 steps:

1. The Sun ONE Studio IDE creates and deploys a web service using its web service plug-in and the Java Web Service Developer Pack (JWSDP). The IDE:
 - Creates the code for the Enterprise Java Bean (EJB) components that implement the server side business logic;
 - Deploys the EJB components to the application server; and
 - Creates a simple GUI client that requests access to the web service.

2. Within the IDE, the application can be created and deployed as a session EJB component to the Sun ONE application server. Then the web service wrapper can be created and deployed to the SOAP server, which may act as a facade to the EJB service.

3. The web service WSDL is generated from within the IDE.

4. The WSDL is published with a UDDI registry, which can later be browsed.

An **OPEN** Web Services architecture

5. From the GUI Client, the web service's access methods can be invoked.

6. User credentials can be extracted from the SOAP header, and used to transparently authenticate and authorize the user. There are web service hooks (SOAP header processors) for both authentication and authorization. This makes it possible to use virtually any security product or scheme to manage access and policy.

7. When the SOAP request header is processed, it triggers the metering and monitoring web services. When the service invocation is complete, a signal is sent to the client, which can be used for statistical measurements such as elapsed time.

8. Much of the web service functionality can be enabled and extended through the Sun ONE Integration Server, comprised of design- and run-time components and tools. Using a graphical interface, web service developers can re-use existing infrastructure to create new web services. At the heart of the Sun ONE Integration Server is a powerful process engine that can manage process execution and state for well over a million run-time business processes.

9. Sun ONE Directory Server, Access Management Edition can be used to provide directory services to validate user authentication and authorization credentials, and access policies. This software offers policy-based authorization, in conjunction with role-based authentication and can be used in conjunction with the Sun ONE Portal Server to create segmented user communities.

10. One can use the soon-to-be-released version of the Sun ONE Identity Server with a Liberty

A usage scenario of ebXML standards.

Project implementation (discussed in the next section) for single sign-on and federated identity and security management.

PHASES IN WEB SERVICE ADOPTION

Business organizations will likely take a phased approach to web services adoption, as summarized in Table 1. Today, in phase one, we have the web and application servers, and enough web service standards to build and deploy basic web services. (Note the Sun ONE Integration Server has been using XML web services long before that term was coined.)

Phase two, the preparation for which is almost complete, begins serious use of web services for enterprise integration. In this phase, the enterprise still exerts tight control. Users, such as employees and business partners, are defined in the enterprise directory, and not discovered dynamically. Enterprises connect to established business partners that agree on which web services to use, and what service levels are expected. In this phase, we begin to enable some middleware into Solaris 9, bring Java standards into the web service world, and provide new tools for creating, assembling and deploying web services. We also have a UDDI registry product built on the Sun ONE Directory Server, which has also been enabled with feder-

Table 1

Phases of web service adoption.

Phase one	Web applications and basic XML services infrastructure
Phase two	Web services for enterprise internal integration, including B2E and enterprise-managed, bilateral B2B
Phase three	Web services extended externally from the enterprise for dynamic B2C and B2B

ated identity standards from the Liberty Alliance Project.

The Liberty Alliance Project was initiated by Sun and other industry players to create standards for an open model, in which multiple identity providers compete to store portions of user identity information. Network identity includes not only username and password, but data such as government issued IDs, biometric information like retinal scans, and context-type information, such as open appointments in your business calendar, preferences, education history, and financial records. Liberty's federated approach allows multiple identity schemes to interoperate. For example, AOL has its own identity schemes, known internally as magic carpet. These 35 million magic carpet entries can interoperate with Liberty. Table 2 lists some of the identity-enabled Sun ONE Platform Products.

The Liberty Alliance has over 80 members, representing two types of organizations: technology providers, such as Sun, HP, Nokia, and VeriSign; and companies with large user databases, such as United and American airlines, major credit card companies, large financial institutions, and many large wireless telephone service providers, such as Vodafone. Together, these companies have over two billion trusted user relationships.

ebXML

In phase three, the ultimate phase, customers and business partners will find and conduct business dynamically. To reach this phase, two key pieces of technology are required: federated identity services, such as from the Liberty Alliance, and public service registries designed with standards for real-world B2B, such as ebXML. In this phase, expect the completion of a public version of the UDDI registry server, the specification and availability of federated services, and an ebXML roadmap.

The goal of the ebXML standardization effort—which is driven by two UN organizations: The Organization for Advancement of Structured Information Standards (OASIS) and The United Nations Center for Trade Facilita-

Table 2

A sampling of the identity-enabled Sun ONE platform products.

Directory server <ul style="list-style-type: none"> • LDAP directory • XML, DSML2, LDAPv3 • massive scale, performance, replication • multi-platform support • LDAP proxy (separate package as directory proxy server) 	Identity server <ul style="list-style-type: none"> • Web SSO • authentication, authorization, audit • federated identity, privacy • self-administration/registration, centralized
Meta-directory <ul style="list-style-type: none"> • publish, synchronize, join across databases, applications and directories • provision accounts • connectors for AD, Oracle, NT Domains 	Certificate server <ul style="list-style-type: none"> • create/publish/revoke X.509v3 certificate • PKCS standards compliance • registration/certification authority • OCSP • FIPS compliance

tion and Electronic Business (UN/CEFACT), as well as over 2,000 members—is to build an open marketplace framework in which any business regardless of size can participate in a global electronic marketplace.

Some ask why we need yet more standards when we already have SOAP, WSDL, and UDDI. But these standards do not provide all that's necessary for ad-hoc electronic business transactions. The closest to ebXML from the functionality standpoint is the old Electronic Data Interface (EDI), which is too heavy for many smaller business organizations. ebXML standardizes business processes such as purchasing, ordering, shipping, and payment, so they can be performed by machines without manual pre-configuration. It also allows business partners to choose quality of service in their message delivery. Secure and highly reliable message delivery is vital in many business transactions, for example in ordering \$500 million worth of Korean merchandise, or executing a buy order of 10,000 Sun Microsystems shares. Basic SOAP does not address these security and reliability requirements. The Sun ONE Platform will soon include an ebXML appliance geared toward small businesses. Figure 9 illustrates a B2B

An **OPEN** Web Services architecture

usage scenario of ebXML standards in industry, where the following occurs:

1. Company A checks the ebXML registry to see what types of business processes are defined.
2. Once it finds the business process it wants, such as “purchase order,” it builds and deploys its own application that captures the semantics of that business process.
3. Company A submits to the ebXML registry its business profile, which describes its business processes and its ebXML capabilities and constraints.
4. Company B discovers business processes supported by company A in the ebXML registry.
5. Company B indicates its interest in engaging business transactions with company A.

DIFFERENT APPROACHES TO WEB SERVICES

It does seem like every major technology company has an architecture for Web services. While there is some common ground around standards such as XML, UDDI, and SOAP, there are also important differences—the most notable being between the Java Platform and Microsoft .NET. On the one hand, we have an open, industry-wide community that works together to continually extend the potential of the network with compatible, vendor-neutral technologies. Also the application programming interfaces of the Java Platform—as well as the source code are available to any one via download. To date, there have been almost 5 million downloads of all the APIs and source code of the Java platform by interested companies and developers. This openness allows developers to learn the Java platform from the inside and allows them to become productive more quickly. It also fosters innovation because developers can actually see the source code that determines how the software works.

The following chart (Table 3) summarizes the differences between building Web Services using the Java Platform versus building using the .NET platform.

WIRELESS WEB SERVICES

In addition to all the aforementioned strengths of J2EE, one of the key markets of tomorrow is the Wireless market, where Java technology-enabled devices from handset manufacturers such as Nokia, Ericsson, Motorola, Sharp among others, are being deployed by key carriers such as Nextel, NTT Docomo and LG Telecom. This market

Table 3		
A summary of differences between the Java and .NET platforms.		
Feature	J2EE	.NET
type of technology	standard	product
middleware vendors	30+	Microsoft
interpreter	JRE	CLR
dynamic web pages	JSP™	ASP.NET
middle-tier components	EJB	.NET managed components
database access	JDBC, SQL/J	ADO.NET
SOAP, WSDL, UDDI	yes	yes
implicit middleware (load-balancing, etc.)	yes	yes

is so huge that there are about 1 billion Java enabled handsets to be available in the next 2 or 3 years, and these devices—mobile phones, pagers, PDAs, automotive systems are all joining the traditional client, the personal computer, on the Net. They represent new platforms—always on, always with you—that are opening up exciting new opportunities.

With a wide range of options beyond the PC, individuals will use the device most convenient for them at that time—which means developers will need to design their service for delivery in a variety of forms. Java Technology facilitates this new usage model because it provides the only open, cross-platform and cross-device application and service platform.

Finally, the key here is interoperability—web services won't work without it. With Java—indeed, with all of Sun ONE, thanks to the open, integratable design—users are free to choose the products that best suit their needs. Q

STANS KLEIJNEN (Stans.Kleijnen@sun.com) is Vice President of Market Development Engineering at Sun Microsystems, Inc.

SRIKANTH RAJU (Srikanth.Raju@Sun.com) is a Staff Engineer/Technology Evangelist at Sun/Microsystems.

BIO