Web Services 2002

Market Milestone Report

Contents

DELPHI® GROUP

www.delphigroup.com

Fever Pitch

The most passionate supporters of Web services have proclaimed it as "bigger than the Internet" as well as the answer to every CIO's prayers and the one technology to finally bridge the gap between IT and business. Others have taken a more pragmatic view, casting Web services as over-hyped, overblown, and perhaps just one more excuse to sell new software. The truth is that both may be right. Web services offer nearly limitless potential to change the way software is deployed and managed, yet the near-term reality will undoubtedly fail to meet the lofty expectations set by its most feverent proponents.

Regardless of the steadfast positions held by both extremes, Web services hold the promise to solve many, often recurring, business problems. From delivering information between applications to dynamic building of business processes within disparate environments to enabling the cost-effective reuse of exisiting technology assets, Web services will play a large role in making organizations more efficient. Certainly, there are still many issues to be resolved including, but not limited to, interoperability, security, and revenue models, just to name a few—but the future of Web services is bright.

What are Web Services?

A basic definition of Web services is the building of composite applications using a standard set of network services to find and link loosely-coupled software components. While this definition may seem myopic in the face of the full potential Web services has to offer, it gets to the heart of what indeed they are about—a module programming architecture based on specific standards.

At a more granular level, a Web service is an XML object comprised of content, application code, process logic, or any combination of these, that can be accessed over any TCP/IP network using the SOAP standard for integration, the WSDL standard for self-description, and the UDDI standard for registry and discovery within a public or private directory.

What differentiates Web services from other past and current approaches, however, is a level of software reusability that provides for not only faster development of new applications, but also the ability to transform computing infrastructure from a cost center into a profit center (for example, by delivering software resources as a service to partners and customers). To this end, Delphi defines Web services as "software-based business assets that can be shared, combined, used, and reused by heterogeneous computing resources within an organization or between firms."

Understanding Software as Services

In order to understand how Web services will affect enterprises, it is imperative to understand the function of software as services. A services-based framework implies that every Web service brings together a number of software components and manages their relationships without the need to know the specifics (i.e. underlying language) of each component. Since only the business-level services are searched for and used within a Web services framework, a business service-based directory can be used.

If you are wondering exactly what a service-based directory is, don't fret. There are a number of issues that users must first understand before undertaking an evaluation of the Web services market.

Standards

Conceptually, Web services will deliver on the promise of interoperability—the ability for components created in different programming languages to work together as if they were created using the same language. This interoperability will be provided through industry standards that have been emerging throughout the past few years. The holy trinity of Web services standards is SOAP (Simple Object Access Protocol), WSDL (Web services Definition Language), and UDDI (Universal Description, Discovery, and Integration). There are several other emerging standards of which to be aware, among them XML, and all of which deal with transacting data from disparate systems over the Web. Please see the sidebar on pages 5 and 6 for a more detailed discussion of Web services standards.

The standards effort in Web services is led by a consortium of companies. There are several standards organizations out there that count among their members leading organizations that are trying to pave the way towards acceptance of common standards. Several of these standards organizations are doing critical work in the area of Web services. Among them are:

- WS-I—promoting Web services interoperability across platforms, operating systems, and programming languages;
- OMG—producing and maintaining computer industry specifications for interoperable enterprise applications;
- W3C—developing interoperable technologies to exploit the Web as a forum for information, commerce, communication, and collective understanding;
- OASIS—designing and developing industry standard specifications for interoperability based on XML.

The work of these organizations is not to be overlooked. The critics of Web services point to true interoperability as the biggest issue holding back Web services. For technology providers to provide real interoperability that will have a business impact, these standards organizations will have to gain traction with leading vendors and enterprises.

Required Services Beyond Standards

Standards are certainly important to the adoption of Web services technology. There are, however, several other issues that decision-makers will need to pay keen attention to as they evolve Web services initiatives for their enterprises.

Development

Before a complete discussion of Web services development can begin, it is necessary to make the critical decision as to the framework for a development environment. Today, an enterprise can choose between the J2EE and .NET frameworks. Much has been written and debated about the "enterprise wars" between these competing frameworks. In Delphi's view, the major advantage to .NET is that Web services are architected in from the ground up, whereas APIs are retrofitted to allow XML communication for the J2EE framework. Microsoft has no competitors with the .NET framework, though, while J2EE vendors are focused on keeping products interoperable and non-proprietary—this could be an important consideration down the road should Microsoft be seen as having a monopoly.

Integrated Development Environment

Before a Web service can exist, it needs to be created. Each Web service is created in a development environment, typically referred to as an integrated development environment, or IDE. For the .NET framework, the IDE is Visual Studio.NET. Several vendors provide IDEs for the J2EE framework, including Sun Microsystems with the Forte product.

What the IDE does is provide custom modules to create applications or services within a given platform. Thus, Visual Studio.NET is the tool for developers to create services within the .NET framework, while Forte is one of several choices for doing the same within the J2EE framework.

Web services are created in an IDE like VisualStudio.Net, Forte, or Eclipse by IBM and then deployed on an application server like BEA's WebLogic, IBM's Websphere or Fujitsu's Interstage. In order to use a Web service once it has been created, a user needs to know that it exists and be able to find it.

Discovery

Web services will enable the dynamic discovery and use of applications on the Web. For discovery to occur, software will search UDDI directories, examine specifications of Web services, and construct applications from a composite of services that are found in these directories. No application will need to be hard-coded to use other applications, rather the software will understand the need to find and use Web services dynamically—making for a more flexible software infrastructure.

How exactly will this discovery happen? One way will be to utilize USML (UDDI Search Markup Language), an XML-based protocol for carrying a search request, including multiple queries, key words, UDDI sources, and aggregation operators. Another method will be to use Web services Inspection language, an XML format for assisting in the inspection of a site for available services and a set of rules for how inspection related information should be made available for consumption.

Dynamic Discovery

One of the major technological benefits of Web services with respect to discovery will be the enabling of dynamic discovery. Dynamic discovery can refer to the exposure of a collection of services that live together under the same URL. Such service exposure is called dynamic because the number and types of services in the collection existing at the endpoint can change over time, but clients need only know the one URL in order to access any or all of them despite any changes to the collection. Alternatively, dynamic discovery can refer to systems in which clients search through registries to first discover and then invoke services supporting the capabilities they require. Such systems are described as dynamic because the clients supposedly have no prior knowledge of the services they're searching for.

Dynamic discovery is not as easy as it sounds. Clients and services must share common definitions of the concepts involved in the Web service that might be shared. A lack of that shared information could mean that a Web service would fail to interact with other services in the way it is

intended. As standards evolve, they will begin to define those common concepts. One technology that shows promise in helping to organize and facilitate discovery of Web Services is the use of business rules to define selection criteria that can be updated to meet the current needs of the business.

Dynamic Binding

A basic definition of Web services is the building of composite applications by linking together as loosely-coupled (i.e., not explicitly bound or "hard wired") software components. The action of building these links is known as "binding" and may come in the form of "static binding" whereby the two components on either side of the link are explicitly identified in advance; or "dynamic binding" either component being linked is discovered during the process of compiling the composite application. Dynamic binding offers the "third leg" of Web services architecture, which allows for components to be discovered, bound, then published. The standard descriptions for Web services which allow them to be dynamically bound are defined using WSDL and UDDI, defined and discussed throughout this document.

Discovering and binding Web services will be one challenge; Guaranteeing the security of Web services will be another major challenge.

Security

The issue of security is high on the list of issues for any network-based resources or initiatives, Web services being no exception. In the context of Web services, security is viewed from three perspectives: Identity Management, Authentication, and Confidentiality.

Identity Management addresses the issue of "who" is using "what" resources as delivered by Web services. For example, if a Web service I access is retrieving information from another system (i.e., delivered as a Web service) then it would be problematic for me to go through the process of signing-on to that or any systems. Instead, I should have transparent access and let the Web service worry about passing on information about my identity. This is what initiatives such as Microsoft Passport and Sun Liberty handle (see Current Security Initiatives section below).

Authentication provides a guarantee that the sender of a message in a Web service is who they claim to be and provides a guarantee that the message is what it is intended to be and has not been tampered with. It is imperative that Web services provide authentication of both the sender and receiver of the message in a Web service interaction. Authentication is another area where some companies are defining business rules to control authorization levels and criteria for accessing and using Web Services.

Confidentiality provides a guarantee that information meant for a particular party is not visible to unauthorized parties. Typically, confidentiality has been achieved by encrypting messages with keys. In the Web services world, automatic encryption and decryption will be a must-have in any solution. As Web services are effectively content of one form or another, ensuring the confidentiality of this content is an important issue for any network-based service.

Current Security Initiatives

Several initiatives to address security issues in the Web services realm head-on are underway. Microsoft's Passport and Sun's Liberty Alliance are the most recognizable of these initiatives. Both initiatives deal with the concept of federated authentication—the process of utilizing a single-sign-on (SSO) to access multiple Internet Web sites instead of requiring the user to establish an account at each site.

Microsoft Passport

Microsoft .NET Passport provides users with Single Sign-On (SSO) capabilities as a Web services resource. Passport users can create a single sign-in name and password for use across participating .NET Passport sites. The SSO service provides a common Internet authentication mechanism across participating Web sites. The SSI service also

SOAP (Simple Object Access Protocol)

A protocol for exchange of information in a decentralized, distributed environment. It is an XML-based protocol that consists of three parts—an envelope that defines a framework for describing what is in a message and how to process it; a set of encoding rules for expressing instances of application-defined datatypes; and a convention for representing remote procedure calls and responses. SOAP is a key standard for delivering Web services.

WSDL (Web Services Definition Language)

An XML format for describing network services as a set of endpoints operating on messages containing either document-oriented or procedure-oriented information.

UDDI (Universal Description, Discovery, and Integration)

A platform-independent, open framework for describing services, discovering businesses, and integrating business services using the Internet.

XML (Extensible Markup Language)

A form of self-describing data that creates common information formats in order to share both the format and the data across the Internet, intranets, and other networks. XML frees Internet content from the browser, making it available to real applications.

WSFL (Web Services Flow Language)

An XML language for the description of Web services compositions. WSFL considers two types of Web services compositions. The first type specifies the appropriate usage pattern of a collection of Web services, in such a way that the resulting composition describes how to achieve a particular business goal; typically, the result is a description of a business process. The second type specifies the interaction pattern of a collection of Web services; in this case, the result is a description of the overall partner interactions.

ebXML

A modular suite of specifications that enables enterprises of any size and in any geographical location to conduct business over the Internet. Using ebXML, companies now have a standard method to exchange business messages, conduct trading relationships, communicate data in common terms and define and register business processes.

WSRP (Web Services for Remote Portals)

Visual, user-facing web services-centric components that plug-n-play with portals or other intermediary web applications that aggregate content or applications from different sources. WSRP defines a web services interface description using WSDL and all the semantics and behavior that web services and consuming applications must comply with in order to be pluggable as well as the meta-information that has to be provided when publishing WSRP services into UDDI directories. The standard allows WSRP services to be implemented in very different ways, be it as a Java/J2EE based web service, a web service implemented on Microsoft's .NET platform or a portlet published as a WSRP Service by a portal. The standard enables use of generic adapter code to plug in any WSRP service into intermediary applications rather than requiring specific proxy code.

WSRP services are WSIA component services built on standard technologies including SOAP, UDDI, and WSDL. WSRP adds several context elements including user profile, information about the client device, locale and desired markup language passed to them in SOAP requests. A set of operations and contracts are defined that enable WSRP plug-n-play.

WSIA (Web Services for Interactive Applications)

Formerly known as WSCM (Web services Component Model), the purpose of the OASIS Web services for Interactive Applications (WSIA) is to create an XML and Web services centric framework for interactive web applications; harmonize WSIA as far as practical with existing web application programming models, with the work of the W3C, emerging Web services standards, and with the work of other appropriate business information bodies; ensure that WSIA applications can be deployed on any tier on the network and remain target device and output markup neutral; and promote WSIA to the status of an international standard for the conduct of XML and Web services based web application development, deployment and management.

RosettaNet

RosettaNet is a non-profit consortium of more than 400 of the world's leading Information Technology (IT), Electronic Components (EC), Semiconductor Manufacturing (SM) and Solution Provider (SP) companies working to create, implement and promote open e-business process standards.

RosettaNet is named after the Rosetta Stone, which, carved with the same message in three languages, led to the understanding of hieroglyphics. RosettaNet, like the Stone, is breaking language barriers and making history.

By establishing a common language—or standard processes for the electronic sharing of business information—RosettaNet opens the lines of communication and a world of opportunities for everyone involved in the supplying and buying of today's technologies. Businesses that offer the tools and services to help implement RosettaNet processes gain exposure and business relationships.

Companies that adopt RosettaNet standards engage in dynamic, flexible trading-partner relationships, reduce costs and raise productivity. End users enjoy speed and uniformity in purchasing practices.

RosettaNet Partner Interface Processes "PIPs" are specialized system-to-system XML-based dialogs that define business processes between trading partners. Each PIP specification includes a business document with the vocabulary, and a business process with the choreography of the message dialog. PIPs apply to the following core processes: Administration; Partner, Product and Service Review; Product Introduction; Order Management; Inventory Management; Marketing Information Management; Service and Support; and Manufacturing.

The RNIF Core Specification provides exchange protocols for quick and efficient implementation of RosettaNet standards. The RNIF specifies information exchange between trading-partner servers using XML, covering the transport, routing and packaging; security; signals; and trading partner agreement.

WSCI (Web Services Choreography Interface)

WSCI describes the flow of messages exchanged by a Web service in a particular process, and also describes the collective message exchange among interacting Web services. One of the key benefits of WSCI is that it bridges the gap between business process management and Web services by describing how a Web service can be used as part of a larger, more complex business process. As of the publishing of this research paper, the WSCI standard was in a public review period. After the public review period is complete WSCI will be submitted, on a royalty-free basis, to an industry standards body.

allows users to avoid repetitive data entry by storing a limited set of basic demographic information that can be shared with participating .NET Passport sites when signing into those sites.

Passport users can also create a wallet that stores their billing and shipping information in a secured location. Consumers can then make online purchases at participating .NET Passport express purchase sites by signing in to their wallet and sending their purchase information to the merchant with one click and without the need to retype any information. .NET Passport uses encryption technologies, such as Secure Socket Layer (SSL) and the Triple Data Encryption Standard (3DES) algorithm, for data protection.

Liberty Alliance

Sun Microsystem's Liberty Alliance was set up in response to Microsoft's Passport, sharing the same goals of managing user identities and providing authentication resources to used in connection with Web services. The technologies that make up the Liberty Alliance will have benefits on both the vendor/corporation side as well as the consumer side. For vendors and corporations, these technologies will assist in verifying that users are who they claim to be. For consumers, these technologies will ease the process of logging into secure public Web sites, applications, and any network resource created with Web services authentication in mind.

The Liberty Alliance Project is a business alliance formed to deliver and support an identity solution for the Internet that enables single sign-on for consumers as well as business users in an open, federated way.

The view of Alliance members is that federated identity will facilitate federated commerce. Thus, a person's online identity, their personal profile, personalized online configurations, buying habits and history, and shopping preferences are administered by the users themselves, yet securely shared with the organizations that the user chooses. A federated identity model will allow every business or user to manage their own data, while ensuring that the use of critical personal information is managed and distributed by the appropriate parties, rather than a central authority.

The Liberty Alliance's role is to support the development, deployment, and evolution of an open, interoperable standard for network identity. The stated goals of the Liberty Alliance Project are:

- To allow individual consumers and businesses to maintain personal information securely;
- To provide a universal open standard for single sign-on with decentralized authentication and open authorization from multiple providers;
- To provide an open standard for network identity spanning all network devices.

Web Services Security Specification (WS-Security)

In April of 2002, Microsoft, IBM, and VeriSign submitted a proposed Web services security specification, entitled WS-Security at the time, to OASIS. While still in its earliest stages of evolution into a standard, it's now referred to as the Web services security specification under OASIS.

The Web services security specification defines a standard set of SOAP extensions, or message headers, that can be used to implement integrity and confidentiality in Web services applications. The new Web services security specification will support security mechanisms of several types, each using implementation and language-neutral XML formats defined by XML Schema. Some of these include: use of XML signatures to provide SOAP message integrity for Web services; use of XML encryption to provide SOAP message confidentiality for Web services; attaching and/or referencing security tokens in headers of SOAP messages; carrying security information for potentially multiple, designated participants; associating signatures with security tokens; representing specific forms of binary security tokens as defined in the original WS-Security specification.

Other Initiatives

In addition to the vendor-driven initiatives of .NET Passport, Liberty Alliance, and WS-Security, a number of public domain approaches have been proposed, notably SAML (Security Assertion Markup Language) and XACML (eXtensible Access Control Markup Language). Additional information on these two standards can be found on the OASIS Web site (www.oasis-open.org).

Lastly, another Microsoft-driven standard is XKMS (XML Key Management Specification). The XKMS initiative is led by Microsoft and VeriSign and designed to integrate public key infrastructure (PKI) and digital certificates (which are used for securing Internet transactions) with XML applications. The concept is to delegate the signature processing to a "trust server" on the Web, so that thin clients don't require the innate ability to do this themselves. XKMS relies on the XML Signature specification already being worked on by W3C (one of the standards organizations discussed earlier) and on anticipated work at W3C on an XML encryption specification.

If a Web service can be created, discovered, and secured, then the next step is to use it to complete a task or a process.

Orchestration

As with all technology segments, the Web services segment has evolved its own vernacular. When it comes to utilizing services, the common term is orchestration. That term works well, as it brings up the context of a symphony, where different instruments must work together in harmony.

Relation to Process

Typically, Web service orchestration occurs in the context of a process, whether that process involves the simple exchange of data between two applications or is a complicated stock transaction deal. The act of orchestration involves defining and executing the logic and rules that assemble multiple synchronous and asynchronous web services into a long-lasting multi-step business process. Thus, once a service has been created, it needs to be consumed in some way. Orchestration defines how a service will be consumed.

Business Versus Technology

As with every other piece of enterprise software, there is a technological side to orchestration and a business side, too. On the technology side, there are Web services orchestration specifications that exist; two of the better known are Web services Flow Language (WSFL) and eXtensible LANGuage (XLANG).

WSFL is an XML language for the description of Web services compositions. It considers two types of compositions. First, flow models specify the appropriate usage pattern of a collection of Web services, in a way that describes how to achieve a particular business goal. Second, global models specify the interaction patterns of a collection of Web services, with the result being a description of the overall partner interactions.

XLANG is a notation for the specification of message exchange behavior among participating Web services. It is widely thought that XLANG will serve as the basis for automated protocol engines that can track the state of process instances and help enforce standard protocol in message flows. At its core, XLANG is an XML business process language that provides a way to orchestrate applications and XML Web services into larger-scale, federated applications by enabling developers to aggregate large applications as components in a long-lived business process. XLANG relates to WSDL in two ways. First, an XLANG service description is a WSDL service description with an extension element that describes the behavior of the service as a part of a business process. Second, XLANG service behavior may also rely on simple WSDL services as providers of basic functionality for the implementation of the business process.

On the business side, the concept of Business Services Orchestration (BSO) wraps the process context around the orchestration of services—BSO

is a process-driven method for managing business services created both internally and externally. A BSO solution would make use of an orchestration specification, such as WSFL, XLANG or potentially through the recently announced standard WSCI (see glossary on pages 5-6).

As Illustrated within the diagram below, a typical instance of BSO would involve steps in a process invoking specific Web services from a bank of available resources. The process engine would capture state information, and determine when to invoke or destroy short-lived services, perhaps through the use of company-defined business rules. Orchestrating Web services does not end the challenge of managing them. It is imperative that organizations also make sure that their solutions cover transaction support.

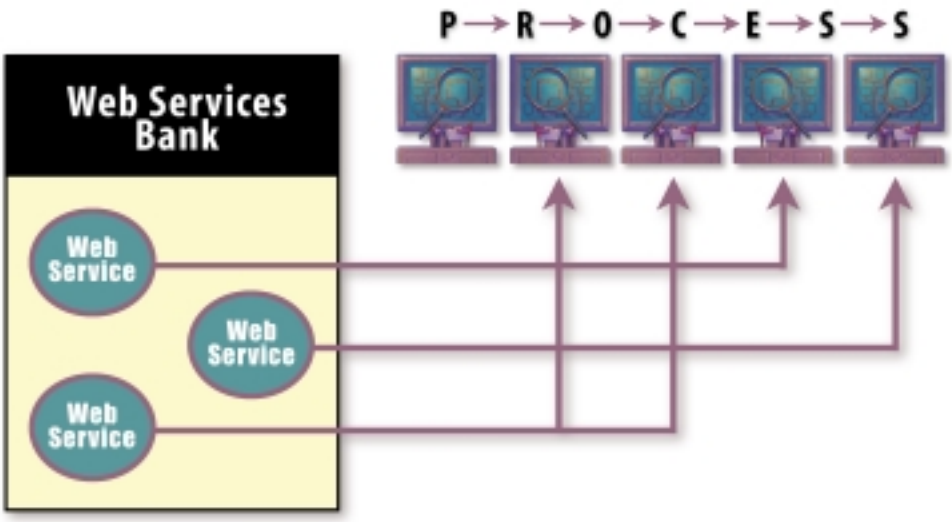
Transaction Support

Any organization uses complex business processes to tie in with disparate applications from the many departments in the enterprise. These organizations are interested in learning how Web services can work within a business process along with transaction support, routing of messages, etc. The concern is that most of the Web services platforms available today provide a quick way of Web service-enabling existing applications and invoking that functionality. Business processes however automate a set of operations that the organization performs. These operations could encompass several discrete Web services, invoke other applications, and perform transactions all as part of the business process. Hence it is important for an organization to understand how Web services would seamlessly fit into previously defined business processes.

Thus, applications will need to be able to transact with each other so that information can be changed by one and shared back to another. One example is a Web service that does the function of credit card validation. After validating a credit card number, the balances of two accounts need to be changed. Dollars must be debited from one account and credited to another. Should something happen within the process that is not right, the process needs to be rolled back so that dollars are not created or lost along the way.

Operations Management

Like any other technology within an enterprise, Web services must be both reliable and manageable. A Web service is considered reliable if it works in the way that it is intended to work. Any messages passed back and forth during a Web service lifecycle should be delivered to the correct recipients, in the correct amount of time, in the correct order, and the correct number of iterations. If a message were to be incorrectly delivered for any reason, then there must be functionality that alerts both the sender and recipient to the circumstances.



The important issues to consider with regard to reliability are guaranteed delivery, non-repudiation, and "once-and-once-only delivery." A guarantee that a message is sent over the Internet as part of a Web services interaction and that it reaches the intended recipient is an absolute necessity. Time-outs and resends need to be handled automatically and transparently. Guaranteed

delivery also means that an application can make a request to send a message and not have to write state-keeping routines that check whether a message has reached its intended destination with the intended results.

With guaranteed delivery in place (and security, of course), it is time of focus on non-repudiation. This means guaranteeing that a recipient cannot repudiate a message that the recipient received or a sender cannot repudiate a sent message. Non-repudiating adds a sense of accountability to Web services. It is also necessary to ensure that a message that is intended to be delivered only once is not delivered in perpetuity, but rather only the one time it is intended to be delivered.

As Web services are deployed throughout an enterprise, managing access to them, tracking their usage, logging the services performed, potentially billing the users for the service—these become challenges that will need to be solved. Real-time knowledge of the state of a Web service is required and rules for system behavior based on the possible states must be incorporated into system operations. The importance of tracking increases if messages are queued asynchronously for delivery. Organizations will also need continuous visibility into the performance metrics of Web services requests and replies, endpoint states, and message delays. Every event happening at the requester end, the provider end and in-between while invoking a Web service has to be logged and easily visible to the managers of the Web service. In addition, when deploying Web services, organizations will need to pay close attention to user roles—giving different users various levels of access to different Web services.

Commercial Interchange Models

The hype surrounding Web services has been partially driven by the promise that Web services will fundamentally change business models. In the short term, Web services will serve as a tool to optimize business operations. The business models of financial services institutions and pharmaceutical firms are not going to change overnight due to Web services. There are, however, interesting business models that will grow around the concept of Web services.

Several niche vendors are addressing market needs that are evolving as Web services gain traction. For example, there are small firms developing capabilities for monitoring services available in UDDI registries. Eventually, these firms will offer services for monitoring availability and inventory of services, usage monitoring, and billing services. Other firms are tracking services in order to provide a guaranteed delivery service, in essence becoming the Fedex of the Web services world. And, in time, Web services brokerages will open for business, creating a market for services that are developed by companies that don't have the resources to market and sell their own services.

Global Grid

As Web services become more a reality over the next few years, computing power will gradually become more of an issue. Grid computing is a potential source of that computing power. The Global Grid is a worldwide project to organize distributed processing by creating a grid of networked computers using the Internet and Web services. The goal is to build a processing grid that subscribers can plug into and use as needed, just as we do today with the electrical power grid.

The most prominent example of grid computing applicability is the SETI project, where a network of computers lend processing power during individual PC downtime to the search for extraterrestrial life. Delphi research shows that a significant amount of enterprises (42%) are not familiar with the Global Grid. In contrast, though, 37.5% of organizations view the Global Grid will be important or imperative to their Web services strategy. The continuing globalization of business is one of the primary reasons the availability of a worldwide computing utility is viewed as being so important. Another key driver is the desire to minimize costs associated with the acquisition and

use of computing power. Web services and the Global Grid present a platform from which to respond to these market drivers.

Web Services Networks

Web services networks (WSN) bring practicality to the deployment of Web services. They provide the infrastructure and services that the requesters and providers of Web services need to conduct business. These services include non-repudiation of messages, guaranteed delivery, "once-and-once only" delivery of messages, encryption of messages, and authentication. In addition, Web services networks also provide a registry of categorized Web services available to meet the needs of requesters and providers, thus providing a framework to search for available Web services. WSNs also provide the ability to manage application-to-application interactions.

Web Services in Action

Web services promise to make the lives of businesspeople easier. The best way to understand how Web services will do this is to analyze an example. One of the best examples of the impact Web services could have is on the corporate portal.

The portal is the desktop to the businessperson—a single, unified interface to the various pieces of information necessary to conduct business. A user can order supplies, make travel plans, check their calendar, and send off an email to a colleague all through the portal. What has typically made this possible is a "portlet," which allows information from different applications or Web sites to appear in windows within the portal. In the portlet model, the user is interacting directly with an application or a Web site, but just happens to be doing so in the portal interface.

The Web services paradigm profoundly affects how the user interacts with the information and Web sites, with minimal impact to the user. Take a businessperson using his/her portal interface to make a flight reservation. In the portal model of the last few years, the user might see the Travelocity interface in one window of the portal. The user would then use that interface to make flight plans. With Web services, though, the portal is not forced to simply display the Travelocity content in a window, or portlet. The Travelocity capabilities can be delivered as a service. In this model, the user would simply see their portal interface, type in their travel preferences, and that information would travel as a service to Travelocity, which would then deliver back the reservations as a service. The user is not required to use Travelocity interfaces or even to understand that Travelocity is providing the service.

This is just one of a myriad number of potential Web services examples. In this example, a whole process (Travelocity reservation) is wrapped as a Web service and invoked in another process (corporate travel reservation). As more and more services, both simple and composite, are developed and publicly registered, the business opportunities for leveraging those services will increase exponentially.

Where Web Services Can Impact the Enterprise

As with all emerging technology paradigms, Web services must prove their benefit to the organization. Web services will benefit both the business and IT sides of the organization—and in the process, help to bridge the communication gap that so often exists between business managers and IT managers.

Business Benefits

On the business side, Web services will facilitate "on-the-fly" application development, process optimization, and "codeless" development. Business managers will have the components of applications and processes at their fingertips for the purposes of creating applications that will help them do their jobs more efficiently. Gains in process efficiency will become increasingly

evident as managers become accustomed to using Web services as a tool to improve process performance. All of this will be available to business people without having to possess any programming knowledge. Suddenly, business people will be able to perform functions that they used to have to rely on IT managers for. In many cases, this is where the communication breakdown between business and IT occurs. Web services will play a large role in smoothing that communication.

As Web services begin to gain traction in IT departments, enterprises can expect several business benefits:

- Improved operational efficiency through inter-application communication;
- Lower operational costs due to the use of existing infrastructure and the minimal infrastructure needs of Web services;
- Improved integration of the value chain, both internally and externally, because of the ease of exposing internal processes to trusted business partners;
- Improved organizational flexibility (and thus responsiveness and innovation) through the exposure of services to new and existing channels.

Technology Benefits

IT executives will also reap benefits from Web services. Web services will help enterprises to capitalize on component architectures in a way that has heretofore not been possible. Most large organizations spent the 1990s transitioning from deep legacy software to some degree of adoption of enterprise applications. These packages promised to solve a variety of linked business problems in a single package, primarily in the area of transaction management and coordination. As organizations found out, however, deploying such software often involved expensive and tim e-consuming customization. Conversely, the component architecture of Web services will help to eliminate some of the time and expense associated with application development.

Within any enterprise, it is increasingly necessary for applications to "talk" with each other. Web services will provide the IT executive with language-independent connectivity between applications. The IT executive will especially like the ability to wrap legacy applications in a standardized, consistent, and reusable format—doing so will allow for improved utilization of existing infrastructure investments.

Finally, Web services will help to eliminate one level of integration by enabling applications to expose functionality directly from the machine on which that application exists, rather than having to add additional infrastructure in the form of servers required by middleware solutions. This will have positive budgetary implications for IT executives.

The confusion around Web services will begin to dissipate as consumers learn more about issues such as discovery and orchestration and become more secure in the business benefits that can result. Already, organizations are beginning to test the waters of Web services and move up the learning curve. Delphi's recent survey of over 250 organizations revealed an interesting picture of the Web services market—from the end-user point of view.

The Software Buyers' Perspective

The Web services market is evolving quickly—the pace of change can lead to confusion for software buyer. Delphi recently conducted a survey of software buyers from approximately 250 leading organizations. The perspective of these decision-makers points out some of the leading challenges and opportunities ahead in this market.

Clearly, confusion exists in the minds of software buyers as to exactly what Web services are. There have been many definitions propagated in the

marketplace, creating a obvious sense of confusion. There are signs, however, that the "fog" is beginning to lift. Many buyers recognize the standards involved with Web services, with SOAP being the most recognizable, followed by UDDI and WSDL. This understanding of the importance of standards points to the fact that software buyers comprehend that Web services need to deliver interoperability of diverse systems to provide real benefits.

Show and Tell

Many cynics point to a lack of case studies as a huge weakness of Web services technology—they claim that Web services are not truly being used in enterprises today. This is not entirely correct. A significant amount (72%) of survey respondents indicated that someone in their organization is working with Web services technology now. In many cases, though, it can be an overstatement to say that Web services are truly being employed. XML is the Web services technology with which most are working. Very few organizations have hands-on experience with UDDI or WSDL at this point—typically, companies are first using SOAP to experiment behind the firewall.

Of those organizations that are not yet using Web services, 57% indicated that experimentation would begin once the benefits of Web services had been made clear to them and once experience-based best practices were captured and shared by other organizations. Another 57% of respondents noted that software tools facilitating the construction of Web services must be generally available before they will begin to work with the technology. Interestingly, while respondents want to wait until tools have been on the market for some time, they are also willing to work with point tools offered by Web services specialists and do not necessarily require a comprehensive platform offering. This attitude points to an opportunity for niche vendors to gain traction with products that are point solutions for users.

Web Services Strategies

Software buyers, while not necessarily knowing everything there is to know about Web services, certainly understand the impact they can have on business—80% of respondents said that Web services are important or imperative to their business strategy. This group understands that Web services will not only change the way that computing resources are built and used, but that it will also enable new business models.

One thing that the last decade has taught software buyers is to have a strategy for new technology paradigms. Over a third (35%) of respondents indicated that a Web services strategy is currently in place within the enterprise. Another 26% will have a strategy formulated within six months. Only 13% foresee strategy development being more than a year down the road. Organizations are predicting that Web services will light a fire quickly, and that they need to be ready when that happens.

Within the enterprise, it is typically the CIO who will function as the Web services strategy leader. The CIO is a logical choice considering that this is typically the individual best positioned to combine both the technical and business understanding of Web services needed to set strategy and to oversee execution.

There still exists a group of organizations (15% of respondents) that view Web services (and all technology, for that matter), as the domain of IT staff other than the CIO. These organizations are behind the learning curve, having failed to realize the importance of bringing together IT and business domain experience. The Web services technology sector is one at the direct intersection of business and technology.

Standards Recognition

XML is clearly the foundational technology upon which Web services will rest. The standard was nearly a unanimous choice, indicating that it is a "must have" component of Web services. In fact,

53% of organizations are processing XML-based information with their business applications. Of those not yet working with XML, the majority (54%) expect to begin doing so within the next six months.

For many organizations, the conversion of legacy information, especially content, into XML is a daunting task that may slow their ability to shift to a Web services computing model. While content stored in any format may be wrapped in XML to create an information object, there is a large performance difference between packaging content in XML and representing it natively in XML. The lesson to enterprises is that critical, frequently used content must be converted to XML as a first step in Web services adoption, and all new content should be captured in XML format.

Typically, XML will be used for multiple purposes. Naturally, XML will be used to describe content and document structure with the language (its original purpose as intended by the W3C). Beyond just describing structure, though, early adopters of XML technologies have found additional ways to utilize the language: to control content presentation through the description of graphical templates, which enables a write-once, present-many content strategy (especially relevant to the concept of accessing Web services on mobile devices); and for the integration of data between internal applications, and between internal and external systems (accomplished by wrapping information with XML code to create self-describing packets).

XML has been in the market's collective consciousness far longer than SOAP, WSDL, or UDDI. Thus, software buyers tend to be more comfortable with XML in concept than with the other Web services specifications, though their understanding of these standards are increasing. SOAP is widely viewed (64% of respondents) as an important component of the Web services infrastructure, whereas UDDI and WSDL were not seen as such a high priority. Most organizations will gain initial Web services experience by deploying them internally to integrate applications, and this can be done using only XML and SOAP. UDDI and WSDL will be useful later within large, multidivisional companies, but do not become as essential until Web services are shared outside the firewall.

In terms of organizations preparing for application interoperability, approximately 20% of respondents have begun to fit their applications with SOAP interfaces, while another 52% plan to begin this process within one year's time. While SOAP is gaining mindshare and momentum within organizations, UDDI is just at the beginning of the learning curvw as only 8% of respondents are official members of the project. UDDI is a technology that is "on the horizon" for companies-with 23% of respondents definitely planning to leverage UDDI and 25% closely monitoring its development.

Grid Applicability

As discussed earlier in the section on grid computing and the Global Grid, one of the features made possible by the Web services architecture is the ability to break large computing tasks into smaller components, each of which can be sent to another machine for processing, then returned to the original computer. The most public example of this model is the SETI@home project (Search for Extraterrestrial Intelligence). This effort to find extra-terrestrial life, coordinated by the University of California-Berkeley, uses the otherwise idle time of volunteers' desktop computers around the world to process cosmic data gathered by observatory equipment.

Of the organizations surveyed by Delphi, 14% are currently using Web services for distributed data processing and nearly one-third plan to begin doing so within the next year—these numbers point to a recognition of the potential that distributed computing holds within the enterprise marketplace..

That 42% of respondents are not familiar with the Global Grid speaks to the fact that we are at the

beginning of the adoption cycle for distributed computing; full-scale adoption is most likely three years away, at best. Consistent with a technology in its early stages of adoption, a significant number of respondents are aware of the Global Grid and its potential impact on their business—37.5% of respondents said that the Global Grid will be important to imperative to their Web services strategy. The continuing globalization of business is one of the primary reasons the availability of a worldwide computing utility is viewed as being so important. Another key driver is the desire to minimize costs associated with the acquisition and use of computing power. Web services and the Global Grid present a platform from which to respond to these market drivers.

Enterprise Deployment Environment

Much has been made of the enterprise wars: J2EE versus .NET. Software buyers will make the ultimate decision in that battle; so far, they are hedging their bets. The largest group of respondents (41%) is preparing for a mixed environment, meaning they will be able to utilize either J2EE or .NET. Microsoft's penetration into the enterprise can certainly be felt, though, with 36% of respondents indicating that they will prepare to deploy Web services in a Microsoft-centric environment (compared to 16% that have committed to J2EE). The enterprise wars are not over; software buyers realize that and are preparing for the future in a way that leverages their existing investments.

Applicability of Web Services

Web services are widely viewed as a means to extend existing investments in information repositories, applications, and business processes throughout organizations and across extended value chains. One of the many questions not yet answered in the Web services realm is exactly where Web services will impact the enterprise.

Over two-thirds of respondents (70%) have a business portal project underway. It makes sense that these organizations will use Web services to integrate information and applications into these portals, rather than building hard-wired portlets that must be recoded each time an application or information source changes. Additionally, portal deployments are reaching more frequently across the firewall. As this trend continues, the business portal will become a place where individuals build, publish, access, and use Web services.

The concept of dynamic object integration is not new to many individuals managing unstructured enterprise content. Most content management applications are adept at creating virtual documents on the fly by assembling disparate content chunks into a single framework. Web services provide a standards-based means to broadcast, aggregate, and use content, which will replace the proprietary methods currently used by most content management vendors.

Another large group of respondents (39%) will concentrate their Web services efforts on application-to-application integration. These respondents have struggled with proprietary EAI solutions and need to develop and implement standards-based integration services that will allow them to quickly bind applications internally and to share them with external constituents.

Nearly half of the survey respondents have a BPM project underway in their organization. Process integration is the next focal point for those companies that have successfully integrated content and applications in their computing environment (often through a business portal). As more organizations codify and modify their internal processes and those shared with external value chain members, they will use Web services to extend XML representations of those processes and to integrate them with content and applications, providing best practice business context to corporate information. As noted in Delphi's 2002 BPM Market Milestone Report, Web services will nicely complement BPM technology.

Inside-Out Approach

Technology projects have tended to start out as small pilot initiatives before gaining widespread acceptance within the enterprise. Web services, as a technology sector, is following that same pattern. Two-thirds of respondents will deploy Web services initially for internal application integration purposes. Half of these projects will be confined to one physical location, but an equal number will involve multiple locations. Organizations will use Web services internally first, especially as they wait to learn how security issues evolve.

Applying Web services internally first makes good sense, because these initiatives are simpler than projects involving external constituents. It is far easier to build and maintain consensus around internal efforts, which translates into improved ability to fund and control projects. It won't be long, though, before organizations will be hungry to reap the potential savings that can come from using Web services outside the firewall.

Necessary Resources

In any technology deployment, an organization needs talented resources to optimize results. The majority of respondents (77%) indicated that they will leverage internal resources to deploy Web services. This is appropriate if they have the requisite skills in-house, since the cost of internal resources is generally less than that of contracted skills. Fifty-five percent of organizations realize that they need more experience with service-oriented architectures, XML, and other Web services technologies and thus will rely on business partners or third-party service providers to implement projects. For the most part, organizations will use a combination of internal resources and third-party experts—this allows the organization to get its people educated on all the technological issues that go along with Web services.

Perceived Obstacles

Software buyers note four hurdles to successful Web services deployments. While no one obstacle is seen as prohibitive by a large number of respondents, there were four hurdles that were cited more frequently than the others.

Foremost among these is inexperience in architecting services. Few respondents (5%) see a lack of trained developers as the most significant problem, but developers cannot do their job well unless the Web services they are building have a clear and efficient architecture. It is probable that organizations will seek external help in architecting Web services while using internal development resources to engineer and build them.

Multiple standards implementation methods are also viewed as problematic. Standards are merely technical specifications that have been agreed upon by a governing organization or by the market itself. There is no mandate to implement the specification in a prescribed manner. Without a clear architecture that represents established and proven best practices in place, there are many choices that must be made by a developer of Web services (or applications). As with any technology, early adopters of Web services will learn how to architect them by traveling the hard path of trial and error, and followers will learn through the experiences of those that have gone before them.

Two of the largest obstacles to Web services deployment efforts are issues that can derail any IT project. It is impossible to receive project funding in today's economy unless the initiative demonstrates a relatively significant and rapid return on investment. And it is far easier to build the technology than it is to get people to use it. Web services are all about the collaborative sharing of resources to reach a business goal, but many corporate cultures are still attuned to hoarding and protecting information and business processes, especially when dealing with other organizations. Individuals and firms must

indoctrinate themselves in this new collaborative mindset first if they are to leverage Web services successfully and reap meaningful return on investment (ROI).

Web Services Ownership

Over 40% of initial Web services projects will be funded by senior executives. Most respondents (80%) had indicated previously that Web services are important if not imperative to their organization's business strategy; therefore, it is only natural that sponsorship would come from senior management. Also, most respondents had indicated that their initial Web services deployment will be an internal one and the level of intra-organizational cooperation necessary to develop and deploy enterprise scale Web services requires high-level leadership. Visible, active leadership from senior executives will be even more critical for initial attempts at inter-organizational integration through Web services.

In nearly half (48%) of respondents' organizations, the CIO, IT Manager, or Line of Business Manager will fund the first Web services project. Sponsorship by IT and Line of Business managers is best suited to skunk-works projects and pilot deployments linking a few workgroups or departments. As Web services become more pervasive within these organizations and deployments become mission critical, CIOs and other senior executives will play a more active role in the efforts. As noted previously, the CIO is the person tasked with bridging the business and technical sides of the organization, so it is natural that they fund many strategic Web services projects.

Spending Habits

The largest group of respondents (23%) expects to spend less than $100,000 on Web services projects within the next three years, and the next largest group (18%) anticipates spending less than $250,000 during the same time period. Spending is generally proportional to the size of an IT project, so one must assume that these groups envision starting with small Web services initiatives. However, one of the promises of Web services is the extension of existing IT assets as opposed to wholesale replacement of systems. In many instances, deploying Web services will not force new investments in hardware or applications and will require only limited spending on XML and SOAP coding.

There is a correlation between the size of respondents' companies, as expressed in terms of revenue, and their spending plans for Web services. In general, the larger the company, the more they plan to invest in Web services. On the other hand, there appears to be little correlation between the level of planned spending and the strategic importance of Web services expressed by respondents.

IT history teaches us that most projects come in over budget, often because not all the costs were identified and planned for up front. In the case of Web services development and deployment, it is easy to overlook necessary initial expenditures such as developer training and third-party architectural services, as well as reoccurring costs like the organization and maintenance of Web services components. Many firms find that their original projections for Web services spending were overly optimistic or simply ill informed.

Metrics

Every technology is judged on its ability to deliver benefits, either through cost reductions or revenue generation.

Respondents are evenly divided as to their financial goals for Web services. Equally sized groups believe that Web services will be a necessary cost of doing business that provide no measurable revenue upside, that Web services will be revenue generators that make a positive contribution to the firm's profitability, and that Web services will be a break even proposition.

The equality of the split in expectations underscores the lack of familiarity that many organizations have with service delivery. Without similar experiences to rely on, many companies have no way of knowing what to expect in terms of financial payback from Web services. That two-thirds of respondents are anticipating Web services will be cost negative or neutral at best matches well with the perception that proving ROI will be a significant obstacle to Web services implementation.

Twice as many respondents (31%) are looking for Web services to reduce the cost of integrating internal applications than for any other benefit. Current, proprietary EAI methods are very expensive because of the amount of time required to write and modify connectors that are specific to one pair of applications. Web services allow an application to integrate with any other by passing XML-based business objects between them using a standard transfer protocol (SOAP). The cost of building a standards-based one-to-many data relationship is much lower than building multiple proprietary one-to-one relationships.

Improved integration of applications between organizations is also a benefit that Web services users expect to reap. The potential cost savings and revenue generation from external integration far outweighs the benefits of internal application integration. However, it will be a couple of years before most organizations have finished their internal Web services projects, so not as many respondents are looking for externally-oriented benefits at this time.

It is reasonable to expect that reduced total cost of ownership (TCO) would have been the most cited IT benefit, if only because all of the benefits provided as choices contribute to TCO. That TCO was only the third most mentioned benefit suggests that most respondents are more focused on initial IT benefits of Web services deployment than on long term gains.

The internal integration of business functions is the most attractive business benefit of Web services for over a quarter of the respondents, paralleling their choice of the improved ability to integrate applications within an organization as the primary IT benefit. Web services enable the sharing of more than just applications between corporate functions; they also let various departments share information and, potentially, business processes. Web services promise to obliterate the silos that information systems have created over the last five decades and to unify organizations. However, it will take more than application integration to realign corporate value chains. Organizations implementing Web services must evaluate and, potentially, redesign the business processes that cross-functional and departmental boundaries.

As with the external application integration IT benefits expected of Web services, respondents' next highest business benefit goal is improved integration of external value chains. When information, application, and processes can be shared easily and quickly as Web services with business partners, suppliers, and even customers, then organizations will operate more efficiently and profitably. Integration enables collaboration, and collaboration drives revenue and cuts costs. Again, inter-organizational process evaluation and design will be necessary to achieve complete and effective integration.

The expectation of these two benefits helps explain why some respondents viewed Web services projects as a cost center and others said that they would contribute to corporate profit. The majority of those that named internal integration as the primary anticipated benefit also said that Web services would be a cost center in their company. On the other hand, respondents that viewed external value chain integration as the key benefit expected Web services to drive profitability and also sought the creation of new revenue streams through Web services application. CIOs and IT managers tended to be in the first camp, whereas CEOs and line-of-business managers were most often in the latter.

As stated previously, businesses are looking to Web services as a cost effective means of extending existing computing assets across the organization and to external constituents. Indeed, 43% of respondents indicated that this is how they will measure the potential impact of Web services when they justify related projects.

Respondents are looking to leverage more than just internal infrastructure and software; they will argue the merits of Web services based on utilization of internal development resources as well. The expectation that respondents will be able to use internal IT staff as their primary Web services development resource was clearly articulated. The business case for adopting a Web services architecture hinges on cost efficiencies resulting from improved use (and reuse) of existing internal IT resources, both human and machine.

Buyers' View of the Future

The majority of respondents (55%) believe that Web services will become the foundation for most organizations' on-line activities. Only 12% said that Web services would not become the pervasive operational model for Internet-based business. One third of respondents felt that it was too early to make a call either way.

These numbers suggest that there is considerable momentum behind the Web services movement, but just under half of the market still needs to be persuaded through conceptual education and practical demonstration.

What's Missing?

A majority of respondents (52%) said that Business Process Management (BPM) standards are noticeably absent from first generation Web services offerings. Many companies are looking to codify and share business processes as Web services, both internally and with external value chain members. However, none of the widely supported Web services standards (XML, SOAP, WSDL, and UDDI) address process management. The Business Process Management Initiative (BPMI) has proposed the Business Process Markup Language (BPML) as a standard schema with which to describe business processes in XML. Similarly, IBM has put forth its Web services Flow Language (WSFL). In time, it is likely that BPML, WSFL or another XML schema will become the accepted standard for management of Web services in the context of a business process, so this shortcoming of current Web services offerings should be viewed as a temporary void.

Distributed authentication methods were seen as lacking in Web services by 46% of respondents. Most initial Web services deployments will not be hampered by the current requirement for centralized authentication, as they will be internal projects that can leverage corporate profile directories to authenticate users and groups. When more companies begin sharing Web services externally, distributed authentication will be critical. Without it, there is little security and no trust unless it has been established previously. Web services or any other enabler of on-line collaboration cannot succeed with the security and trust that are provided by rigorous identity validation and authentication mechanisms.

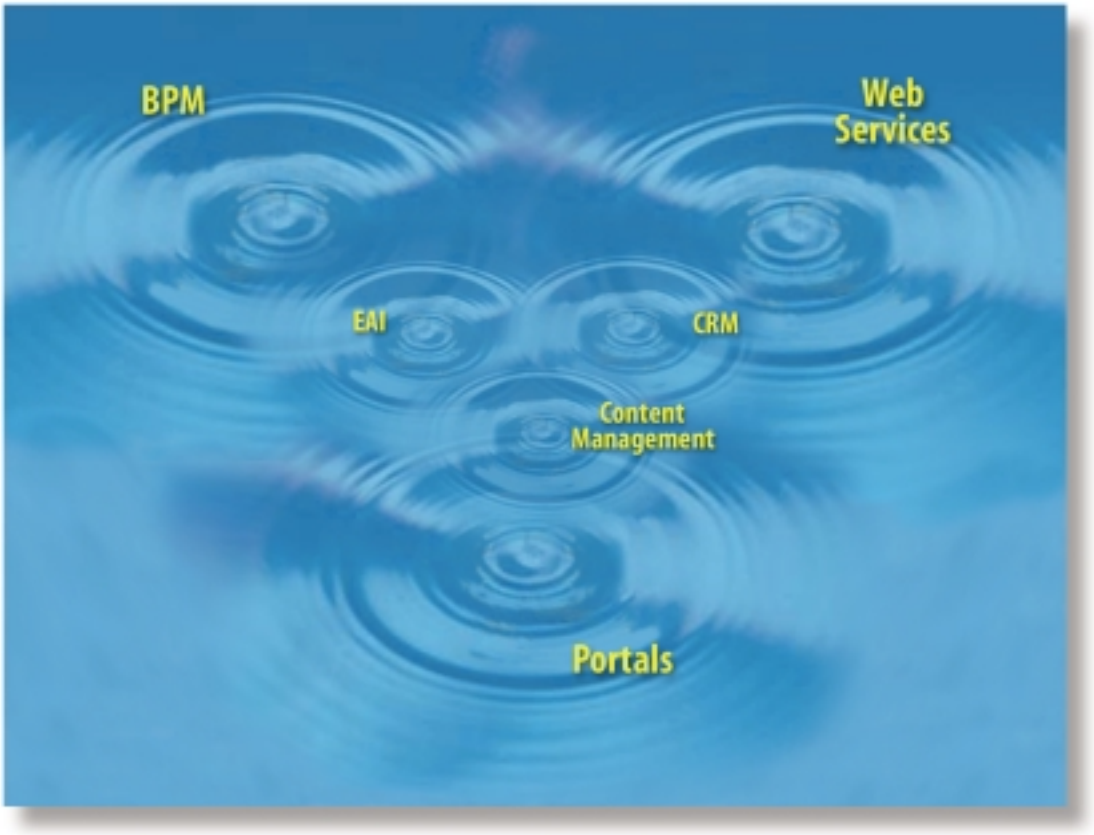
Applicability of Web Services

Over the past year, Web services has grown into its own technology sector. The value of Web services, however, will be had through other applications and systems. Enterprises will use Web services to enhance the functionality of existing technology infrastructure.

To truly understand where Web services will impact the enterprise, it is necessary to recognize the convergence of diverse technologies into Web services. An important evolution in the software world has led to the need for Web services. A number of technology sectors are complementary to Web services, and in turn complemented by Web services functionality. What is striking is the fact that all of these sectors are touched by Web services and made more whole by Web services,

yet none of them will be subsumed by Web services.

The graphic to the right helps add perspective. Think of technology sectors as existing in and of themselves. At their beginnings, they are like a small tree, with just one ring showing their young age. As they evolve, they grow concentric rings as their base widens. In the technology world, as those rings widen, they are bound to overlap. Thus, portal technology overlaps with BPM technology now, which makes perfect sense considering that users interface with their process tasks through a corporate portal.



Likewise, as Web services evolve, this sector too, will complement many other technologies. The areas of great immediate impact will be:

- Portals: as mentioned previously, business users will access Web services through their portal interface. In this way, portals will be the delivery mechanism of the future for Web services and the destination for employees to execute their business tasks;
- BPM: because Web services will be used in the context of a process, it will the job of BPM tools to orchestrate services and to automate their usage. Web services will also be processes in and of themselves (as a process can be wrapped up as a service to be delivered to another process or application);
- EAI: Web services could play the role of "lightweight" integration, helping to provide a layer of separation between applications and processes.
- Content Management: Web services are important for content management applications both because the expense and difficulty involved in provisioning Web content for business applications today, and the ability of content management suppliers, most of whom already utilize XML in their applications, to develop the standards-oriented interfaces for Web services operations. The challenge will be to get existing content that is not in XML form converted at a low cost and in a timely fashion.
- CRM: information about customers will play an increasingly large role in business processes, especially those that are customer facing. Integrating this information seamlessly into these processes will be critical—Web services can be that method of integration.

Market Analysis and Definition

Many would describe the Web services market as immature. Certainly, there are many vendors vying for mindshare and plenty of debate over the importance of standards and the state of interoperability. A look at the supplier landscape, however, does show that the market is maturing and focus is sharpening.

Web Services Supplier Landscape

Categorizing vendors is always a difficult task; it is even more difficult in the realm of Web services because any vendor can lay claim to being a provider of Web services. In fact, this situation has led to much of the confusion surrounding Web services. Many software vendors simply SOAP-enable their applications in order to claim Web services functionality. While it is true that SOAP-enabled applications are important, it is also true that they are not enough—more sophisticated Web services tools are necessary to provide real value.

Platform Vendors

Vendors such as IBM, Microsoft, BEA, Sun Microsystems, and Fujitsu are considered platform providers, supplying products and services across the realm of Web services functionality. These vendors provide both infrastructure for building and deploying Web services, as well as tools for utilizing Web services within business operations.

Infrastructure Providers

In addition to the platform vendors, there are those that provide back-end technology, such as applications servers, to ensure performance and scalability. This back-end is infrastructure both for the creation of Web services and the connection/integration to necessary systems within an enterprise. Fujitsu is also a good example of a company that provides infrastructure in the form of its Interstage product.

Service Development & Design

While the application server provides a technological foundation for Web service creation, the integrated development environment is where services are actually designed. Sun Microsystem's Forte and BEA's "Cajun" are examples of tools used as an IDE for the creation of Web services. These development environments are typically used for the creation of simple services, whereas applications (such as portals) can be development environments for composite services (where several Web services are combined to create a "meta" service).

Deployment Services

Deployment servies are offered by Web services networks, as described earlier in the report. These vendors provide support for Web services lifecycle management, complete with security services and delivery mechanisms.

Orchestration Services

Orchestration is where process and Web services intersect—these vendors facilitate usage of Web services by managing how services function within processes. Examples of orchestration vendors include Fuego, HNC Software, and Nobilis.

Application Development

There are several software vendors (among them portals), that produce applications that both consume Web services and use simple services to create composite applications and services.

Collaborative Services

One of the driving forces behind Web services acceptance is the need to better communicate with and work with partners. Vendors such as divine are offering collaborative services—secured messaging, team interaction, group feedback, telephony webinars—that allow businesses to work together in ways that were not possible five years ago. Deliver these services as Web services allows them to be seamlessly integrated to processes and to have immediate impact on business execution.

Content Services

Content management vendors are focused on turning content processes into Web services. By doing so, they are changing the game. Content management used to be focused on getting content to the Web browser so users could interact with it. With Web services, these vendors are now simply driving content to the application that needs it. In this way, the content is serving the employee, customer, or partner that needs it.

Vendor Business Models

Mirroring the dot.com phenomenon in many ways, there are pure play Web services tools that are taking the lead in niche functionality and peripheral movers that are developing Web services capabilities on top of or within their existing products. As can be expected, these vendors are coexisting for the moment, though significant merger and acquisition activiey has already begun to take place.

How Platform Vendors Shape This Emerging Market

The term "platform" has different connotations depending on who is interpreting it and their context. In the software world, the word platform conjures up images of Java, .Net, Perl, Open Source, and MacOS. The major platforms (or frameworks) in the context of Web services are J2EE and .NET. Vendors that are considered "platform providers" tend to offer products that offer a range of Web services functionality for either or both of the two major platforms. Microsoft has several products that offer Web services capabilities for the .NET platform, while IBM, BEA, and Sun Microsystems do the same for the J2EE platform. What distinguishes a platform vendor from other providers is that platform vendors provide both infrastructure products, in the form of application servers, as well as tools for orchestration and/or composite application development.

The platform vendors are essentially selling technologies to create and use Web services, not Web services themselves. The actual sale of Web services will not occur until enterprises create services that other companies may want to leverage. These platform vendors, though, shape how the market will evolve—they are driving many of the standards efforts and they provide much of the infrastructure that will be the foundation for Web services going forward.

By no means is this a deathknell for non-platform vendors. In fact, there will be ample opportunities to piggyback on the efforts and initiatives of platform providers. The added-value services that niche vendors begin to provide down the road will shape the stage in the evolution of Web services.

Company and Product Profiles

The next section details the companies that participated in this project with Delphi Group. The companies listed in this section are arranged alphabetically based on the name of the organization. This arrangement does not imply a ranking or endorsement by Delphi Group.

Closing Point

Why is there so much confusion surrounding Web services—after all, a Web service itself is a relatively simple thing? The reason is that this simple thing, the Web service, could play multiple roles within enterprises. The possibilities are literally endless, and it is difficult to forecast beyond the near-term. For now, though, it is best to remember the complementary nature of Web services—this technology will help many other technologies fulfill their promises in a more efficient manner.

Vendor Report: BEA Systems

Overview

BEA Systems addresses Web services directly through its WebLogic Platform with the inclusion of core resources, development tools, and support for standards (both J2EE specific as well as the non-Java Web services standards discussed throughout this document). WebLogic Platform 7.0, the latest version of the BEA application platform, provides a rich framework for developing and managing Web services.

BEA Systems has grown rapidly in the last several years, earning the distinction as the fastest company in history to reach $1 billion in annual sales. This growth was largely driven by popularity within the Java programming community of its development tools and application server. Today BEA is widely recognized as having the largest single marketshare for J2EE-compliant application servers.

Recognizing, however, that successful development of Web services must involve a constituency of users and designers beyond expert J2EE programmers, BEA had made the simplification of its development environment and framework for Web services a top priority. As a result, the release of WebLogic Platform 7.0 provides a number of tools and resources that enable both faster deployment and easier management of Web services.

BEA has two paths for Web services development. One is to export J2EE components (stateless EJBs) as Web services, allowing J2EE developers to continue to program in a format that is most comfortable to them. The container automates the creation of WSDL and SOAP, and allows for fine grain manipulation of Web services requests and responses. The other is to leverage the new development environment and framework called WebLogic Workshop, which allows J2EE and non-J2EE developers to create Web services based applications at a high level of abstraction above the J2EE APIs, significantly reducing the skill set, time and expertise needed to develop, test and deploy enterprise ready Web services.

Although sold as a comprehensive solution, WebLogic Platform 7.0 is not a single product but rather of a family of products focused on the building and deployment of Web services and Web-based applications. These include: the WebLogic Workshop for the development and testing of Web services; WebLogic Server for the deployment and runtime of Web services; WebLogic Integration for application integration and business process management; and WebLogic Portal for the presentation of Web services.

The Infrastructure

The foundation of WebLogic Platform 7.0 is WebLogic Server, the latest version of BEA's widely-used application server which provides for the deployment of integrated, enterprise-class applications that share information, deliver Web services, and automate collaboration among networked companies.

WebLogic Server has a J2EE 1.3-compliant, tiered architecture, supporting rich tool sets that facilitate the separation of presentation, business logic, and data in order to provide the core functionality necessary for the development and deployment of business-driven applications. WebLogic Server 7.0's support for J2EE 1.3 provides enterprises with the standards-based services necessary to build scalable and reliable J2EE applications and Web services. Beyond J2EE, WebLogic also provides clustering, security and management architecture for running large scale, mission critical applications.

BEA WebLogic Server 7.0 extends its Web services offering with Web services utilities and run-time services for integrating applications. It automatically handles the details of generating the necessary Web services interfaces, including the WSDL file that describes the service, and provides access to SOAP messages before and after invoking the back-end EJB components. This mitigates the need for specific developer expertise in either SOAP, WSDL, or XML and significantly reduces the time for the developer to write this mundane and error prone code.

One critical infrastructure aspect is support of integration—invariably, enterprises will have a mix of tools in place that will all have to work together. BEA WebLogic Server 7.0 enables interoperability with other environments, such as .NET from Microsoft or any other compliant Web services implementations. This allows for Web services developed on the BEA platform to consume .NET services. Applications not exposed as a standard Web service can be integrated by using an application-specific adapter built using the BEA framework.

J2EE Connector Architecture

The J2EE Connector Architecture (JCA) defines a standard way to integrate J2EE-based Java applications with existing enterprise applications via an adapter and an integration framework. This provides a more open platform for integration, which can reduce the need for proprietary EAI solutions. BEA WebLogic Server 7.0 includes the most recent JCA specification, as well as additional enhancements to connection and security management for the J2EE Connector Architecture adapters.

Asynchronous Messaging Services

Web services are invariably delivered as messages. Essentially, a Web service is a message or a piece of XML-based code, inside of a SOAP envelope. As a message, however, the modality Web services can either be synchronous or asynchronous. Synchronous Web services are those that establish a persistent connection between the service requested (consumer) and the service provider. For example, checking inventory levels in an ERP system can be done synchronously. Synchronous Web services establish a connection that persists until the desired information is received. Multiplied across several instance of Web services, however, this approach presents the problem of inefficient system utilization, by tying up both applications while waiting for the answer.

Many times, for example, a Web service may make a request of a remote Web service that is down for planned maintenance, requires a batch process, or needs human intervention that could take hours or days to be resolved. In these cases, synchronous Web services could tie up valuable system resources while waiting for a response. When multiplied across a large volume of requests, this limitation can dramatically slow, or even bring down a system. To further complicate matters, a Web service might get a spike in demand, potentially overloading the system.

In contrast, asynchronous messages work in a similar manner as e-mail, sending a requesting and not standing by, waiting for a reply, without requiring a connection between two applications. Also, if there is a spike in demand, the requests can be cued and processed appropriately, without ever putting too much load on the system.

BEA has developed a Web services framework that utilizes asynchronous messaging, which allows for greater efficiency in the delivery, management and processing of Web services. While building asynchronous Web services is possible with J2EE (by leveraging Message Driven EJBs, Java Messaging Services, correlating messages, etc.), it requires significant development and maintenance by the Architect and Developer. When developing Web services with WebLogic Workshop, the application framework allows developers to simply select “asynchronous” as an attribute of the Web service and the tool and framework implement all of the underlying plumbing to make it work, saving substantial time and effort.

BEA WebLogic Server 7.0 provides an integrated, "pluggable" messaging architecture for connecting diverse applications in an asynchronous manner. BEA WebLogic Server's Java Messaging Service (JMS) ensures high availability and location transparency through the clustering of connection factories and destinations.

Additionally, BEA WebLogic Server 7.0 integrates with other messaging providers through messaging bridge. This bridge supports store-and-forward (asynchronous) integration with third-party messaging implementations without the need for additional coding. Furthermore, this

bridge allows for integration with WebLogic Server's J2EE container-provided services, such as connection pooling, security management, and transaction management.

Presentation

Web services can have a sizeable impact on the enterprise, as long as users can access them to utilize them in the execution of business. BEA WebLogic Portal aims to allow just that—to simplify, personalize, and lower the cost of access to information, applications, and business processes. BEA WebLogic Portal delivers a platform that includes a portal framework with Portal Foundation Services, personalization and interaction management, intelligent administration, and integration services.

The Portal Foundation Services provide base portal functionality to simplify complex portal development, maintenance and security, and maximize the overall effectiveness of IT resources (by saving their time).

Personalization and Interaction Management help to improve the user experience and take advantage of a framework that is used to define and measure user interaction to achieve business success through targeted employee, partner, and customer incentives.

Intelligent Administration exists to reduce administration backlog and, thus, portal cost of ownership. This capability helps to ensure that users have fast access to resources they need to do their jobs, and helps provide flexibility by improving the ability to adapt the portal to new and changing business requirements.

BEA WebLogic Portal includes a Web services Portlet Builder, making it extremely easy to create a portlet front end to a remote Web service.

Integration and Orchestration

BEA WebLogic Integration delivers application integration, business process management, and B2B integration functionality to the enterprise. As noted previously in this report, Web services will have a major impact on application connectivity, and is very complementary to business process management, particularly when building new composite applications that include Web services. In response to this emerging trend, BEA is providing the infrastructure for Web services in addition to tools that provide the ability to use Web services as a method of connectivity and the process management tools to orchestrate Web services in such a way as to execute a process.

There are three functional areas that are key to the BEA WebLogic Integration platform: the Application Integration (including the Adapters); Business Process Management; and B2B Integration. The Application Integration solution is based on J2EE Connectivity Arcitecture (JCA) and bridges the gaps associated with integrating new, Web-based, packaged, and custom applications. The Business Process Management functionality then enables enterprises to design, execute, and optimize complex processes that encompass applications, systems, and human decision-makers. This functionality is what allows companies to orchestrate Web services within a given process. Finally, the B2B Integration functionality provides a process-focused environment for managing relationships and interactions with distributors, retailers, suppliers, and other business partners.

Developing Web Services

BEA WebLogic Workshop is an integrated development framework designed to allow all application developers, including those that aren't J2EE experts, to create, test, and deploy enterprise-class Web service applications on the BEA WebLogic Platform.

BEA WebLogic Workshop provides a unified development platform that enables developers to build and connect components, data, and application business logic. Developers don't need to think about EJB, JMS, SOAP, XML parsing, JAX interfaces, maintaining correlation, or managing state from point to point in a conversational Web service; rather, they need only be concerned with

the application or business logic they need to write and the back end systems they want to connect to. This approach enables application developers whose experience may be with COBOL, Visual Basic, or other procedural languages, to quickly build enterprise-class Web services on the BEA WebLogic Platform without having to learn object-oriented programming or the J2EE APIs.

BEA is focusing efforts today on the integration piece of the puzzle—WebLogic Workshop provides a development framework for building Web services that address key enterprise integration issues. It also makes building enterprise-class Web services as simple as setting controls and properties to connect existing data, business logic, and applications, either within the enterprise or externally.

WebLogic Workshop enables developers to build Web services that use asynchronous, loosely coupled, and business-level messaging. The framework allows developer to easily loosely couple their public contract (WSDL) from the underlying application logic. Unlike most other vendors solutions, when the developer needs to alter the underlying application logic, the WebLogic Workshop framework allows them to maintain the same interface, whereby not breaking existing connections with other applications every time a change is made, dramatically easing the deployment of a success Web services strategy.

Testing

BEA WebLogic Workshop speeds the testing process by generating a test harness that exposes a Web service through an easy-to-use, browser-based interface. The WebLogic Workshop Test View allows developers to run Web service methods and monitor incoming and outgoing messages using the message log - a "wiretap" on the Web service. Developers can go from writing code, to testing, to writing code again in seconds. The test harness also generates WSDL descriptions and Java-client code to access the Web service, so the developer can quickly build clients to use the new application.

Standards Participation

BEA is committed to driving J2EE and Web service innovation with close adherence to standardization. BEA WebLogic Workshop is an example of how BEA has innovated on the Java platform and is now working on standardization that will benefit the entire community. BEA has already submitted the JWS file format as a JSR through the formal standardization process.

At the Web services layer, WebLogic Workshop works with the latest version of the three major Web services standards: SOAP 1.2, WSDL 1.1, and UDDI 2.0. BEA WebLogic Workshop fully supports both the JAX-RPC and document/literal styles of SOAP messaging.

Summary

BEA Systems has been a notable player for several years for their deep level of support for J2EE and its constituency standards. This enabled developers to build applications on the WebLogic platform with a high level of inter-connectivity between other J2EE applications. What this has traditionally required, however, is hard-wired connections between these applications, which are prone to breaking as soon as one of the application components changes.

The Web services capability within BEA's WebLogic Platform 7.0 is delivered via loosely coupled, asynchronous connections rather than hard-wired, connections between applications. What further differentiates v7.0 from previous platforms, however, is the addition of resources on top of BEA support for standards, which extends the capability offered by the platform beyond that of the Java developer community. Specifically, WebLogic Platform 7.0 provides a GUI-based development environment and application framework that allows non-Java experts to become productive quickly and build Web service applications without having to learn the complexities of SOAP, WSDL, UDDI or any of the J2EE APIs.

Vendor Report: divine

Overview

With the increasingly electronic nature of business has come the ability to "extend the enterprise" through cross-firm alliances and chains of suppliers and customers. Systems that once supported internal business processes must now support the entire value chain. In response to this evolution, software providers have rushed to create solutions that support the extended enterprise.

The concept of the "extended enterprise" is espoused by divine and drives the company's Web services approach. divine's goal is to help companies to extend their enterprises by improving collaboration, interaction, and knowledge sharing across the entire value chain.

divine has a three-pronged approach to fulfilling this goal—professional services to provide strategy and implementation assistance, software services to provide the technology and premium content to extend knowledge throughout the value chain, and managed services to ensure sound infrastructure and operations.

All three groups play an important role in fulfilling divine's Web services strategy. The professional services group provides the experience of integrating diverse systems and brings to the table the knowledge of how Web services can affect this process. The managed services business provides, mission critical support for extended enterprise applications, and can provide the performance, security, and scalability "fabric" in which Web services-based applications can execute. It is also important to note that divine's software services division provides products which enhance the value proposition of Web services—without providing Web services infrastructure per se. Rather, divine adds innovation and value-add applications and solutions which leverage state-of-the-art service-oriented platforms like Weblogic, Websphere, and .NET.

Software Services

The software services divine provides are concentrated in the areas of collaboration, content management, and customer interaction management. These collaboration, content management, and CIM technologies give divine a foundation from which the company can deliver Web services solutions that support the extended enterprise.

Collaboration

The collaboration technologies help businesses better interact with customers, partners, and employees. They enable secured messaging, team interaction, group feedback and webinars help the enterprise interact and connect internally and externally with the professionals important to conducting business.

Content Management

divine's content management technologies give content consumers the power to create and use content in the way that best suits them. The goal is twofold—to reduce publishing delays (and thereby optimize timely content) and to build stronger relationships with everyone in the enterprise by allowing them to directly contribute to online business in their own unique way.

Customer Interaction Management

Customer Interaction Management (CIM) became a hot tool when enterprises realized the potential to turn browsers into buyers. divine's CIM services provide call center analysis and reporting capabilities, inbound and outbound telephony services, predictive dialing, and campaign management.

Utilizing Web services, divine's technologies can increase customer leverage of existing back-end systems. Enterprises can better use their existing infrastructure to share knowledge internally, to get knowledge to partners and suppliers, and to use knowledge more efficiently to service customers.

WSIL (Web services Infrastructure Lab)

The mission of the WSIL is to deliver the interoperability layer for divine's family of products. The focus of WSIL is to ensure that divine's products will interoperate not only amongst themselves in a "federated suite" approach, but they will also be able to integrate and interoperate with the rest of the systems within the IT ecosystem of the extended enterprise. WSIL has developed a set of agile tools, platforms, and APIs that will allow divine's systems to interoperate with each other while also delivering the lightweight integration of an enterprise's systems. To achieve this mission, divine has set up a methodology in which divine's labs, managed services, and professional services are treated as customers; WSIL then undertakes the responsibility to deliver to these customers quality products with frequent releases. In this way, divine can drive the innovation necessary to meet customer needs while also creating solutions that optimize internal operations.

Methodology

Within WSIL, the basic methodology used is to first identify target services using a scenario-driven approach. As with any business unit, a list of potential go-forward scenarios is compiled—in this case, it is a list of potential services to create. The costs and risks of each potential service are then compared in order to prioritize the development of each service. The factors considered in prioritizing services include: strategic value; revenue generation capability; interoperability within divine's family of products; capability to extend the enterprise; urgency; customer acquisition or retention capability; differentiating capability; number of divine applications that can use the service; potential to generate cost savings; amount of reusable code the service can deliver; and the cost to develop the service. divine can then decide if the needs for each service to be developed can be met with existing divine products or will need partner products, too. Within the WSIL, divine's team of professional service staffers can begin to spec out and build services into a service repository, an internal Web site containing downloads, code, and other pertinent service information related to common code and services.

While divine will eventually roll out a host of services, the current focus is on security services, the Web services engine, user management services, search services, and document services. The security services include single sign-on, delegated authentication, and authorization—creating the secure environment necessary for Web services to gain traction. The Web services engine provides Web service enablement, federated access, and service registry. The user management services allow for managing user preferences and metadata. Search services empower federated searching capabilities. Finally, the document services include DAV-enabled document sharing (DAV is short for Web-based Distributed Authoring and Versioning—a set of extensions to the HTTP protocol that allows users to collaboratively edit and manage files on remote web servers), check-in/check-out capabilities, versioning, and federated access.

Core Services

divine's Core Services are a set of APIs that allow interoperability and a common application framework to be centralized and leveraged across divine products. These Core Services allow for a high level of flexibility and extensibility while also maintaining enterprise scalability and reliability. divine also provides the secure and transactional environment required of solutions that utilize Internet infrastructures.

The Core Services are implemented on open platforms in order to fulfill their purposes. Both existing proprietary divine products and infrastructures and open, and widely-accepted platforms such as J2EE and Web services are leveraged. This approach allows for the leveraging of existing assets while also opening the platform to interoperability and integration with other divine products and divine's customers' extended enterprises.

Benefits

As discussed in the core body of this report, there are several capabilities that Web services architectures have to provide in order to both provide value and meet customer requirements. The Core Services architecture is designed to do just that, providing flexibility, loose-coupling, scalability, reliability, security, multi-tenancy support, leverage of existing technology infrastructure, and interoperability of applications.

Flexibility

Web services solutions will interact with diverse applications and products. Thus, flexibility becomes a key aspect of any solution. In response to this, divine offers a centralized common application framework that covers the basic needs of all divine products, while also offering the flexibility to implement domain-specific features of each individual product.

Loose-Coupling

Part of the promise of Web services technology is the ability to have modular components of applications that are reusable by several applications or processes. What makes this possible is the loose-coupling of services. divine provides this loose-coupling to support an architecture in which services can dynamically join and leave an application. These loosely-coupled services will help to ensure that the development and maintenance of services will have minimal to no impact on the applications that consume the services.

Scalability and Reliability

Scalability and reliability go hand in hand—and both are crucial aspects to any Web services solution. Because divine's Core Services are centralized and shared by any number of divine products, their underlying architecture must be able to scale to handle the combined load of the entire divine product suite. To ensure reliability, divine creates load-balanced, fail-safe, and hot-deployed services that, in turn, create an "always on" environment. Furthermore, the architecture is built on a platform that is transactional, allowing for the validation and guaranteed-delivery of transactions and applications.

Security

As discussed several times throughout this report, security is one of the hot issues surrounding Web services. Any time an Internet infrastructure is utilized for the transfer of data or information, the question of the security of that data is of supreme importance. While there are several security initiatives in the Web services realm that are jockeying for widespread acceptance, divine is also addressing security on its own. divine services can be secured at both a fine and coarse grain-level and encrypted on the wire.

Multi-Tenancy

When most enterprise software packages were created, they were architected for one instance of the software to sit on one server. This one-to-one ratio was perpetuated by the ASPs (Application Service Providers) that popped up in the 1990s—the ASP developed expertise in managing the software and essentially rented it out to customers. The problem with that model was scalability. For every incremental customer, the ASP had to take on the additional burden and costs of managing another server and software instance. In order to scale, multi-tenancy becomes a requirement. Defined, multi-tenancy is the ability for more than one enterprise to share one instance of a software license. In the ASP analogy, this would mean the ability for the ASP to host multiple customers on one server. For divine, multi-tenancy is a factor in being able to deliver low-cost services by taking advantage of the economies of scale possible through multi-tenancy. Thus, multi-tenancy support is a crucial aspect in allowing divine's Managed Services group to deliver software as a service.

Leverage of Existing Technology Infrastructure

For many organizations, the goal of Web services initiatives will be to optimize the use of

technologies already in place. divine's strategy is to help companies do just that. The Core Services will be implemented on both proprietary divine products and infrastructures, as well as open platforms such as J2EE. The J2EE technologies will provide the scalable and reliable container to deploy secure, transactional services. At times, the Core Services may leverage proprietary divine products and server infrastructures; these services would rely on the particular containing application to achieve some of the features lost by not deploying to a J2EE application server. In addition to J2EE support, divine fully embraces Microsoft-shops and the .NET framework. The goal is to be in a position to provide products for the many enterprises with heterogeneous computing environments.

Interoperability

Lastly, divine is driving the interoperability that Web services promise to provide. One way to view the Core Services is as a common layer for interoperability of the divine suite of products.

By providing an interoperability layer for divine products, the WSIL will allow divine to sell the Extended Enterprise Suite. divine's products will be available as both a suite and as best-of-breed point products. This in turn will enable cross-selling opportunities amongst divine's family of products, as well as create many professional services integration opportunities.

An interoperability layer for divine's products creates an added value proposition for customers that leverage more than one of divine's products. Many, if not most, customers will have a diverse range of products. divine is addressing the interoperability issue for products other than its own, too—WSIL is also a place where divine can ensure that the company's products can be integrated with partner products and platforms.

The true value of WSIL lies in its ability to be a place of innovation. Professional services team members can apply their knowledge of the requirement for integrating diverse systems; divine staff can apply experience with software that will be important to a services architecture; and the managed services group can provide the infrastructure to test innovative ideas. Many software companies are waiting for the Web services revolution to create a business case that drives them to a services paradigm. divine is proactively creating a business case for Web services.

Standards and More Standards

As stated previously, divine helps its customers leverage their back-end systems to more efficiently use knowledge. For this reason, divine is very involved in Web services standards development. Part of divine's Web services strategy is to both drive industry standards development and to ensure compatibility of divine products with accepted standards. WSIL provides a place where divine can track and implement Web services standards that will allow more efficient interoperability.

divine is taking an active leadership role in the development of the WSRP (Web services for Remote Portals) and WSIA (Web services for Interactive Applications) standards. Using WSRP, portals can integrate content and applications from many internal and external content providers. The portal administrator chooses the desired services from a list and integrates them; no programmers are required to tie new content and applications into the portal.

The WSIA initiative is creating an XML and Web services-centric framework for interactive Web applications. The goal is twofold: first, to enable businesses to distribute web applications through multiple revenue channels and second, to enable new services or applications to be created by leveraging existing applications across the Web. By driving the development of the WSRP and WSIA standards, divine can make the claim that “they don't make the portal server, they make it better through content and applications.”

Case Study

Leveraging XML Web services for Bridging .NET and J2EE; Allows for Faster Time-To-Market and Graduation from a Stateless World

The Worldwide E-Comm Architecture team at Compaq (now Hewlett-Packard) was faced with the challenge of bridging two otherwise disconnected worlds – those define by Microsoft infrastructure and the J2EE environment.

At the root of this initiative was the need to connect a SAP-based order management system through SAP's J2EE-based IPC (Internet Pricing and Configuration) engine with Compaq's Siebel-based CRM solution, running within a Microsoft environment.

Once in place, the system allows a call center representative in Scotland, for example, to initiate customer orders within the version of Siebel hosted in Munich, by invoking the IPC configurator from a server in Houston. The order is configured and executed transparently through a series of transcontinental XML messages linking all necessary applications.

Given the widespread Microsoft orientation within Compaq's IT environment, this solution needed to address many Web-based requirements with Java, but also be managed by a team most familiar with the Microsoft ASP (Active Server Page) environment.

To address this, Compaq and divine built core backend functions in Java, and developed XML services and consuming applications written in Microsoft's C# (one of the primary languages of the .Net framework). A custom Web services interface developed by divine communicates with the J2EE-based IPC engine via SOAP messages generated by the Apache SOAP engine.

Accelerating Time-To-Market

Allowing Compaq to maintain and customize services within the .Net framework, while building backend functions in Java, provided for fast development and experimentation with new applications. For example, an application was quickly developed which allows product configuration to be executed on iPAQ handheld devices.

Delivering access to the IPC engine as a Web service, however, also addressed another key business driver – faster time-to-market with new product offerings. With the new flow of products every 6-9 months, Compaq needs to ensure that development cycles keep up, otherwise risk a lag-time from when a product is released to when it can be configured within the system.

Prior to integration with the IPC engine, the introduction of a new product required explicit definition within the catalog structures used by reps and partners. With the Web services interface developed by divine, the IPC engine now provides access to product descriptions and allows real-time validation of configuration.

As a result of this need, now when a configuration session is initiated, the product is validated against engineering rules code, which govern what is possible vis-à-vis product requirements and business rules. For example, determining which product components and accessories are compatible with one another. It also provides for distributed control over what to display/not to display, such as not displaying the option of a CD-ROM if a DVD is selected with a PC order.

From Stateless to Stateful

Given the global nature of the solution, Compaq and divine chose to transport XML messages over HTTP to avoid problems piercing the firewall. To provide the level of functionality above, however, required much more than simply displaying product data as HTML.

This required moving from a "stateless" environment, to one that supports sessions (i.e.,

managing the relationships between multiple interactions). The Web itself is stateless – when a page is viewed there is no awareness to what might have been done on previous page or what may happen on the next page.

When configuring products online, however, the stateless nature of the Web would otherwise lose all the information entered when moving from one page to another. This requires "session management," the ability to manage all related interactions within a single process.

The Web services interface developed by divine allows for the IPC session to be launched from an ASP page generated from a Microsoft IIS server. By managing the session data as XML transactions rather than stored procedures, flat files or function calls (popular approaches in the absence of a Web services architecture), the solution provides the flexibility to extend this session data to other applications.

Building the Extended Enterprise

One of the advantages of a Web services approach to enterprise computing is linking the extended enterprise—the value chain of participants across the network of any business' customers and partners. In the case of Compaq, the initiative of connecting its Siebel solution to the SAP-based application could have been accomplished with a proprietary middleware approach. What that approach would have sacrificed, however, would have been the agility to reuse services by other applications or the flexibility to add other participants such as partners and customers.

It is the nature of this notion that has represented the guiding principal behind the development and management of divine, inc. As divine founder and CEO, Andrew "Flip" Filipowski, explains, "the premise divine was founded under is that the extended enterprise opportunity is a new growth ring, redefining how IT can affect business strategy and business outcome. Before this growth ring, almost all business systems were created for internal consumption, used only by employees of companies who deployed them."

Web services presents a fundamental shift in the way software is deployed, putting it in the hands of users outside the company. As Flip explains "this growth ring bridges the chasm to where the predominant user of new applications is going to be a member of the value chain community, someone other than an employee of the company."

With the deployment of enterprise software, one the largest issues relative to long-term cost and realizable value is not what is directly involved in the initial rollout but rather resources and effort required to implement changes. For solutions targeting the extended enterprise, this cost has the potential to be several orders of magnitude greater, increasing the frequency with which changes will be required. The loosely-coupled nature of Web services, enabled the core set of standards described throughout this document, offers a significant advantage in the reuse and interchangeability of application services, over previous IT architectures.

Summary

divine's three-pronged approach to Web services positions the company well to deliver an interoperability layer for both its own products and those of other software providers. By working with standards organizations and driving the development of standards such as WSRP, divine is working to ensure that solutions remain open and scalable. Further development of open standards can only help to make divine's interoperability layer more efficient. The utilization of WSIL will allow divine to continually create innovative solutions to both internal business problems and those discovered by the professional services team. These solutions will drive the ability of companies to work more collaboratively with members of the value chain and to extend the enterprise.

Vendor Report: Fuego

Overview

The paradigm shift that is Web services drives the need for enterprise systems that are flexible and customizable. The integration of these systems must address an enterprise's need for new services, new processes, and new or updated applications. EAI is one potential solution, though implementation of EAI software can be a long, complex project. In today's world EAI vendors have recognized the need to add workflow and/or process flow capabilities to their integration suite. The problem with the approach of cobbling together an integration broker with one or more BPM engines is that, so far, it hasn't served the purpose of shortening or simplifying integration projects.

Earlier in this report, the convergence of BPM, Portal, and Web services technology was illustrated. The concentric rings of that diagram show how the lines between these technologies are blurring. While one sector will not necessarily replace another, the complementary nature of these technologies will grow—and smart vendors and enterprises will exploit that. Fuego's products and services sit directly on the concentric rings of the convergence triangle. The critical difference is that Fuego provides a single engine that manages a work portal, workflow, process flow and is a fully functional integration broker.

Fuego espouses a new model for integration, and calls it Business Services Orchestration (BSO). BSO is a process-driven method to coordinate and manage internal and external business services. What Fuego calls a business service is what this report has referred to as a Web service. Because business managers deal with processes, and BSO is one way to manage processes, it makes sense to use the term business service in place of Web service. After all, if Web services cannot make the lives of businesspeople easier, they will have zero effect on enterprises.

Fuego allows companies to identify and catalog business services, and then orchestrate them based on a defined business process. The system gives users the ability to design process flows, to create and implement business rules, and to identify the business service integration points within a process. It then generates an executable process model to manage enterprise processes. This executable process model can then, in turn, be published as a Web service (if the interaction is discrete) or a process contract that includes a sequence of Web services that fulfills a public process.

The real value of orchestrating services stems from the ability to leverage existing IT and human resources. Fuego's Web services strategy is clearly to be a services orchestrator, giving companies the ability to manage services within processes and to connect services from one application to another, as well as publishing selected orchestrations as new, higher level services.

It's About Process Management

While many integration projects in the past have focused on the data being shared between applications, Fuego takes a broader approach. The number one focus is on business objective—what do business managers need processes to do? This concentration on the business side of process management and integration is a key differentiator for Fuego. Because businesspeople are responsible for the performance of a process, it is important for them to be able to design and manage it to their needs. Furthermore, especially in today's rapidly changing business environment, where the need is to change processes on-the-fly, it is imperative for a businessperson to be able to change their processes quickly, on-the-fly, without having to worry about lengthy integration challenges, and without affecting the internal and external users of the processes.

Fuego's software uses process-modeling facilities to sketch the activities, transitions, and user roles within each business process. As opposed to other systems that often require highly skilled IT technicians to create or refine complex business rules and adapters in Java or C++ code, Fuego employs uses intuitive drag-and-drop tools to quickly, easily design even both simple and complex business processes.

When the process activity has been defined, Fuego automatically generates Java proxies that connect to the services needed from underlying systems by discovering the metadata that exposes them or simply introspecting Java Jars. Peer-to-peer and n-level process nesting ensure the maintainability, readability, and reusability of process designs. In this way, each designed process can become a service in and of itself for reuse in yet another process and/or, when published as a discrete or composite Web service by other applications. And, because process models are stored as XML files, process models and their complete documentation can be shared throughout the organization or with external partners.

A New Way to Adapt

When it comes to integrating the applications that support enterprise processes, the use of technology adapters is commonplace. The adapters ensure that all events are updated in a "standard" form that can be recognized by several different applications. The problem with adapters in the Web services paradigm is the extensive customization that is typically required, which can lead to their not being reusable.

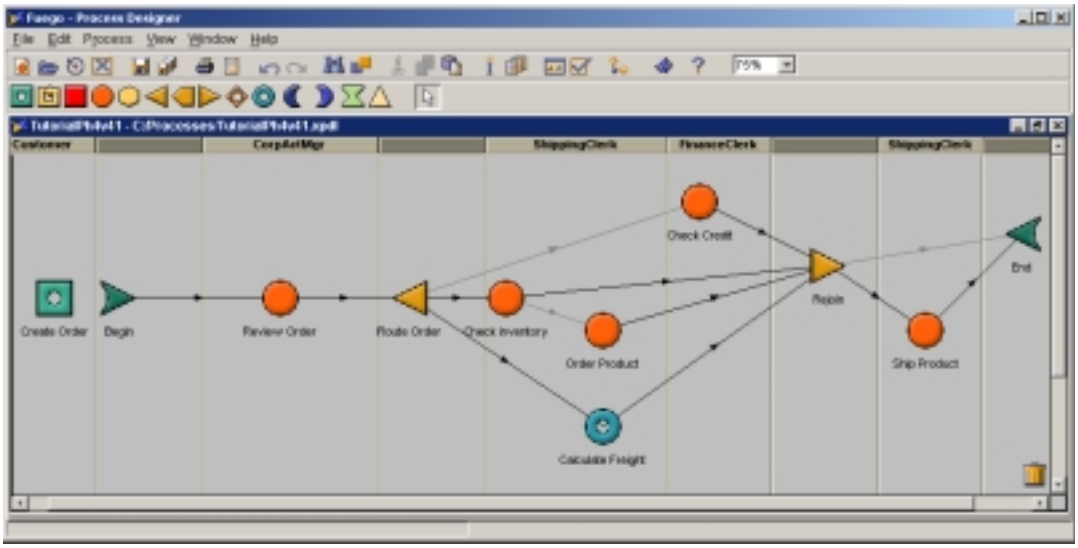
Fuego uses its Fuego Component Manager, a dynamic, self-generating adapter facility as opposed to the traditional adapter method of forming a hard link from a particular application to a particular messaging infrastructure. Fuego's adaptive technology employs a services discovery mechanism to connect to a wide range of common, industry-standard technologies including WSDL descriptions of Web services. This mechanism introspects the metadata from any API to identify the business service and incumbent data, then translates those elements to automatically create a component of the larger business process that acts as a proxy to the API.

This new way to connect services is one of the key distinctions between traditional EAI systems and Fuego. The Fuego approach is designed to allow the business user to simply design a process and go, without having to understand the underlying systems and how they communicate technologically.

While existing EAI systems tend to use complex scripting formulations, Fuego instead employs a more intuitive integration language—the Fuego Component Integration Language (CIL). The CIL defines the rules used to control business services, to transform and use data, and to manipulate the common data objects used in orchestration. Furthermore, it incorporates interactive debuggers, syntax checkers, and other tools to further simplify the scripting task.

The Driving Engine

At the heart of Fuego's solution is an orchestration engine that manages and executes the business services according to rules in the designed processes. Orchestrated processes themselves can be exposed as Web services and executed across both internal and external processes or other applications. In cases of more involved B2B processes, multiple parallel engines can be linked via TCP/IP to operate in a federated configuration. To



minimize the friction with security policies in B2B applications, Fuego employs XML and SOAP over HTTP or HTTPS.

Moving Toward Zero Latency

Because applications and processes need to stay updated, Fuego's engine also has a version-control capability that streamlines the update process. This version-control function allows multiple version of a process to run simultaneously— an important consideration given that with traditional EAI systems, making changes typically meant that the messaging bus and all process activities must be completely shut down. That led to higher costs and poor overall process performance.

The best of this new breed also incorporates high-availability backup and fail-over capabilities. These version control and backup features are especially useful in orchestrating critical, long-running business processes.

The Human Factor

Traditional integration projects have focused on building the middleware and adapters that allow applications to communicate. Often, the fact that humans run business processes is overlooked. Fuego addresses head-on the need for humans to be directly involved in processes. It creates a work portal designed specifically to encourage and support human participation within a business process. When one or more workers must contribute to a process, Fuego creates a single work queue for all participants. Web browsers or Java thin-client applications can then be used to present queues to workers connected by a federation of orchestration engines.

Reporting

A business integration scheme or business process tool must be able to give managers a picture of what is happening with execution at any given moment. With Fuego one or more orchestration engines can be configured to capture and forward reporting information to a centralized Operational Data Store (ODS). The information from that ODS can then be presented into customized reports utilizing Online Analytical Processing (OLAP).

Case Study: Tesoro

Tesoro uses Fuego to slash the process time, error rates and training times related to ERP system.

Tesoro Petroleum's Marine Services unit serves the marine and offshore energy industries by operating fueling and re-provisioning stations for ocean-going vessels on the United States Gulf Coast. Tesoro places the utmost importance on customer service, which means the company must be effective at fulfilling every need a ship might have relating to replenishing supplies.

Tesoro uses an ERP system into which employees enter and access transaction data for sales orders, credit checks, new customer setup, order fulfillment and billing. Extensive training, at times lasting up to a year, is required to properly teach employees to use the ERP system's complex interface, which itself requires 13 to 15 screens and 50 to 70 mouse clicks to process an order.

The difficulty of mastering the ERP system had been the weak link in Tesoro's customer service orientation because it led to inaccurate data entry, which in turn created erroneous inventory records. Imprecise inventory and demand information produced out-of-stock conditions and lost sales. In addition, a high turnover rate among employees meant new workers were continuously being trained, perpetuating the problem.

Tesoro chose Fuego to develop a system that integrated the company's SAP ERP system via Fuego. The resulting three-page interface integrates with SAP, allowing users to access the customer and product information that resides in the SAP system.

After a user has accessed this interface and entered relevant sales information, the interface automatically retrieves price and tax information from the SAP database and runs a credit check on the customer. The user is guided through the process of fulfilling the order and verifying tax status and pricing with the customer. Inventory and

sales information is then automatically updated to the SAP system. Finally, all billing information is sent electronically to the accounting department in order to create an invoice. In total, the new Fuego process invokes up to 14 SAP transactions.

As always, any solution is judged by its results. With Fuego, Tesoro was able to create a supervisory application that integrated a browser-based interface with its ERP system and back-office accounting processes in five weeks. The new process demonstrated that training requirements could be significantly reduced, from months to less than a day. Additionally, Tesoro realized its ability to synchronize its SAP and non-SAP databases, as well as rapidly develop new supervisory applications as needed. Also of great benefit to Tesoro was getting a system that can automatically handle exceptions and forward them to the proper department in the company. Thus, not only did Tesoro realize cost savings by improving process performance and reducing training time, but because this system helped with utilization of SAP, it reduced the total cost of ownership of the ERP system.

Summary

Fuego's solution can be an interesting approach to enterprise integration. It has the potential to reduce the complexity that can accompany an EAI project. It also can help enterprises better utilize existing systems and thus, become more flexible. That flexibility is what is crucial to survival in today's business climate.

Vendor Report: Fujitsu Software

Introduction

The Interstage product suite was born in 1998 as an application server for Web technology and online transaction processing. Since then, it has grown into a standards-based Web services and B2B Commerce enabler. Interstage provides an architecture through which to integrate public services to processes and applications through a portal that makes those services available to other applications or end-users in the form (XML or HTML) that is necessary. The Interstage platform is comprised of the following three layers:

- **Foundation layer:** which consists of the Interstage Application Server, Interstage Security Director, Interstage Traffic Director
- **Integration layer:** which consists of Interstage PortalWorks, Interstage Contentwiz and Interstage CollaborationRing
- **Development layer:** consists of Interstage Solution Suite, Interstage Apworks and Interstage Apcoordinator

Application Server

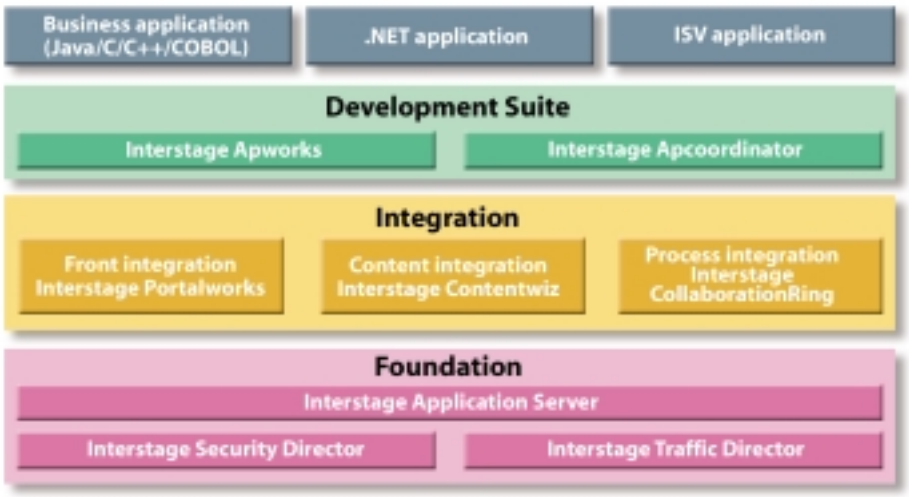
The Interstage Application Server is the core of the Interstage product suite. It employs the latest standard technology such as Java, CORBA, and XML in order to link existing information system resources to the Web environment. The Application Server is J2EE-compliant and delivers Web service correspondence through the use of standards such as SOAP, UDDI, and WSDL. It is the job of the Application Server to generate SOAP interfaces for legacy applications, allowing an organization to capitalize on its existing infrastructure. With these Web services capabilities, in addition to reliable online transaction processing technology and a bridge function to achieve mutual linkage between EJB or CORBA applications and SOAP, a company can construct a Web system that brings existing systems into the Internet age.

The hot-standby function, in combination with the dynamic application change function, help to ensure system management with high reliability and availability. Based on the load status of the Application Server, the Network Access Server can control system traffic to help prevent any overloads that could lead to downtime.

APWORKS is the integrated development environment (IDE) of the Application Server. It provides tools and component parts necessary for application development supporting all Interstage products. APWORKS features Wizard functions that allow for the utilization of open Web applications such as JSP. It is designed for ease-of-use, with tutorials and practice samples to help users understand the development flow. The User Interface offers functions such as screen control, input support, and database processing in an effort to allow a high degree of productivity in application development.

The Application Server can help to position an enterprise to capitalize on the business opportunity available with Web services. Using the Application Server, companies can prepare their

Collaborative Business Integration: Interstage



environment to harness the both internal and external system components.

APWORKS has always been equipped with development environments adapted to the technologies that the Application Server supports. Users have been able to create EJB and CORBA applications, building business logic into templates created through Wizards. As the Application Server has grown to support Web services, so too has APWORKS—there is now a mutual linkage between EJB or CORBA applications and SOAP. This allows developers to effectively utilize existing EJB and CORBA resources. APWORKS automatically generates WSDL to describe how Interstage Web services interact with other services.

Integration Server

Integration Server is the base software for business collaboration. It uses EAI technology to integrate information systems in order to facilitate business/work integration on the Internet. Integration Server allows for the construction of a flexible and reliable system using standard workflow technology such as SWAP, Wf-XML, and XML. The Integration Server consists of three component products: CollaborationRing PM (Process Manager), for the integration of business processes between enterprises; CollaborationRing TPM (Trading Partner Manager), for trading on the Internet; and various adapters linking disparate systems.

CollaborationRing is designed to integrate enterprise systems both internal and external to a company without requiring a major overhaul of the systems themselves. CollaborationRing PM enhances document and process management functions, allowing for the exchange of forms or documents between different systems and linking business processes according to company procedures. The document management function in CollaborationRing PM automatically converts forms created in existing systems, such as ERP, for use in the Web services environment.

CollaborationRing PM provides a process management function, allowing users to design processes independent of the underlying applications. This process management function gives users a drag-and-drop interface for process design. These processes can then interact with the underlying applications through data mapping, format conversion, and character code conversion. CollaborationRing PM also includes several sets of adapters to communicate with enterprise systems. There are adapters for the Internet (http, SSL, Corba, etc.), file transfer (FTP, SMTP, NetOS, etc.), messaging (MQD, MQSeries), ERP (Glovia, SAP, Oracle), and Legacy EDI (EDIFACT, Zengin TCP/IP (Japanese Banking EDI protocol) and Basic/JCA (Retail EDI protocol defined by Japanese chain store consortium).

The functionality provided by CollaborationRing PM is pushed outside the boundaries of the enterprise by CollaborationRing TPM, which provides the XML-based e-commerce framework for enterprises to utilize the Internet and the definition/execution environment for inter-enterprise trading scenarios such as RosettaNet. CollaborationRing TPM determines the format of business documents exchanged among enterprises according to standardized rules and controls the process involved with inter-organizational trading. It also analyzes the processes involved in trading so that companies can improve the efficiency of trading processes and eliminate bottlenecks.

The combination of CollaborationRing PM and TPM allows a company to take advantage of internal process management and apply that to external process management. For example, a company could add a trading partner using CollaborationRing TPM without having to alter the internal processes that CollaborationRing PM is running. CollaborationRing PM concentrates on managing the internal connections (i.e. ERP systems) that processes need, while CollaborationRing TPM concentrates on managing the trading processes that involve other companies.

Portal Server

Portal Server is the foundation for business portal construction, integrating the views of the various front-end business systems. It consists of 2 products: PortalWorks, which provides an integrated front-end for enterprise information with personalized presentation to a Web browser or mobile phone.

PortalWorks provides an environment for building a Web-based front-end system that can be personalized by consumers, employees, or partners. It is used to build a portal that combines a user's needs, whether they relate to processes, applications, or information, into one common interface. The architecture of PortalWorks is comprised of two main elements. The first element registers and manages services (Web sites, information systems, etc.) while the other element, a scenario mechanism, defines the handling of data stored in each registered service. There are several plug-in modules to increase the number of available services. Some examples of the types of services that can be created are full text search, groupware communication, and business intelligence via OLAP. The system administrator can manage access to these services by user and users can register or unenroll from services, as well.

Once services have been registered, PortalWorks automatically creates a portal window that groups services together based on business rules. It automates the data exchange among services by defining the delivery relation of the data between displayed windows as a scenario. All of the data exchange is performed via XML and linked between services by SOAP. This allows a user to have one interface to all of the services they need to conduct their jobs, while all necessary data links happen automatically.

Network Access Server

Network Access Server is the network management base software that ensures security, reliability, safety, and high availability being indispensable to the Internet-base enterprise network. This consists of two products, Security Director that accomplishes a firm security management with linking Application Server, and Traffic Director, which raises reliability/availability of Application Server by load distribution function with flexible traffic management.

When an enterprise begins to provide services to users through the Internet, several problems can arise: data communication security; bulk system access from an unanticipated large number of users resulting in costly downtime; and traffic reduction due to network traffic congestion. The Network Access Server is designed to address these problems. It is located between the Application Server and the Internet in order to provide a level of security and reliability.

The Traffic Director is responsible for managing load distribution on the Web servers, controlling Quality of Service (QoS), and data caching. By using multiple servers to handle large numbers of system accesses, Traffic Director helps to provide a stable response, and construct flexible and expandable systems that can continue to provide uninterrupted service even when a WWW server fails or another server is added to the system.

Load distribution is also performed for multiple WWW servers according to individual cookies or SSL sessions. This means that user requests are not simply allocated to a server with a light load, rather interactive transaction processing from the same user is allocated to the same server. Thus, the system can handle applications that require a user to enter information in a number of windows.

Traffic Director provides a function to allocate the bandwidth of the network line leading from the Internet to Interstage Application Server according to each application and as determined by QoS policy. This enables a stable service to be provided even when lines are congested because important traffic bound for the Interstage Application Server is protected from other excess traffic.

Interstage Traffic Director provides a function to cache data received from a WWW server as it is returned to a Web browser. This makes it possible to respond quickly to frequent requests, and

reduce the load on the background WWW server.

Interstage Security Director provides functionality that aims to ensure security on servers that perform inter-corporate transactions and disclose information to customers on the Internet. Its job is to prevent illegal access to the network. Intranet server information (host name, IP address, etc.) is concealed from the Internet to protect the Intranet against attack. SSL user authorization and functions to allow or disallow access to individual URLs make it possible to construct a safe internal network access environment. Security Director also needs to protect application servers through collective access management. Because Security Director authenticates users at the application server front end, it protects the Intranet and application servers from multiple, unspecified attacks. Furthermore, because authentication results are passed to application servers, users can obtain access permission through a single authentication. This makes it possible to construct a system that is both secure and user-friendly.

Security Director offers a firewall function—if an attack intended to interfere with service or a pre-registered event occurs, an alert notification will be issued to allow users to take action. Network operating conditions can also be monitored or investigated by logging or monitoring the operational status. Additionally, clustered configurations are supported, improving the reliability of the firewall function.

There is also an application gateway function that offers collective management of access to application servers through user authentication, address conversion, and access control. To protect the Intranet or application servers from multiple unspecified attacks, Security Director can authenticate users at the application server front end and allow only approved users to access the server. With address conversion, the relay function of HTTP and IIOP is used, and Intranet server information (host name, IP address, etc.) is concealed from the Internet to protect the Intranet from attack. For servers that can be accessed via Security Director, functionality is provided to allow or disallow access to individual URLs or CORBA applications. This minimizes the information disclosed to the outside (i.e., the Internet) and improves the safety of corporate systems on the Internet.

Summary

All the products of the Interstage suite work together to develop and deploy Web services. The Network Access Server provides system security and load balancing to ensure uptime and reliability. The Application Server provides and infrastructure layer that is J2EE-compliant and supports SOAP, UDDI and WSDL. The Integration Server can leverage RosettaNet, and ebXML, while the Portal Server can deliver services to end-users where they need them, whether they are on a mobile phone or for consumption by an application.

Vendor Report: HNC Software

Overview

Blaze Advisor from HNC is a rules management system used to design, develop, execute, and maintain rules-based business applications. Blaze Advisor aims to improve the way enterprises manage business applications and processes by enabling them to more quickly develop complex applications, respond to changing customer needs, implement regulatory compliance, and reduce the total cost of day-to-day operations. Blaze Advisor is concerned with five major things in the context of rules-based systems: development, testing, execution at runtime, maintenance, and management. It is built to be a callable component; an application would call upon the rules in Blaze Advisor and Blaze Advisor would come back with an answer based on existing information.

Data used in making business decisions may be read from database tables, XML documents, MQ Series data streams, Java or COM objects, or passed parameters from the calling application. At the same time, Blaze Advisor can make calls to external applications, functions, and business systems from within rules to initiate external processes, make complex calculations, or update values in external data stores.

Defining the Relationship Between Rules Management and Business Process

To be sure, overlap exists between the concepts of workflow, BPM, and rules management. BPM is more concerned with the overall tasks used to drive a complete process from start to finish, while workflow tends to concentrate more on the steps used to drive information or products through a subset of an overall business process. Both of these process concepts include the integration of automated systems with physical tasks occurring over time and which often include waiting for one step to be completed before being able to continue later with the remainder of the process. Real-time business decisions differ from workflow and BPM in that they focus on taking a specific action or making a particular decision based on the conditions present at a given time. Such decisions may be used as tools to drive the succeeding step in a larger business process. It may be helpful to think of business decisions as a subset or component of a workflow or business process.

The more complex decisions typically require external programming in order to be executed. Rules management addresses the needs of condition-based decision making, separating the logic behind business policies and practices from the mechanics involved in carrying out the recommended actions. As such, business decisions may be shared between different business processes without requiring them to be specified and maintained in duplicate systems.

Product Highlights

Blaze Advisor From HNC gives business users the ability to change and maintain their own production systems without relying on IT for programming help and without interrupting operations by shutting systems down. This means that business users can build and integrate sophisticated object models incorporating corporate data sources including databases and XML schemas. This is possible through drag and drop visual layout tools where users design complex rule-driven process flows with conditional branching, reusable rule sets, decision tables, functions, and question sets. The end-user gets a personalized view into business processes and portals with user-specific data displays, interactive dialogs, and product recommendations.

Through the separation of business rules from procedural code, Blaze Advisor can help to change the way applications are designed, built, and maintained. By allowing business decision processes to be designed with graphical layout tools and by expressing business logic in simple, declarative statements, the walls between business goals and programming implementation are removed. Development teams can even use Blaze Advisor to design customized web pages giving business policy makers secure control over

decision factors used in system applications. In effect, business policy modifications can be made at any time without involving programming teams and new software releases.

HNC Software touts the concept of Critical Actions through Advanced Science. Critical Actions include such business process tasks as product and service recommendations, risk analysis, identification of sales opportunities, exception identification, and throughput efficiency. The Advanced Science that HNC uses includes progressive technologies such as neural networks and context vectoring, as well as the business rules technology of Blaze Advisor. Business rules are useful in specifying desired system behaviors for any combination of conditions that might be encountered in a transaction. A triggering event can take information from virtually any source and combine it in real time with business rules and the user's individual profile or specific history with the organization.

Blaze Advisor is built conceptually on four pillars. The first pillar is support of BPM. Enterprises run their businesses with repeatable business processes driven by general business rules (with specific rules for specific situations and customers). These processes are fundamental to an organization's ability to execute its business. Thus, supporting the process requirements of business is critical. At their core, rules are simply "if, then" statements. These rules, though, need to work together within a business process—they need to be easily built and manipulated in modular units with the necessary controls built in. In addition, process support requires that rules can look up information from diverse sources and create temporary records to allow calculations and decisions to be collected and stored in memory. Rules need to calculate values and make assessments. Missing and unavailable data events must be handled gracefully; questions may need answering. Finally and most importantly, the right rules must be evaluated at the right time, given the unique needs of the case at hand.

Taken together, these capabilities represent the business process management pillar—allowing enterprises to do business using independent rule services made up of executable, sequenced, declarative rules, rather than being forced to write embedded, procedural computer code.

The second pillar upon which Blaze Advisor is built is individualized response to individual cases. An example of this would be knowing an individual user well enough to know what account information they want to see on their homepage for their bank account. This functionality has become increasingly important, especially in customer service applications. And this functionality is driven by rules, which can personalize applications based on any combination of group profile, individual profile, or transaction history associated with an individual application user or subject.

The third pillar of Blaze Advisor is support of application maintenance. To achieve maximum competitive advantage at minimum cost, companies require faster and easier application modification. As business people discover necessary rule changes within processes, they will need to be empowered with both authority and a toolkit to make the necessary application changes themselves. This can be achieved by giving them access to rule maintenance applications that allow them to maintain the policies, procedures, and rules for which they are responsible.

Blaze Advisor's final pillar is support of enterprise consistency—the goal being to "write once, deploy anywhere." With the number of distributed applications within enterprises growing, it is necessary for individual pieces of business logic to be in more and more places. Increasingly, different applications with overlapping or identical purposes are being deployed over different communication mediums. Web customer self-service applications, call center applications, and mobile applications are good examples—when business rules change, it is increasingly common

that more than one application must be changed. It is good practice to make all necessary applications changes at one time to ensure consistency throughout the organization. By separating business logic from application code, Blaze Advisor rules management technology allows the creation of sharable business logic services. This means that the same business logic (i.e. how to treat platinum customers) can be applied across multiple customer touchpoints including the increasingly important set of mobile devices.

What Rules Management Delivers

Rules management technology delivers a set of very specific benefits to application developers. First, it enables faster application development by separating rules from procedural code. It can also aid in an enterprise's ability to respond to change faster because it gives business staff control over business logic and allows for "write once, deploy in many places" functionality. Rules management can also enable reuse of work—both business policies and application data. Finally, it can help to lower the total cost of application development and ownership.

Rules Management Considerations

There are several factors that are key to successfully implementing rules management technology. First, enterprises must integrate rules with the object models used throughout their existing systems and data sources. Not only will companies be able to take advantage of data in many different locations, they can also extend existing classes and define dynamic classes that can generate temporary objects for use by the rule execution process. Truly optimizing such advantages requires technology built to accommodate such thinking. Blaze Advisor From HNC includes tools to automatically interface with external object models and data sources. Second, companies will need to build up experience in rule service design. Rule services can be designed in different ways—some designs maximize reusability; some designs maximize performance; some designs maximize controlled rule maintenance by multiple groups and individuals. Companies will need to learn which is best for each given situation. Blaze Advisor's visual flow diagrams help to accomplish this task and make it more intuitive. Finally, companies need a keen knowledge of the deployment environment (generally an application server environment such as WebSphere or WebLogic). Knowing the environment well makes for a smoother implementation and thus, better experience with rules management technology. Blaze Advisor addresses this aspect with automated deployment generators that produce integration code for various application servers and execution environments.

How Do Rules Management and Web Services Relate?

This report places much emphasis on the importance of being able to orchestrate Web services. The ability to place rules around where Web services are either created or consumed can allow enterprises the ability to orchestrate Web services into processes. Blaze Advisor can be especially useful in complex processes with multiple business rules that are not always clear-cut. Using Blaze Advisor, enterprises can utilize Web services for even the most complicated processes where it would have seemed difficult to do so.

As previously mentioned, Blaze Advisor is designed to be a callable component of a business application. Thus, in many ways, each rule system designed with Blaze Advisor can be a Web service. There is an automated feature to let users call upon a rules service and there is an easy-to-use Wizard function to wrap and publish each set of rules as a service. But that is not the only relationship Blaze Advisor has with Web services. Because of its SOAP and XML capabilities, Blaze Advisor can access Web services within decision processes, publish Web services, and read and write data using SOAP protocols so the data can be passed to other Web services.

A provider of Web services will have a number of cataloged services that subscribers need to find. Blaze Advisor can be used to create and manage selection rules so consumers can find the right Web services for their needs. This helps to facilitate process efficiency and eliminate the need to spend large amounts of time searching for services.

Blaze Advisor can also help companies with the Web services security issue through connection management. Establishing links to Web services requires authorization and authentication; companies can use Blaze Advisor to build the rules to drive that complex security process.

Finally, Blaze Advisor makes it possible to bring syntax to Web services—applying the "if, then" logic that is necessary to utilize Web services. This functionality can also help Web services to be truly dynamic. It will allow companies to set up rules such that if x condition is true, then subscribe to y Web service, but if z condition is true, then subscribe to b Web service. This can make enterprises more flexible and processes more adaptive to change.

Technical Considerations

Because rules management applies to many high volume applications, such as personalized portals, on-line stock trading, consumer lending, insurance sales, and customer support call centers, it is vital that a rules-based approach can scale.

Blaze Advisor has a threefold approach to scalability. First, design the rule engine to be as fast and lightweight as possible, thereby giving the operating system / Java Virtual Machine / application server more headroom. Having multiple service agents for a rule service is fundamental to provide this required level of scalability. Second, have shared rule networks (rulebases). The Blaze Advisor Rule Server has only a single copy of the rulebase in virtual memory. All active rule agents share this single copy. By using such an approach, memory footprint is dramatically reduced and the Java Virtual Machine is considerably less likely to page swap. Finally, be a "good citizen" in all of the leading application server environments. The philosophy is to leverage the tens of millions of dollars invested by other vendors in scalable, fault tolerant application servers. For HNC Software, this includes testing on, and support for, operations with products such as IBM WebSphere, BEA WebLogic, Microsoft Transaction Server, Sun ONE, and others.

And because rules can be changed while the Rule Server is processing active business events, Blaze Advisor also delivers uninterrupted service. New business events use idle agents associated with the "new" rulebase, while the active sessions are left to complete execution based on the "old" rulebase. After all in-flight sessions using the old rulebase are completed, the old rulebase is purged from memory.

Blaze Advisor is built to function in either a J2EE or .NET environment. The Rule Server component runs on a wide variety of Java virtual machines. Blaze Advisor takes advantage of today's distributed computing environments. A Blaze Advisor-based rule service can provide rules-based processing against either local or remote objects. Remote object access is done intelligently to minimize network round trips.

Information from restricted access applications such as back end ERP systems (accessible through EAI software such as provided by IBM MQSeries, Java1 and others) can be combined with enterprise database data, XML documents and transient, session-based data.

Because Blaze Advisor is built with a Business Object Model Adapter (BOMA) layer, the rulebase is cleanly separated from where the actual objects reside. Through the various concrete implementations with which Blaze Advisor ships (COM, Java, JDBC and XML) or through the BOMA Kit (for other object models such as messaging), the Advisor rule authoring environment is provided with the structure (schema) of an enterprise's business objects. The Blaze Advisor runtime environment has direct high performance access to all properties of enterprise business objects as well as methods, if any. Whether it is a

distributed COM, EJB, CORBA, RMI or database environment, the BOMA feature makes this corporate data available to the rule service.

Blaze Advisor operates as a 100% Java application. This allows a single product release to run on hardware and software platforms from individual users' laptops to multi-processor mainframes. Companies do not have to worry about purchasing different versions of the software for different operating systems or compiler release levels. Blaze Advisor also runs on any of the major Java application servers without modification. A rule service may be called as a logic component from any enterprise application, whether written in Java, COBOL, or C++—the rule execution stays consistent no matter what system calls for it or what machine runs it. This capability allows the decisions at the heart of legacy systems to be renovated for clarity, processing efficiency, and ease of maintenance without having to replace the entire system. Once rule services are defined, they can be reused by additional applications and physical systems throughout the enterprise. If the organization chooses at some point to change its computing architecture and platforms, Blaze Advisor can continue to be used without modification.

Standards Involvement

As part of the Business Rules Traceability Initiative, Blaze Advisor is building XML-based integration with analysis and design tools from companies such as Rational, Ptech, BR Solutions and more. HNC Software is also working with vendors who specialize in legacy code analysis to help identify business rules that have been "lost" in voluminous lines of program code over time. Partnering with leading business rule methodology vendors such as BR Solutions and KPI helps ensure that Blaze Advisor supports best practices in enterprise rule design and definition. Because Blaze Advisor can be used as a complementary technology to workflow and business process modeling tools, partnering with vendors such as BEA, IBM and Vitria allows the creation of integrated services.

Blaze Advisor is also a member of key industry standards organizations and working groups such as the Java Community Process and the Business Process Management Initiative to help ensure adherence to industry-wide standards.

Summary

Blaze Advisor from HNC is a J2EE-based business rules management system. Delivered via an embeddable engine, it provides for the definition and management of business rules and business logic independent of other applications, such as a database. This provides for separation and centralized management of business logic distinct from specific application logic. Blaze Advisor offers the ability to generate Web services based on business rules, as well as the orchestration of existing Web services based on business process logic defined within the rule engine.

Vendor Report: IBM

Overview

IBM's strategy for Web services is at once both holistic and modular—items from across IBM's product line are available to take advantage of Web services, yet customers can also buy only those products they need. An IBM "Web services product" does not exist; rather, support for Web services has been built into many of IBM's product families. This distinction is what makes IBM a true platform provider for Web services.

IBM's strategy for Web services is not just about products—it is about driving the industry as a whole forward. In addition to adding Web services support to its family of products, IBM is helping to lead and support standards initiatives, such as the W3C, OASIS, and WS-I. The company is also using its alphaWorks site and professional services teams (jStart and IBM Global Services) to get code to developers quickly and to support open-source efforts.

The advantage to the customer is twofold. If the customer already has IBM products in its infrastructure, it can build Web services capabilities into the existing systems. If a customer has an infrastructure that is non-IBM, it can still use IBM products to leverage its existing systems and take advantage of Web services.

Through Web services capabilities, IBM seeks to deliver to its customers the ability to conduct dynamic e-business. This means allowing customers to be flexible in responding to market needs while leveraging existing technology investments, a requirement of virtually every organization. IBM's Web services support will help to break down the barriers of incompatible technologies and platforms by utilizing open standards, such as XML, SOAP, and UDDI.

Perhaps most importantly, IBM realizes that Web services are delivered through a network of products and IBM is therefore not creating a "Web services product." IBM's goal is to deliver Web services functionality throughout its family of enterprise software products (WebSphere, DB, Lotus, Tivoli), in addition to driving Web services evolution from lessons learned in IBM Global Services engagements.

A Platform Approach

There are four pillars of IBM's approach to Web services: the IBM WebSphere, a J2EE-based Web application server for execution of Web services; IBM DB2 for managing data and supporting stored procedures; Tivoli for management and security of Web services environments; and Lotus products for collaboration (Domino, LearningSpace, Discovery Server, K-Station, Quickplace, and SameTime). While these IBM infrastructure software families have Web services capabilities incorporated into them, it all begins with the execution environment for Web services—this is why IBM has extended the IBM WebSphere brand to provide this execution environment for Web services.

How WebSphere Delivers Web Services

The WebSphere Application Server is designed to be production-ready—companies can create solutions for dynamic e-business by deploying Web services through WebSphere. By leveraging the SOAP, UDDI, and WSDL standards, WebSphere can deliver interoperability between Web services and J2EE applications. Security issues are addressed via HTTPS support and implementations of XML Signature and Encryption.

Tightly integrated to the WebSphere Application Server is the WebSphere Studio Application Developer. It is a development environment for creating, building, testing, publishing, discovering, and integrating of Web services-based applications that support UDDI, SOAP, WSDL, and XML. The WebSphere Studio Application Developer is built on Eclipse open-source development platform, allowing customers to add third-party tools or to create their own tools. IBM's goal with the WebSphere Studio Application Developer is to provide a set of tools that speeds both the

deployment of Web services, as well as a company's ability to find and integrate a Web service.

The WebSphere Studio Application Developer is complemented by IBM's WebSphere Software Developer Kit. The Developer Kit allows programmers to use the Java language to build and run Web services. It features the basic tools and software that programmers need to build, test, and run Web services. It also features prototypes, which programmers can use as examples for designing Web services. The tool kit, aimed at Linux and Windows developers, complements IBM's existing software and tools, including the WebSphere family of e-business software and WebSphere Studio Java development tool. The kit also includes the basic application server, a database, and a private online directory for listing and finding Web services. The online directory supports the Universal Description, Discovery and Integration (UDDI) specification.

Eclipse

In an effort to address the lack of interoperability among tools that has plagued developers, came the creation of Eclipse, which is the foundation of IBM's newest WebSphere Studio products.

The Eclipse Platform is a functional development environment consisting of a framework that is extendable by plug-ins. It includes a fully functional IDE written in Java, meaning that Eclipse is more than just a collection of APIs. Eclipse has tools that plug into the platform are, providing users with a set of common services that make tool-to-tool integration possible. Eclipse has many extension points that a plug-in can interact with—the plug-ins can interact with each other or with the platform itself.

For IBM customers, the benefit provided by the Eclipse Platform is the ability to create custom IDEs depending on their specific requirements. Currently, more than 1,200 individual developers from over 150 leading software tool suppliers in 63 countries have already participated in the eclipse.org community.

In the Eclipse Platform, code access and use are controlled through the Common Public License1, which allows individuals to create derivative works with worldwide re-distribution rights that are royalty free.

MQSeries

Also important to the Web services equation is the IBM WebSphere MQSeries transaction messaging system and integration server, which leverage the IBM WebSphere product line. The MQSeries products allow enterprises to automatically transmit, receive, filter, and personalize digital information in real-time across platforms and across applications. The messaging system and integration server complement each other, with the messaging system providing a secure, cross-platform transport channel and the integration server linking business operations both internally and externally.

One of the areas where Web services will have a large impact is in application integration. The MQSeries integration software products allow businesses to connect internal operations to business partners and/or customers in real-time. The products connect multiple applications, Web sites, databases, or other information sources within a common framework. The result for IBM customers is an ability to extend their business through the Internet and optimize relationships with partners and customers. At the core of the integration server are integration brokers, which are supported by adapters, connectors, XML, transformation engines, rules engines, workflow, and business process engines. Together, the brokers and supporting mechanisms link operations internal to the enterprise as well as with operations of partners or customers.

Portal Server

The WebSphere Portal Server builds upon the capabilities of the WebSphere Application Server, enabling enterprises to build next generation portals—personalized, secure, single points of interaction with people, content, applications and processes. The portal server gives customers the dual ability to publish WebSphere Portal Server

portlets to a Web services directory and also to search the directory and add a Web service as a new portlet for use in the Portal Server.

This report described earlier the increasing overlap between Web services and portal technologies. Delphi sees portals as a major delivery mechanism for Web services. IBM addresses this evolution with the Portal Server. Customers will be able to use WebSphere Portal Server to set up distributed enterprise portals that will allow dispersed employees and partners to share portlets of all types using Web services.

IBM Web Services Toolkit

For those customers who want to develop Web service solutions that execute atop this IBM infrastructure software IBM provides the IBM Web Services Toolkit (WSTK). This software development kit includes a run-time environment, a demo, and examples to aid in designing and executing Web services applications that can automatically find one another and collaborate in business transactions without additional programming or human intervention. For example, there is a demo of how technology standards, such as SOAP, UDDI, and WSDL, work together.

The basic software components needed to create a Web services environment are provided with WSTK. Included within the kit are an architectural blueprint, sample programs, utility services, and tools for developing and deploying Web services. It also includes a Web services client API that can be used to directly access a UDDI registry, thereby allowing firms to publish directly to public registries or to access external services that already exist in those registries. WSTK can be used with the IBM family of software products to begin creating Web services solutions.

The IBM Web Services Gateway reduces development costs to make selected services available to different divisions within an enterprise or customer and partners either outside the firewall or who use different protocols. With the Web Services Gateway developers and IT Managers can safely "externalize" a Web service so that it can be invoked by users from outside the firewall in a managed, controlled and secure environment.

Delivering Added Value

IBM's strategy for Web services is to provide more than just the infrastructure that WebSphere brings to the game—providing management services and tools is an important part of the equation, as well.

Tivoli

The IBM WebSphere Application Server program provides a solid base for Web services solution creation. IBM's commitment to Web services does not end with the IBM WebSphere family, as several IBM products come into play in delivering Web services functionality. The market overview section of the paper illuminates the need for security and availability in an environment that uses Web services. Tivoli has products available today to help ensure that Web services have a high level of security, availability, and performance management.

Web services standards help to simplify development and integration of management software. Tivoli is a product that interprets and acts upon management data from all sources—including Web services applications. It is possible for IBM to help service providers gain value in the Web services environment. With that in place, they must ensure that their Web services can be delivered via existing standards, WSDL for describing services, UDDI for publishing and finding services, and SOAP for communication between the service provider and the service requestor.

Services providers will then have to deliver on Service Level Agreements that meet the needs of their customers. Doing so will mean that service providers have the infrastructure management products that go beyond the standards of today and deliver an environment for true interoperability. This is where Tivoli comes in—providing the security, availability, and performance products that will allow Service Providers to compete in a world of Web services.

Tivoli manages the IBM infrastructure software that runs Web services. The core set of Tivoli availability and performance management products, in conjunction with Tivoli Managers for the WebSphere Application Server, Lotus, and DB2, give customers management of the applications that execute their Web services. The integration between IBM Directory and IBM middleware products e.g. WebSphere, also provides policy-based security for the Web services environment.

Tivoli Web Services Manager gives service providers and requesters the ability to monitor Web services and to ensure that services are performing within set thresholds. Customers can monitor transactions in real-time or via synthetic transactions.

Tivoli is adapting many of its existing products or building new products to take advantage of the emerging standards that promise to improve interoperability and integration capabilities. Also, Tivoli is aiming to allow groups of customers to provide management across organizational boundaries by allowing service providers to utilize standards to publish vital management information about their Web services.

There are also emerging standards for managing and securing of Web services—Tivoli is working to help define those, as well. Tivoli provides the non-repudiation functionality that is a part of SOAP by leveraging the standards on XML Digital Signatures and XML Encryption. Tivoli has also published a reference architecture, Tivoli Management Extensions for Java, for management of emerging Web services applications built in Java.

DB2

DB2 is a relational database that supports the open Internet standards, such as XML, WSDL, and SOAP, required by Web services infrastructure. DB2, through the XML Extender, allows Web services applications to access data stored in DB2 as XML-structured documents. The advantage to businesses being more efficient in developing new dynamic e-business applications using XML interfaces is lowering costs and increases in efficiency. With DB2 using the Web services infrastructure, IBM aims to help customers integrate their current business processes by sharing information with their partners and improving efficiencies and reducing costs.

Web services access to procedures and data stored in DB2 is provided through tools and runtime software. There are three types of Web service operations that are supported. The first is XML-based—retrieving and storing XML. The second is SQL-based—query and update. The third operation is based around stored procedures invocations.

The DB2 XML Extender provides the support for the XML-based operations by allowing XML documents to be stored or retrieved from a DB2 database. This means that stored procedures can be exposed as Web services, and they can also serve as an integrating technology because they can call out to other Web services.

Lotus

As part of the IBM family of products, Lotus will play a large role in IBM's Web services capabilities. With Web services, the applications running on Lotus platforms today can be integrated with other applications, both within an organization and beyond. Through Web services Lotus can help to enable person-to-person interaction in context with enterprise information discovery and e-business transactions.

A Tool For Lotus Products

Lotus provides a Web Services Enablement Kit, which will include instructions for using the IBM Web services tools to configure custom applications running on Lotus servers to be Web services consumers and/or providers. The Enablement Kit is not a product in and of itself, but rather a for enhancing Lotus software with Web services. The larger Web services strategy for Lotus is to add on Web services modules to the collaboration, knowledge management, and e-learning products.

The value proposition is that applications, such as Domino, can be utilized by customers as Web services. Domino, as a workflow tool, can orchestrate Web services created in the Websphere environment or a trading partner or supplier can invoke a complete Domino process as a service. Again, the benefit will be the flexibility to do business in the most efficient manner possible.

Several Lotus products play a role today with respect to Web services:

- Lotus Domino Application Server—Domino collaboration, workflow, and messaging capabilities are accessible as Web services by adding SOAP interfaces and WSDL descriptions to new or existing Domino applications, using XML, with the native XML capabilities in Domino;
- Lotus Domino Workflow—works on top of Domino, providing the ability to develop, manage, and monitor enterprise-scale, human-interactive business processes. By adding SOAP interfaces and WSDL descriptions to existing and new workflows, developers can create Web services that allow external applications to use Domino Workflow-based applications over the Web;
- Lotus Knowledge Discovery System—developers can add SOAP/WSDL interfaces over the expertise database of the Knowledge Discovery System, enabling the information to be served up as a Web service. The Knowledge Discovery System has two components. First is the Lotus K-station, a knowledge portal with collaborative capabilities that uses a browser-like user interface to access information sources. Second is the Lotus Discovery Server, a knowledge server enabling search and expertise location;
- Lotus Sametime—a real-time collaboration solution combining chat, whiteboarding, audio, video, and application-sharing capabilities that enables immediate between people anywhere. Sametime Everyplace extends Sametime's real-time capabilities to mobile devices. Using Java APIs and Beans, developers can build Web services that incorporate online presence awareness and instant messaging as part of any Web services-enabled system;
- Lotus LearningSpace—a distance learning platform enabling integration of live, asynchronous, and self-paced course content delivery. Using Java APIs and Beans, developers can build Web services that incorporate LearningSpace capabilities, such as course lists or schedules, as part of any Web services-enabled system;
- The Lotus Web services Enablement kit is designed to further demonstrate the development of Lotus-based Web service applications. The kit includes instructions for using the IBM Web services Toolkit to configure custom applications running on Lotus servers to be Web service providers and/or consumers; it also includes sample Web services applications.

Partnering Program—Web Services on WebSphere (WoW)

Through WoW, IBM is developing a partner community that will focus on building Web services solutions with the goal of enabling customer development and deployment of dynamic Web services solutions. At the core of this initiative is IBM's application server, WebSphere and its three partner constituencies. The first partner constituency is the application software vendors. IBM will provide the technical, business, and marketing ability to help these vendors rapidly adopt Web services on WebSphere then quickly announce and deliver WoW-enabled products and services.

The second partner constituency is systems integrators. Web services content, certification, and support have been added to current programs in order to give systems integrators the ability to add Web services to their existing WebSphere practices.

The third constituency is complementary technology software vendors. The WoW initiative

will provide complementary technology software vendors with the products and skills they need to develop the highest quality support for WebSphere. In this way, customers will be able to leverage investments they may have already made in complementary technologies.

Additionally, there is a WoW Advisory Council that guides and directs development efforts. The advisory council consists of a small number of key members of the three partner constituencies. The functions of the advisory council are multifold. The council represents the broader community's technical and marketing needs and acts as the primary feedback conduit to IBM. It also provides business enablement guidance to the broader community in an effort to smooth the transition to Web services business models.

The Advisory Council also provides thought leadership for the WoW community, driving constant innovation. Finally, the council develops a set of best practices for use by the greater WoW community to model, design, build, develop, deploy, change and manage Web services solutions.

IBM and Grid Computing

IBM is one of the first companies to begin acting on the synergy that exists between Web services and grid computing. The company realizes the effect that grid computing could have on eUtility—the utility model for delivery of software will allow businesses to use simple front-ends to access the software they need, and pay for it on usage or contract basis rather than buying it outright. This model is also expected to provide different levels of quality of service (QoS) with corresponding fee structure in their service-level agreements (SLAs).

IBM and the Globus Project (a multi-institutional research and development effort creating fundamental technologies for computational grids) have developed a set of new specifications that allow businesses to share both applications and computing resources over the Internet, making grid computing applicable to real business situations. The specifications are referred to as the Open Grid Services Architecture (OGSA)—a set standards that combine the benefits of grid computing and Web services.

Using the OGSA, customers can access and share computing resources on demand over the Internet, relying on an infrastructure that is resilient, self managing and always available. The goal of this initiative is to allow customers to integrate applications and share data and processing power while capitalizing on potential cost and efficiency savings.

This set of specifications builds on standards such as XML,WSDL, and SOAP for grid computing, which are used to locate, schedule and secure computing resources. IBM intends to leverage OGSA as a key foundation in its Project eLiza initiative. Project eLiza is IBM's Autonomic Computing initiative to build an open, heterogeneous, self-managing server infrastructure for e-business and commercial grid implementations.

IBM and Standards Development

For Web services to have a real impact on businesses, standards must be developed and adopted. IBM has taken a leadership role in developing the specifications needed to create Web services standards. IBM donated the Web Services Invocation Framework, and the Web Services Inspection Language for Java, two Web services technologies, to Apache, a nonprofit organization best known for creating a popular and widely used Web server technology that delivers Web pages to browsers.

The Web Services Invocation Framework allows Web services to travel across different network protocols. It supports Simple Object Access Protocol (SOAP) , a communications format for businesses to link and conduct transactiona online, but it also supports other ways of sending messages, including Java Messaging Service, Remote Method Invocation, and even instant-messaging software. For example, this framework will allow simple Web services, such as stock quotes or weather reports, to be sent as instant messages.

The Web Services Inspection Language for Java, provides a working version of the WS-Inspection specification created in 2001 by IBM and Microsoft. WS-Inspection creates a uniform way for companies to find Web services by connecting them to one another's Web sites and complements the UDDI specifications. While UDDI acts like the "yellow pages", where businesses can find a list of companies that cater to their needs, WS-Inspection is for businesses that already know which companies they want to work with and how to contact them but want to see what Web services they offer.

IBM has also created WSFL and HTTPR, and has worked with other vendors to make SOAP vendor-neutral, to create WSDL and UDDI.org, and to propose changes to HTTP that would make it a reliable protocol as an underpinning to SOAP messaging.

IBM Professional Services Teams

IBM's Web services capabilities as represented through its family of products and solutions are driven, in many cases, by advances in the professional services groups. At IBM, two groups are capitalizing on Web services. IBM Global Services (IGS) is working with numerous clients on heterogeneous systems and seeing the need for cross-platform computing. IBM jStart is a group that is focused on helping customers and partners embrace new and emerging software technologies. The jStart team helps customers design and deploy Web services applications for both EAI and B2B purposes, focusing on pilot projects.

Case Study: Storebrand

Elimination of data entry errors results in more efficient pension plan management for both customers and payroll vendor partners.

Storebrand is Norway's largest provider of pension plan, insurance, and other financial services—more than 390,000 employees of 6,500 companies in Norway alone are members of Storebrand pension plans. With that many individuals involved in the pension program, keeping information up-to-date becomes a major challenge that affects everything from process performance to customer service.

Already, Storebrand kept the data for all its plan members in an IBM DB2 Universal Database on an IBM S/390 Parallel Enterprise Server. The process for changing a member's information within the database was cumbersome. The employer would have to manually revise its records followed by sending updates to Storebrand through FTP (file transfer protocol), snail mail, or fax. Storebrand customer service representatives were then tasked with manually entering the changes into the DB2 database.

This process left ample room for human error, both by employers that are manually changing information and by Storebrand CSRs that are manually reentering information into databases. In addition, two sets of manual entry constitute a sign of redundancy within the process. Storebrand set out to solve this problem by automation the capture of this data, with the goal of reducing costs and errors, improving customer service, and enhancing the company's leading-edge image.

Storebrand decided that the best way to automate the capturing and updating of customer information changes was to create a Web service that its clients' payroll applications could access over the Internet. Storebrand was already using WebSphere software in other lines of business; therefore, it made sense to choose IBM to build the infrastructure for creating and deploying Web services.

IBM offered Storebrand the right mix of products and professional services necessary to build a Web services that could accomplish Storebrand's goals. The IBM jStart program helped Storebrand to leverage its WebSphere application server, and then to use Lotus Domino, IBM MQSeries, and MQSeries Integrator to create a secure channel for its applications to interact with those of its partners. Using this channel, the pension plan

updates can travel directly over the Web through the WebSphere application server and into the DB2 pension plan database.

There are three main components to this project: a Common Object Model (COM) object that runs within the employer's payroll application; the Java-based business logic residing in the application server (WebSphere); and an integration layer based on MQSeries that transforms incoming data to make it DB2-ready.

IBM's Web Services Tool Kit (WSTK) was critical to the project. Storebrand used IBM VisualAge for Java to create the business object for the payroll application. Still, though, it would have to be wrapped in WSDL for it to be accessed as a Web service. WSTK gave Storebrand the ability to quickly create the WSDL wrapper and to publish and deploy the Web service without having to do any reprogramming.

Now, when employers make changes to the information in their payroll applications, the changes can be automatically reconciled with Storebrand's records. How does this happen? The payroll application extracts the data and translates it to XML. Storebrand's COM object then generates a SOAP request that access the Web service through the Internet. Back at Storebrand, the Lotus Domino system authenticates the employer sending the information, then sends the XML data to a business object in the WebSphere application server, which routes it to a queue in MQSeries, thereby triggering a message flow in MQSeries Integrator. The Integrator then passes the information to the server.

The benefits of this Web service are multifold for Storebrand. Very quickly, there have been savings in customer support costs through the elimination of data entry errors. Both customers and payroll vendor partners are pleased with this technology advancement—customers have an easier time managing their Storebrand pension plans and payroll partners can enhance their products with this Web service.

This project also positions Storebrand well to take advantage of Web services technology in the longer-term. It is possible for Storebrand to extent the architecture it created for the pension plan application to serve its other lines of business. This is the flexibility and ability to respond quickly that enterprises have been chasing. Furthermore, Storebrand can publish its Web services for others to access by utilizing WebSphere's ability to integrate with UDDI registries. This ability is but one of the several revenue-generating opportunities that exist due to this project.

Summary

The benefits derived from Web services solutions can drive businesses to new levels of efficiency. The combination of IBM's products and services in the Web services realm make it possible to begin realizing this efficiency now, without the need to learn new programming languages. IBM's offerings allow enterprises to leverage their existing software systems and to integrate those systems together, both behind the firewall and with partners.

IBM is pushing for the universal adoption of standards in an effort to bring true interoperability and platform independence to the market. That leadership, the experience of professional services teams such as IGS and jStart, and a platform offering consisting of Web services infrastructure and tools, positions IBM as a leader in the Web services marketplace.

Vendor Report: Microsoft

Overview

Microsoft's Web services strategy is born out of the .Net initiative. The goal is to be a platform provider for Web services, providing infrastructure products and tools for Web services utilization. The ability to deliver software as a service holds so much potential that .NET has become the centerpiece of Microsoft's corporate strategy.

As a platform provider, Microsoft weaves together several product offerings to offer both infrastructure and tools for Web services utilization. Products that are central to Microsoft's vision for connected software include servers, developer tools and smart clients and devices.

Microsoft Servers and .NET

As with other platform providers, Microsoft's Web services capabilities grow out of its base of servers. Because of deep support for XML, .NET servers accelerate the integration of systems, applications, and partners. This means that enterprises can build upon legacy systems that are already in place, rather than replacing them with newer, more expensive systems. Through simple access to mainframes with Microsoft Host Integration Server and automatic conversions of existing data formats into XML with Microsoft BizTalk Server, the existing infrastructure within an enterprise can be set up to utilize Web services.

As mentioned earlier in this report, enterprises will utilize Web services to deliver key processes through the organization. Through the Microsoft .NET platform, businesses will be able to turn key processes into Web services. As the Web services model takes off and distributed computing become commonplace, server infrastructures will be tested.

The .NET platform, by distributing computing power, demands an agile, reliable server infrastructure, where secure and scalable servers that integrate XML will be the backbone of the platform. Microsoft has a number of server products designed to meet those challenges—Microsoft Windows 2000 Server, the Windows .NET Server family, and the Microsoft .NET Enterprise Servers.

The .NET Enterprise Server family includes several servers, each with distinct functionality. Microsoft, too, recognizes many of the business touchpoints where Web services can have immediate impact, such as application integration and e-commerce. Thus, there are servers that cater to those functionalities.

- The Microsoft Application Server allows enterprises to deploy and manage Web applications—these applications can be deployed as Web services and accessed through the Web by users.
- Businesspeople can utilize Web services to conduct processes through the Microsoft BizTalk Server, which builds XML-based business processes across applications and organizations.
- Enterprises can use the Microsoft Commerce Server to build scalable e-commerce solutions.
- The Microsoft Content Management Server allows enterprises to manage content for dynamic e-business Web sites.
- The Microsoft Exchange Server enables messaging and collaboration anytime, anywhere communications.
- To bridge to data and applications on mainframe legacy systems, enterprises can rely on the Microsoft Host Integration Server.
- Enterprises can enable secure and fast Internet connectivity via the Microsoft Internet Security and Acceleration Server.
- The Microsoft Mobile Information 2001 Server enables application support by mobile devices like cell phones.
- Through the Microsoft SharePoint Portal Server 2001 employees within enterprises can find, share, and publish business information.
- Finally, enterprises can utilize the Microsoft SQL Server to store, retrieve, and analyze structured XML data.

Building Web Services

Earlier in this report, Microsoft Visual Studio .NET was mentioned as an example of an IDE (integrated development environment). Visual Studio .NET complements the .NET framework, giving developers the power to build, deploy, and run Web services. Importantly, it allows developers to use their existing skills to create Web services for a range of devices; developers can leverage existing skills, because the .NET Framework's common language runtime allows for the development of Web services using any modern program language.

Microsoft Visual Studio .NET

Visual Studio .NET advances several programming languages by being able to use them for the creation of Web services. Among them are: Microsoft Visual Basic, which includes new object oriented programming features; Microsoft Visual C++, which advances Windows development and enables the building of .NET applications; and C#, which brings RAD to the C and C++ developer.

Visual Studio .NET provides a single, unified development environment. Built on the .NET Framework, it provides support for working with Web services created in all modern programming languages. Applications and Web services created in one language can be programmed against and debugged in any other language supported by Visual Studio .NET. This enhances the ability to use existing Web services to build new and exciting solutions.

Visual Studio .NET automatically creates the necessary XML and SOAP interface needed to turn an application into a Web service. Developers can concentrate on building the application, not on the plumbing for the Web service.

Developing with Web services is similar to developing with components. Visual Studio .NET gives developers the ease of importing Web services or using Web services hosted remotely and programming against them as they would a COM element today, saving time and giving developers the opportunity to concentrate on core functionality.

Microsoft .NET Framework and Microsoft .NET Compact Framework

The .NET Framework, and the device-focused .NET Compact Framework, are high-productivity, standards-based, multi-language application execution environments that handle essential plumbing chores and ease deployment. The application execution environment manages memory, addresses versioning issues, and improves the reliability, scalability, and security of your application. Components include the common language runtime, a rich set of class libraries for building Web services, and ASP.NET.

The common language runtime is the engine in the .NET Framework that provides a managed, secure execution environment and is designed to support developers using many different languages to create applications. It has a unified type system and enables cross-language inheritance and debugging. By using the .NET Framework developers have the quickest and most productive way of building applications that are truly 3rd generation Web services and applications.

The Components of Microsoft .NET Connected Software

.NET is simply a name used to describe the Microsoft platform of infrastructure products and tools that provide the ability to build, host, deploy, and utilize secure and connected solutions using Web services.

An important aspect of Microsoft's Web services strategy is allowing a variety of devices to access and utilize Web services. To enable this, Microsoft provides "smart" client application software and operating systems. The smart clients allow users to take advantage of Web services when and where they need to and remove the bottleneck of server processing power by distributing computing across the network.

The evolution of technology has been driving to a place where people can get the information they need to do their jobs no matter where they are or what is in front of them (i.e. a PC or wireless

device). The .NET platform aims to deliver this type of computing experience by allowing developers to build rich applications that integrate with Windows and the Internet. The result is "smart" client software.

The smart clients refer to PCs, mobile devices, or any other type of applications that users interface with. Microsoft hopes to use these devices to enable more personal user experiences on the .NET platform. Through Microsoft Windows XP, Windows XP Embedded, and Windows CE .NET, Microsoft is creating this next generation of software clients to enable anytime, anywhere connecting. The key to this computing environment becoming reality is the .NET framework's managed, secure execution environment—as pointed out earlier in this report, security is a major issue for software buyers in regard to Web services.

Device-Ready

Windows XP, the latest version of the Windows operating system, integrates the use of XML more deeply than previous versions. It is an engine that allows users to access the capabilities of the .NET platform from a both laptop and desktop computer users. The enhanced support for mobile computing built into Windows XP makes it easier to roam between networks and connect remotely, allowing the PC to become a hub for activity on the .NET platform.

Built to deliver a robust real-time operating system for small footprint devices, such as handheld Personal Digital Assistants, Windows CE .NET enables developers to quickly bring to market smart connected devices that take advantage of the latest wireless and multimedia technology. It's purpose is to extend the reach of the .NET platform.

Windows XP Embedded, the componentized version of the Windows desktop operating system, delivers a platform for a broad range of smart client devices such as set-top boxes, retail kiosks, and thin clients. Windows XP Embedded enables developers and device makers to take advantage of the many new features in Windows XP and quickly bring to market rich smart devices based on x86 microprocessors.

Smart Devices and .NET

Why call these devices and clients "smart"? Because they are built to utilize information and simply the jobs of users. They provide a simpler, yet personal experience by knowing user preferences, utilizing the .NET identity, profile information, and data to customize the user's experience. They also know about the network and bandwidth constraints, allowing them to "smartly" shift in between offline and online experiences. From a businessperson's perspective, smart devices allow for increased reach—they can discover the presence of other devices on the network and Internet, as well as make themselves available for interactions. Perhaps most importantly, smart devices present and gather information in the most appropriate form for that device—from converting text to speech on a cell phone to recognizing handwriting on a Tablet PC. Using XML, SOAP, and UDDI, smart devices are able to consume Web services as well as understand which services are available.

Smart devices are PCs, laptops, workstations, phones, handheld computers, Tablet PCs, Microsoft XBox game consoles, and other machines that have not yet been invented. What makes these devices "smart" is their ability to access Web services, enabling users to interact with data regardless of the location, type, and number of devices used.

Smart Devices Built on Microsoft Technologies

Microsoft has developed several smart devices that go beyond the PC one typically thinks of when talking about a computing environment. From the Pocket PC to the Smartphone to the Xbox and Tablet PC, Microsoft is continually looking to provide new ways for users to access Web services.

Microsoft and others are developing a core set Web services that can be combined with other Web services or used directly with these smart client applications. For example, Microsoft MapPoint

.NET is a Web service that enables the integration of maps, driving directions, and other location intelligence into applications, business processes, and Web sites. By having smart devices that can interact with this Web services, business increase the number of ways in which they can push information out to users.

Technology for Tomorrow

While Microsoft is helping to lead the way in the Web services evolution today, its technologies for tomorrow are on the horizon. The new Microsoft technologies, code-named "TrustBridge," will appear in products next year. The aim of TrustBridge is to provide the security that many software buyers perceive as missing in today's Web services environment.

TrustBridge will provide single sign-on access to multiple companies' information so that two companies can trust each other as they exchange Web services. For example, a corporation outsourcing human resources functions to an outside company could easily allow its employees to go to the outsourcer's Web site to access benefits and other information. The two systems would share user identity information and data on which users are able to access which applications.

Summary

Microsoft has the products today to enable Web services utilization in business execution. As a leader within organizations like WS-I, Microsoft is helping to drive the standards that will define interoperability in the near term. The sheer breadth of Microsoft's products ensure that its Web services strategy will be holistic, delivering interoperability between applications no matter where they exist in the future. Through efforts such as TrustBridge, Microsoft is working to ensure the secure environment software buyers want from Web services. It's clear that Microsoft is intent on being a leading provider of products and services for the Web services computing paradigm.

Vendor Report: Nobilis

Overview

In the near-term, one of the biggest challenges for Web services (as a technology sector) will be gaining traction within enterprises. No matter how well infrastructure providers evolve their products for creating innovative Web services, unless enterprises can harness the power of those services and realize business benefits, Web services will be dead in the water. This is where Web services orchestrators will play a significant role in the evolution of this technology.

All businesses are run by a series of processes. Each process executes a given task in the hope that the conglomerate of processes creates value that an enterprise can sell. Web services, therefore, need to fit into processes in order to impact the organization. Web services orchestrators integrate Web services into processes. BPM vendors have been quick to begin getting their products Web services-ready. One such vendor is Nobilis.

Nobilis provides a BPM platform that allows business users to not only automate their processes, but also to practically work Web services into those processes without the need for programming. It is important to note that Nobilis gives the business a graphical user interface to define the usage model for Web services and to manage the interactions. Businesspeople are responsible for process performance; thus, they need to define how Web services will run within their processes.

The Nobilis Enterprise Platform

The Nobilis Enterprise platform goes beyond automating strictly system based processes to support the automation of people-oriented processes. In order to accomplish this goal, there are several critical capabilities that converge. First, Nobilis Enterprise allows users to visually build applications quickly and run those applications immediately. Users can build a process using a drag-and-drop interface, replete with a repository of process templates and a library of business rules. The differentiator lies in the user's ability to then run that process immediately, as opposed to being required to wait for a developer to code it for execution. This capability allows businesspeople, who are responsible for process execution, to control their own destinies and to operate quickly and flexibly.

The business rules library available to users designing processes is more than just a listing of the available process rules, though it is that, too. Each rule, whether it is one provided out of the box, or one developed and registered with the system by IT, is a software component that can be reused throughout various and diverse processes within an enterprise. The rules library is also a control mechanism, ensuring that each rule is applied in a consistent and controlled manner across the organization. Because all rules are extensible, each can be configured for the constantly changing requirements of the enterprise.

Always important when designing processes is the ability to integrate data, no matter its format, from disparate systems and sources, from one single point of entry. As shown in this report, this integration is one area where Web services can have an immediate and significant impact. Support for data integration is built into the Nobilis Enterprise platform—the result is a series of data objects that allow process integration of information from back-end systems, data warehouses, legacy applications, and message brokers.

A must for any BPM tool is the ability to support data-driven or event-driven workflow—BPM is really the next step in the evolution of workflow. The Nobilis Enterprise platform creates an environment where activities are initiated upon events or conditions, without requiring human intervention.

It is one thing to automate simple processes, such as where data is simply relayed from one application to another. When humans are necessary for process completion, the challenge becomes a bit more complicated. Rather than simply alerting people to conditions requiring their attention, Nobilis empowers them to take

action. Where a process requires a person to take an action, such an action is facilitated through the delivery of a form or application, rather than a simple notification message. A process agent controls and manages the user to complete the activity according to business rules.

A BPM tool must also be able to control data and allocate it only to those authorized to use it. This is in addition to the actual delivery of the data to the correct person within the process. The Nobilis Enterprise platform maintains control over data ownership, parsing out only the data required for each participant to complete their piece of an activity. The data is tracked and controlled through any process extension all the way to completion. Furthermore, the Process Agent maintains full control over information and adherence to business rules throughout any extension of processes to additional participants.

No BPM tool is complete unless it offers the ability to monitor processes in real time, enabling enterprises to adapt to process needs. With Nobilis Enterprise, administrators are provided an up to the minute history of the executions of all process activities, identifying bottlenecks and activity completion delays.

Nobilis Enterprise Components

The strategy, then for Nobilis Enterprise, is to empower any business user in the value chain to visually select data, apply process, calculation and validation rules, and automate an interaction with any partner, supplier, or customer in the value chain. IT provides the "sandbox" of rules, components, and data connectivity, and the business user plays in it - designing and running processes immediately. Nobilis tracks and controls the collaborative interactions between participants through any medium—email, portals, PDA's, pagers, or wireless devices. Nobilis Enterprise provides process automation that integrates people, processes and data—and delivers it using an advanced, object-based architecture.

To meet these critical BPM requirements, there are several components to the Nobilis Enterprise platform. First is the Nobilis Enterprise Process Designer, which enables users to model business processes by defining the relationship between people, business rules, documents, and data views for any given process. The interface is a graphical, drag-and-drop one that simplifies the job of the user.

The next component of the Nobilis Enterprise platform is the Data Manager, which defines the working data schema for each workflow process. It then enables data from disparate physical sources, in multiple formats, to be accessed from a single point of entry. The result is a series of data objects that allow process integration of information from back-end systems, data warehouses, legacy applications, and message brokers.

The Nobilis Enterprise Document Store is a shared collection of information (data sheets, instructions, installation guides, specification sheets, reference manuals, general information documents, videos and more) that is managed and controlled by Nobilis administrators. For example, an Excel forecast entry sheet, instruction sheet, and company policy manual can be grouped together as a package and distributed to all users of that workflow. Nobilis allows for the centralization of these documents while maintaining accessibility for those with security to do so.

Another crucial component of the platform are the Nobilis Enterprise Business Rules, which enable users to define rules for calculations, data translations, and validations, without the need for programming. There are three types of business rules within Nobilis Enterprise Business Rules. First are validation rules, which automatically validate data associated with specific users or entire processes (i.e. data entry verification or data comparison validations). Second are calculation rules, which perform simple to complex data calculations, including conditional profitability calculations and RFQ response vendor rankings. Groups of calculation rules may be combined to create specific calculation libraries. Finally, there are process rules, which are pre-built and custom software components that handle tasks ranging

from simply distributing documents to invoking external applications or interfacing with legacy systems. These rules are implemented using a visual deployment environment in the Nobilis Enterprise Process Designer and enable the user to interconnect process maps with sub processes to easily create complex conditional workflows.

The Nobilis Enterprise Process Agent enables automated execution, monitoring and reporting, and dynamic extension of user-defined business processes. It is a highly scalable, multi-threaded process engine that executes simultaneous workflow processes for even very complicated business models. Once process maps have been defined by the user in the Nobilis Enterprise Process Designer, the Process Agent controls their automated execution, interfacing with systems, applications, and data sources to retrieve information, deliver applications and documents, evaluate responses, calculate models and rules, validate data, and forward responses and outcomes to sub-processes.

The Nobilis Enterprise Process Agent supports collaboration within a process by monitoring the additional people and data that are dynamically added to the Process Map to insure control and administration of the process, even when changes have been made. For example, a process participant can carve out a portion of the data he has received and extend the collaboration to a sub-participant.

The Nobilis Enterprise Process Agent also includes a Process Monitoring component that logs and tracks all of the steps and events that have been executed in a given process map. It tracks for errors, such as responses failing validation rules, and provides real-time status information at any step of the process. For example, in a collaborative sourcing application when RFQ's have been sent to a group of suppliers, the Process Agent monitors and tracks where each supplier is in the process of responding to the request. Process maps can include specific actions that the Agent would kick off in response to monitored conditions, such as sending a message to a supplier prompting them that they were approaching the deadline for a quote to be considered. Finally, reports can be generated to show monitored status of a process, such as a report displaying which suppliers had submitted responses and which continued to remain outstanding.

The Sum of Components Equals a Platform

Together, the components of Nobilis Enterprise provide a foundation for process automation and integration across the enterprise. They fulfill the requirements of a holistic BPM system by allowing for process design, monitoring, and operation (automation and integration).

Where Web Services Fit In

Nobilis Enterprise is a BPM platform that allows enterprises to orchestrate Web services. As discussed in this report, Web services represent exciting new possibilities in technology; however, unless these Web services can be harnessed to provide value to an enterprise, they are worthless. This is where orchestration comes in—putting Web services to work within an enterprise's processes.

The Nobilis Enterprise Platform provides multiple pre-built components and process activity templates to better enable users to search for, approve, manage, use, and monitor Web services within any automated business process. With Nobilis Enterprise, the orchestration of Web service usage may be defined by end users themselves, without requiring any programming.

The platform provides a new set of components and process activity templates to enable IT resources and business users alike to truly leverage Web services as part of their daily activities. Without programming, these components allow administrators and users to conduct multiple tasks more easily. Users can search for services based upon multiple criteria across public and private UDDI directories. IT can approve which services will be provisioned to end users via exposed private directories. They can manage the business

model-defining interaction with a service, including the cost structure and service level agreement surrounding usage. They can use services within any given business activity by defining the logic controlling how and when to deploy services, and which service should be picked for which conditions. And finally, users can monitor interactions with a Web service to track performance metrics, service levels, and costs.

With the Nobilis Enterprise platform, companies can now have a tool to enable business users to drop a Web service into business processes and to interact with productivity tools they use on their desktop. This helps to make Web services a part of everyday business activities—meaning users get comfortable with utilizing these Web services to better execute business.

Case Study: Multinational Telecom Company

Web services provide means to significantly reduce costs associated with expense reporting.

Problem

A multinational telecom company needed to improve its expense reporting process. 50,000 employees worldwide used a standard MS Excel form to report their expenses and a host of problems accompanied the process. First, the diversity of business units meant that each expense form was not approved in the same manner, following the same procedures. Because approvals were based on managers manually approving forms, it was impossible to manage adherence to corporate policies through a centralized process. Clerks in Finance spent countless hours checking reports for adherence to corporate standards and manually routing mistakes and issues back to the issuer and the associated manager for resubmission and approval.

Likewise, given the global nature of the business, most expense reporting encountered the need to convert currencies from travel across multiple countries into the currency of the country of reporting origin. Because the process was Excel based, there was no central, consistent currency conversion methodology that could be standardized and controlled. In the end, because of these issues, results were untimely and numbers were often suspect.

Solution

The company needed to solve these issues of inefficiency and bloated cycle time, as well as the lack of financial consistency, but it did not want to give up its familiar Excel based form. Rather than implement a new system and require the entire company to learn it and adapt to it, they turned to a solution from Nobilis that would solve their problems, but allow them to continue using their familiar Excel based form.

The Nobilis platform enabled a business manager in Finance to leverage their familiar Excel form as the means of entering expense information, but wrap process automation around it to enterprise-enable the required routing, controls, and approvals. When a user needed to fill out an expense report, they would start an instance of the process and be presented with the Excel form as the entry mechanism. Once they submitted the form, the process would route the report for approval based upon any hierarchy of rules determined by the corporate finance department and the local business unit in question. These included such rules as approvals required for total expenses or for exceeding certain thresholds on line items. All of these rules could be adjusted and maintained by the appropriate business manager without requiring the intervention of IT. Once approved, the data from each report would be automatically deposited into a database for reporting purposes.

In addition, the company used Nobilis components that came out of the box to leverage a web service for currency conversion. Using Nobilis, the company's IT department was able to search a

UDDI directory, provision the desired currency conversion service for use within a Nobilis process, manage the business model for interaction with the service, and monitor its usage on an ongoing basis. The business manager tasked with controlling the expense reporting process was able to treat that web service, which had been exposed to him by IT, as he would any other component within the Nobilis system. When he came to the step in the process to insert currency conversion, he simply dragged and dropped the web service into the workflow and configured its usage for that process. When users started the expense reporting process, they would specify the currencies requiring conversion. When the user was presented with the form, the appropriate conversion rates would be drawn from the service and the totals on the sheet would automatically reflect the proper conversion.

Conclusion

Today, Nobilis has dramatically improved the efficiency and timeliness of the expense reporting process and the accuracy and consistency of the end numbers. Because the Nobilis system leveraged the original excel form, there was no learning curve for participants to climb in implementing the system. Because the system allows for controls, deadlines, and escalations, cycle times for processing and approving expense reports have dropped from weeks to hours. Because currency conversion is handled by a web service, inconsistent application of rates has been eliminated. The numbers are more timely, more accurate, and over $300,000 of manpower annually dedicated to the process has been saved to be better spent elsewhere.

Summary

Web services orchestration may prove to be the "make-or-break" technology of the Web services paradigm. While it is important to have the tools to create Web services and the infrastructure to support Web services, ultimately, a business will only benefit from the technology if there is a standard way to consume Web services such that they make an enterprise more efficient. This is where Nobilis comes in, offering a BPM platform that can automate processes across the enterprise and weave Web services into those processes.

Vendor Report: Sun Microsystems

Overview

Sun Microsystems has developed its Web services initiative based on the notion of "giving users what they want, when they want it" under banner of a strategy it as labeled as "services on demand." The framework through this strategy is delivered to the market is the Sun ONE (Open Net Environment) platform.

Sun describes services on demand as "the point where information assets are fully leveraged for business benefit in the form of services." In the context of the Sun ONE framework, it enables access to information assets on the basis of both Web applications provided as services, and components of applications stored in central directories and assembled in real-time in the form of composite applications.

The Sun ONE platform is designed to address a number of issues surrounding the delivery of software as services, which include: providing a unified view of community, transactions, and infrastructure; a framework for tracking IT progress and management issues; reuse of software components, applications, and services, as well as a variety of options for accessing Web services.

The specific resources to address these issues are delivered through the three main product lines which combined make up the Sun ONE platform: Sun ONE Studio 4, Sun ONE Infrastructure Software, and the Solaris Operating Environment. Each of the three and their constituent components are examined in greater detail below, vis-à-vis the standard framework used in this document for comparing Web services software.

Development

Sun ONE Studio 4 (formerly Forte for Java) provides an IDE (Integrated Development Environment) for Java technology developers to develop and deploy EJBs (Enterprise Java Beans) as well as publish Web services, which can be done from existing EJB components and those newly created within the IDE. Sun ONE Studio 4 integrates to the Sun ONE Application Server for publishing Web services, helping to shorten development cycles.

Infrastructure

Sun ONE Infrastructure Software (formerly known as iPlanet) combines several products that address the management of Web services. These include applications for directory services and security (discussed below under "Identity Management"), as well as application integration (discussed below under "Integration").

The Sun ONE Application Server provides an environment for the development, deployment, and management of application services for a broad range of servers, client, and devices. Designed for the J2EE framework, the Sun ONE Application Server aims to deliver the benefit of application reuse and developer collaboration.

Integration

The Sun ONE Integration Server, EAI Edition, targets the integration potential of Web services. Its role is to integrate packaged, custom, legacy, and new Java applications to build automated business processes across multiple and distributed disparate environments. Taking this concept outside of the firewall is the Sun ONE Integration Server, B2B Edition. This application automates and manages processes that occur between organizations over the Internet and existing private networks, thereby facilitating Internet commerce exchange. Both products are built upon the foundation provided by the Sun ONE Message Queue, a J2EE compliant Message Oriented Middleware (MOM) platform.

Identity Management

As previously mentioned in this report, security ranks as chief concern among software buyers evaluating Web services software alternatives.

Sun addresses security vis-a-vis the Sun ONE Identity Server, which manages secure access to web-based resources. It provides an identity system that includes access management, identity administration, and directory services. The Sun ONE Identity Server also integrates with the Sun ONE Portal Server to provide Web Single-Sign-On capability. In the near future, it is stated to provide cross-domain Single-Sign-On functionality through support for the Liberty Alliance standard specifications expected this summer.

In addition to the functionality offered by the Sun ONE Identity Server, the Sun ONE Directory Server stores and manages identity profiles, access privileges, application and network resource information. And the Sun ONE Directory Proxy Server routes requests from clients to the appropriate directory server automatically through a referral mechanism and provides security services for the Sun ONE Directory Server. Furthermore, the Sun ONE Meta Directory consolidates and aggregates identity information from such sources as customer databases or human resources applications to build a single, unified view stored in the directory.

Presentation

As a platform vendor, Sun offers both the infrastructure to create and deploy Web services, as well as the tools for users to access them. The Sun ONE Portal Server allows users to deploy e-commerce portals that include membership management, personalization, aggregation, security, integration, and search services. Through these portals, users can access available Web services for use in the execution of business.

The Sun ONE Portal Server also features the Mobile Access Pack that enables Web services and content to be delivered to mobile devices. In addition, the Secure Remote Access Pack provides VPN-like functionality without having to install additional VPN (Virtual Private Network) client software. By way of a standard Java-enabled browser along with a SSL connection, remote employees and partners can access their corporate information systems securely from any computer, anywhere in the world.

The Instant Collaboration Pack is another Portal Server add-on that enables secure real-time instant messaging and collaboration for workgroups within the enterprise boundaries.

More From Sun ONE

Also part of Sun ONE are several other servers, each playing a supporting role in delivery of Web services:

- Sun ONE Certificate Server—provides strong enterprise-wide security for authentication of employees, customers, and partners; issues, renews, suspends, revokes, and manages X.509-based digital certificates.
- Sun ONE Web Proxy Server provides traffic management services by collecting data from the network, determining where the data should be routed, and distributing it appropriately.
- Sun ONE Web Server,build multi-platform, e- commerce Web applications; works with Java Servlets and JavaServer Pages to generate personalized content.
- Sun ONE Calendar Server—manage calendars, share resources, and schedule events or appointments collaboratively.
- Sun ONE Messaging Server – enterprise-class standards based e-mail server, including a Web GUI for client access, and seamless integration with the Sun ONE Portal Server.

Other products complement the functionality offered by the many servers within the Sun ONE family. The Sun ONE Trustbase Transaction Manager delivers trust services in a standardized format, for secure communications between financial institutions and their customers. Because the services are standards based, the companies involved do not need to have an existing trading relationship—they can simply leverage the services provided by the Sun ONE Trustbase Transaction Manager.

The products of the Sun ONE platform are designed to come together and deliver both an integrated and integratable solution. Sun ONE products in the platform are built to offer speed and efficiency when used together. For example, by using the Sun ONE Directory Server together with the Sun ONE Portal Server, the speed of user authentication and personalization can be increased because of the strengths of the two products designed to work together.

Integratable refers to the fact that the Sun ONE Platform is open standards-based, providing interoperability with existing and future systems. For example, if an enterprise already has a non-Sun application server, it can still incrementally add Sun ONE products to its systems. Furthermore, an enterprise could standardize on the Sun ONE Platform today, but have the peace of mind that down the road if they need to replace pieces of it with a better offering from another vendor, they will be able do so with ease.

Solaris Operating Environment

The Solaris 9 Operating Environment is the operating system for the Web services paradigm, combining traditional OS functionality, application services, and identity management. The Solaris 9 OE is designed to provide the security, manageability, and performance that IT professionals need to increase service levels and decrease costs and risks. It is the foundation upon which the Sun ONE architecture is built.

Professional Services

An experienced professional services team is important in the technology world, especially when it comes to Web services. These teams have been on the front lines, trying to get applications to work together and processes to function correctly—they know what Web services need to deliver. Sun offers professional services that span several capabilities.

- Architectural Planning—define technical and application platform requirements for an IT architecture to help meet requirements.
- Web services and Biz Apps Integration—best practices for designing and implementing optimized technical infrastructures to support business applications based on Sun ONE technology.
- Platform Install and Integration—installation planning, project management and implementation for Sun servers and storage platforms including integration, migration, and consolidation services.
- Security—Help develop and implement security strategies for protection of systems, data and intellectual property.
- Production Support—Provide proactive systems and software support to help maintain and improve service levels required to fulfill needs in the enterprise and enhance business continuity.
- Customer Enabled Support—Help deliver cost effective self-service options to enterprise data centers and end users.
- Systems Operations Optimization—Provide data collection, data analysis, reporting tools and services based on innovative knowledge management technologies to help organizations improve the way they use IT.

Standards Support

Sun is actively involved in the W3C where it has contributed to the development of key specifications such as SOAP and XML and chairs many of the working groups including the influential Web Services Architecture Group. In addition Sun has been a visible participant in many of the emerging standards surrounding Web services, including ebXML, SAML, and UBL.

Sun has also championed the Liberty Alliance Project, a multi-industry effort to create a single standard for Network Identity, Authentication, and Authorization on the Web. Sun continues to be an important contributor along with more than forty other giants of industry who collectively represent more than a billion identities.

A comprehensive view of Sun's support for standards in its products is illustrated by the matrix at the bottom of this page :

Case Study: TransCanada Pipeline, Ltd.

Utility leverages Web services to lower total cost of IT ownership and automate currency conversion

TransCanada is a leading North American energy company. It is focused on natural gas transmission and power services. The company's network of approximately 38,000 kilometres of pipeline transports the majority of western Canada's natural gas production to the fastest growing markets in Canada and the United States. TransCanada owns, controls or is constructing a total of approximately 2,250 megawatts of power an amount of power that can meet the needs of more than two million average house-holds. The company's common shares trade under the symbol TRP on the Toronto and New York stock exchanges.

In the last few years, TransCanada has embraced a more streamlined, cost effective way of providing service to its customers and to doing business. This "Better, Faster, Cheaper" approach has been embraced by the company's IT department. This enables them to integrate where it adds value and allows them to respond quickly to changes in specific business areas, or even replace entire systems with reduced impact and dependency on other systems.

In addressing the business requirements, the IT department embarked on a pilot project to build the next generation of applications, driven by the goal leverage existing application resources wherever possible. From a development perspective, this goal as defined by the need for a common set of service components, able to be re-used in future application development, available on-demand as needed by TransCanada developers.

Consistent with a Web services view of software development, TransCanada's ultimate goal was to publish specific applications as callable services, allowing developers to focus on capabilities, not the specific underlying code. Another driving requirement was the desire to focus on standard-based development and avoid "vendor lock-in" or being limited to a single, proprietary technology. This notion lead to the search for a solution based on Java and XML.

Business Requirements

Key business requirements, jointly formulated by Sun and TransCanada, have driven the SPINE architecture. They are:

- Technology expense reduction: Decrease TransCanada operating, maintenance and upgrade costs for applications and software infrastructure, as measured by the total cost of ownership (TCO).

Functionality	Standards Body	Java APIs	Product
Core XML	XML, XSLT, XQuery	JAXP, JAXB, JAXQ, JSR, JAXD, JSR	Sun ONE Studio, Sun ONE Application Server, Sun ONE Integration Server, Java Web services Developer Pack
Messaging	SOAP, ebXML MS	JAX RPC, JAXM, SAAJ, JMS, J2ME	Sun ONE Studio, Sun ONE Application Server, Sun ONE Integration Server, Java Web services Developer Pack, Sun ONE Message Queue
Description	WSDL, ebXML, CPP/CPA	JAX RPC, J2ME WS	Sun ONE Studio, Sun ONE Application Server, Sun ONE Integration Server, Java Web services Developer Pack
Registration & Discovery	UDDI	JAXR	Sun ONE Studio, Sun ONE Application Server, Sun ONE Integration Server, Java Web services Developer Pack
Identity	SAML, Liberty	TBA	Java Identity Infrastructure, Sun ONE Identity Server
Security	SAML, XKMS	TBA	Java Signing Infrastructure, Sun ONE Identity Server, Sun ONE Certificate Manager

- Increased business responsiveness: Increase responsiveness to business change requests within TransCanada IT projects, as measured by reduced time to market (TTM) for new business services.

Functional Requirements

For the prototype phase, a simple service was developed. This service is a currency converter, intended to support a trading desk. While straightforward in concept, the prototype is a comprehensive effort, covering: J2EE application creation and deployment, creation, deployment, UDDI registration and lookup, consumption, authentication, authorization, metering, performance metrics capture and reporting, and sequencing

Directory Services

Directory services validate user authentication and authorization credentials. Requests from service components are received in XML format, and checked to ensure that permissions and policies allow access to specific services. iPlanet Directory Server, Access Management Edition was used to provide directory services as well as the management and enforcement of authentication and access policies. This software package offers policy-based authorization, in conjunction with role-based authentication, can be used with portal servers to create segmented user communities. This enables developers to create environments where the access specific users have to applications and services would be based on their role within the organization. While the prototype demonstrates a single service, as the SPINE capabilities expand, role-based services will play an important part in the overall infrastructure.

Web Service Event Management

The implementation of event management for the SPINE Web service architecture is based on an open, standards-based implementation of a management infrastructure, and on the OpenView Operations for Solaris product as the enterprise-grade backend event management solution.

Conclusion

The SPINE project demonstrates that IT departments are embracing the concepts of Services on Demand - the ability to deliver and manage Web services. Sun's architecture for delivering Services on Demand is Sun ONE. Sun ONE's appeal to TransCanada is strong Web services are being built with open technologies on an integratable and manageable framework that can embrace and extend existing IT assets. TransCanada is using the Sun ONE architecture to move ahead with their vision of better, faster, cheaper. With market-leading tools for creating, assembling, integrating, deploying, and testing, Sun offers a comprehensive environment for the Sun ONE architecture. Sun's commitment to open APIs is demonstrated with this project: integrators, third party products, and industry standard technologies all contributed to its success. TransCanada is moving to achieve its goals of an IT environment based on non-proprietary products and design. One that can be almost seamlessly improved, maintained, managed, and updated faster and more easily than before.

Summary

Sun's strategy is defined by the concept of "services on demand" and is productized in the form of the Sun ONE platform. The Sun ONE platform is Sun's framework for leveraging existing corporate IT systems as well as technology advances that have occurred over the past few years through a system of orchestrated Web services. The three main product lines that make up the Sun ONE platform are:

- Sun ONE Studio 4
- Sun ONE Infrastructure Software
- Solaris Operating Environment

The Sun ONE platform is designed to offer a unified view of community, transactions, infrastructure, clear and consistent tracking of IT progress and issues, reuse of components, applications, and services, and a wide array of user access options for Web services.

Vendor Contact Information

BEA Systems, Inc.
2315 North First Street
San Jose, CA 95131

(800) 817-4BEA

www.bea.com

divine
1301 N. Elston Avenue
Chicago, IL 60622

(773) 394-6600

www.divine.com

Fuego
15301 Spectrum Drive,
Suite 175
Addison, TX 75001

(972) 490-6700

www.fuego.com

Fujitsu Software Corporation
3055 Orchard Drive
San Jose, CA 95134

(408) 432-1300

www.fsw.fujitsu.com/interstage

HNC Software
5935 Cornerstone Ct. West
San Diego, CA 92121

(858) 799-8000

www.hnc.com

Vendor Contact Information

IBM Corporation
1133 Westchester Avenue
White Plains, NY 10604

www.ibm.com

Microsoft Corporation
One Microsoft Way
Redmond, WA 98052-6399

(425) 703-3979

www.microsoft.com

Nobilis Software
Russia Wharf 6th Floor
286 Congress Street
Boston, MA 02210

(617) 556-8288

www.filenet.com

Sun Microsystems, Inc.
4150 Network Circle
Santa Clara, CA 95054

(800) 555-9SUN

www.sun.com

About This Document:

The product-specific information contained in this document is intended to provide an overview of a specific product and vendor at the date of publishing. Facts presented have been verified to the best of our ability with the vendor and actual users of the product where indicated, however, Delphi cannot insure the accuracy of this information since products, vendors, and market conditions change rapidly. Delphi Group makes no implied or explicit warranties, endorsements, or recommendations in this report nor should such warranties be inferred from its contents. A complete assessment of your specific application, the method of implementation for a given product or technology, and the current state of that product must be considered in order for a recommendation to be made on any product's suitability for your purpose, needs and requirements.

Delphi Group is a leading provider of business and technology advisory services to Global 2000 organizations. With offices established around the world, Delphi has assisted professionals across disciplines and industries at nearly every major national and global organization and branch of government. Its clients and subscribers include more than half of the Global 2000.

DELPHI® GROUP

A Delphi Group White Paper

Ten Post Office Square
Boston, MA 02109-4603
v (617) 247.1511
f (617) 247.4957
www.delphigroup.com