Web services architect, Part 1: An introduction to dynamic e-business

Dan Gisolfi (gisolfi@us.ibm.com)
Solutions Architect, IBM jStart Emerging Technologies
April 2001

Emerging technologies have played a strong role in the evolution of the Internet over the past five years. Java gave us portable code; portable data came with XML; and Pervasive Computing addressed the connectivity of any device. Now the hype surrounds Web services. In this series of articles, I will discuss the importance of this technology in developing the next generation of the Internet as well as describe the Web services strategy of IBM. Additionally, I will explore the business impact of Web services, how to identify a relevant solution opportunity, and how to evaluate the many vendor strategies building around this technology.

Contents:

Related dW content:

Reflections on SOAP
The Tao of e-business services

This is the beginning of a new column directed at the CIO, CTO, or the software architect of a company. They are the ones most impacted by the technical and technological implementation issues surrounding large-scale deployment of Web services in an enterprise computing environment. My intent, though, is to go beyond simply discussing implementing a Web application server or creating XML-wrappers around your existing applications. I will touch on topics that should concern any adopter of emerging technologies and specifically focus on the hot issues that face early adopters of Web services technologies.

In this, the first installment of the column, I will lay out the basic terminology of Web services and describe the position of these technologies with respect to industry trends in distributed computing. You may not agree with all the definitions I provide, but this will give us a common vernacular going forward.

Industry has spent the last five years trying to define the role the Internet would play in the global economy. During that time, we experienced the early growing phases of the importance effective Web content and the need for e-commerce. Through it all, the concept of *e-business* -- once a brand name closely associated with IBM -- has come to simply mean *business*. That's right, just real business; business aided by a very powerful tool known as the Internet.

Now it is time for the next phase of e-business; the phase where dealing with business transactions and relationships needs to be addressed. In the first phase, we dealt mainly with the interactions on the front-end, where the actor was a customer. In some cases, that customer was a consumer; in other cases, a business; but in all cases, we were dealing with human beings. In the third phase of e-business, we need to deal with the interactions on the back end of the supply chain. Transactions that connect an enterprise to markets and to industries. This includes vital internal transactions: order processing, fulfillment, logistics, manufacturing, and employee processes. And these back-end interactions will, for the most part, be between computer systems, business applications, and software components. Enter Web services.

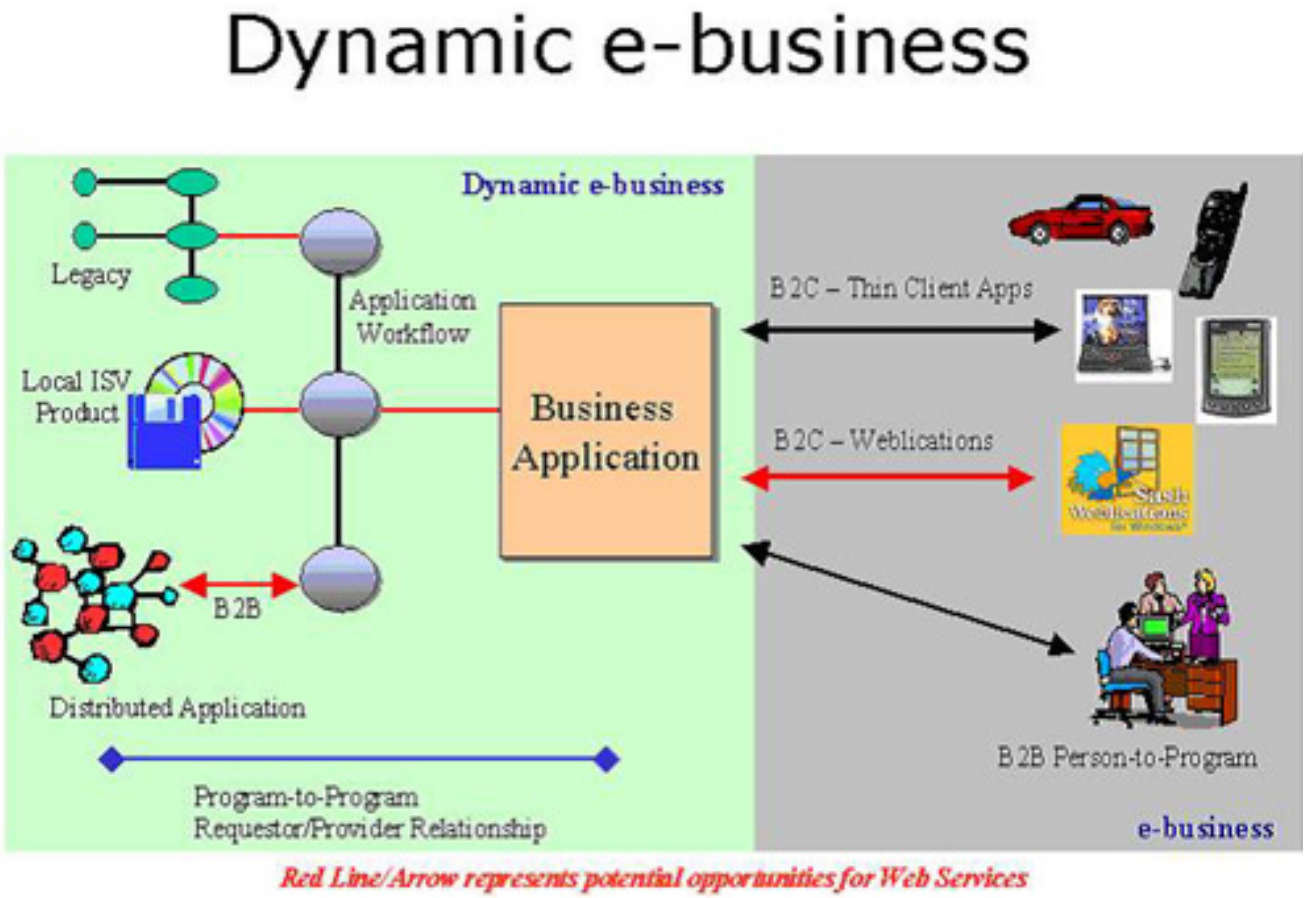
A vision

To truly understand the value of a technology and to turn that value into a business opportunity, it is important to understand the big picture. In the case of Web services from IBM, that big picture is referred to as *dynamic e-business*. A simple definition of dynamic e-business would be,

> The next generation of e-business focusing on the integration and infrastructure complexities of B2B by leveraging the benefits of Internet standards and common infrastructure to produce optimal efficiencies for intra- and inter-enterprise computing.

Essentially, dynamic e-business envisions an Internet where business entities can manage electronic interactions within their own domain and between trading partners programmatically. From the discovery of new partners to the integration of another business entity, dynamic e-business puts the emphasis on program-to-program interactions instead of the customer-to-program interactions that dominated in the early B2C phases of e-business (see Figure 1).

Figure 1: A view of dynamic e-business



In the recent past, we have focused on providing customers with solutions that help manage the proliferation of data between the end-user and a business application over any network to any device. From the concepts of screen-scraping and Web clipping to the importance of transcoding content to any device to support the benefits of a thin-client architecture, the emphasis is on front-end interactions.

The logical extension to the investments our customers have made in thin-client architectures is investing in the necessary infrastructure for the integration of software components that process back-end tasks of e-commerce applications. But what is unique about this next generation of Internet computing?

Principles of dynamic e-business

Twelve to eighteen months from now, as the vision of dynamic e-business is ingrained in e-business, a number of basic principles will surface to help articulate what must be done to address the complexities of B2B integration. For now, I can offer some insight to those principles:

1. Integration between software resources should be loosely coupled.
2. Service interfaces for software resources should be universally published and accessible.
3. Program-to-program messaging must be compliant with open Internet standards.
4. Applications can be constructed by stitching together core business processes with outsourced software components/resources.
5. An increase in the availability of granular software resources should improve the flexibility and personalization of business processes.
6. Reusable outsourced software resources should provide cost and/or productivity efficiencies to service consumers.
7. Software can be sold as a service.

For dynamic e-business to become a reality, there must be common architectures and open Internet standards to support it.

An architecture for dynamic e-business

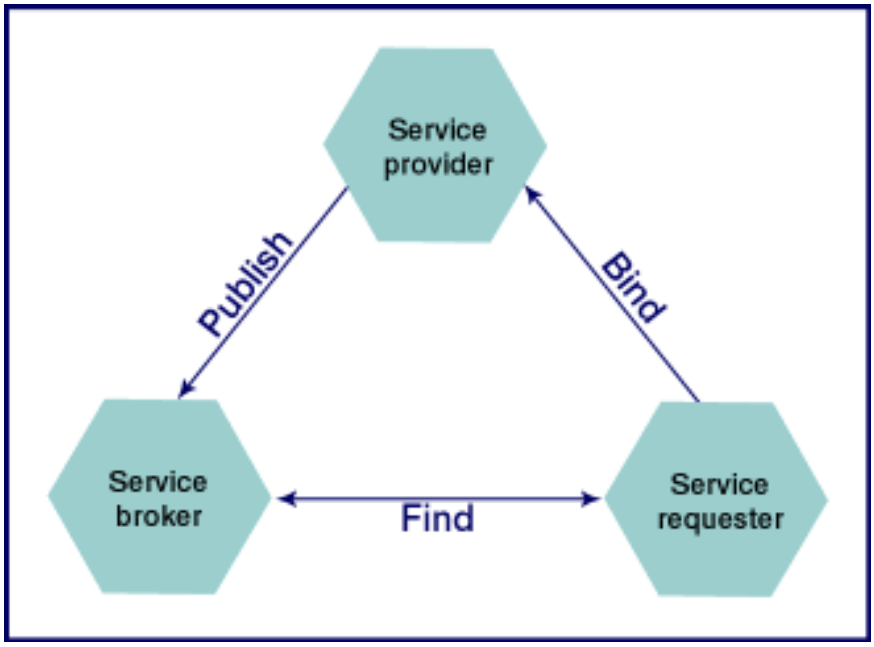
Enter the *Service-Oriented Architecture* (SOA [see Resources]). SOA is a conceptual architecture for implementing dynamic e-business. Today, most of the systems and applications running in the business world are made up of tightly-coupled applications and subsystems. The drawback with this is that a change to any one subsystem can causes breakage in a variety of dependent applications. This brittle aspect of existing systems is in part responsible for the high cost of system maintenance and the limitations around the number of trading partners one can manage.

SOA is not a new concept. In fact, about a year and a half ago, HP's e-speak appeared with a marketing campaign built around a proprietary implementation of SOA. Due in part to its proprietary requirements, e-speak failed to make much of a market impact.

As recently as February 2001, HP revamped their software strategy to embrace the coupling of distributed components via SOAP, but they remain partially proprietary on the service interface definition language (IDL) of their solution. Nevertheless, the potential concept of SOA was found to have merit by companies like IBM and Microsoft who recognized that for SOA to succeed where other distributed computing concepts had failed, it must be implemented on open standards. Thus, the recent cooperation between these companies on recommended standards like UDDI and WSDL. More on that later!

Regardless of the implementation, SOA is comprised of three participants and three fundamental operations (see Figure 2).

Figure 2: The SOA model



A *service provider* is a Network node that provides a service interface for a software asset that manages a specific set of tasks. A service provider node can represent the services of a business entity or it can simply represent the service interface for a reusable subsystem.

A *service requestor* is a Network node that discovers and invokes other software services to provide a business solution. Service requestor nodes will often represent a business application component that performs remote procedure calls to a distributed object, the service provider. In some cases, the provider node may reside locally within an intranet or in other cases it could reside remotely over the Internet. The conceptual nature of SOA leaves the networking, transport protocol, and security details to the specific implementation.

The third SOA participant is that of the *service broker*; it is a Network node that acts as a repository, yellow pages, or clearing house for software interfaces that are published by service providers. A business entity or an independent operator can represent a service broker.

These three SOA participants interact using three basic operations: *publish*, *find*, and *bind*. Service providers *publish* services to a service broker. Service requesters *find* required services using a service broker and *bind* to them.

Enabling technologies

Once you comprehend the concept of SOA, reflect on some of the basic principals of dynamic e-business to understand how best to implement it. A fundamental aspect of a successful implementation is the reliance on open Internet standards. The dynamic e-business strategy is founded on a core set of emerging technologies that reflect the work of researchers and consultants from a variety of companies and industry organizations.

So what technologies make up the existing set of enabling technologies? Let's take a peek:

- XML: The Extensible Markup Language 1.0 standard is a text-based markup language specification from the World Wide Web Consortium (W3C). Unlike HTML, which uses tags for describing presentation and data, XML is strictly for the definition of portable structured data. It can be used as a language for defining data descriptive languages, such as markup grammars or vocabularies and interchange formats and messaging protocols.
- SOAP: Simple Object Access Protocol is an XML-based lightweight protocol for the exchange of information in a decentralized, distributed environment. SOAP defines a messaging protocol between requestor and provider objects, such that the requesting objects can perform a remote method invocation on the providing objects in an object-oriented programming fashion. The SOAP specification was co-authored by Microsoft, IBM, Lotus, UserLand, and DevelopMentor. The specification subsequently spawned the creation of the W3C XML Protocol Workgroup, which is comprised of over 30 participating companies. SOAP forms the basis for distributed object communication in most vendor

implementations of SOA. Although SOA does not define a messaging protocol, SOAP has recently been referred to as the *Services-Oriented Architecture Protocol* due to its common use in SOA implementations. The beauty of SOAP is that it is completely vendor-neutral, allowing for independent implementations relative to platform, operating system, object model, and programming language. Additionally, transport and language bindings as well as data-encoding preferences are all implementation dependent.

- WSDL: The Web Services Description Language is an XML vocabulary that provides a standard way of describing service IDLs. WSDL is the resulting artifact of a convergence of activity between NASSL (IBM) and SDL (Microsoft). It provides a simple way for service providers to describe the format of requests and response messages for remote method invocations (RMI). WSDL addresses this topic of service IDLs independent of the underlying protocol and encoding requirements. In general, WSDL provides an abstract language for defining the published operations of a service with their respective parameters and data types. The language also addresses the definition of the location and binding details of the service.
- UDDI: The Universal Description, Discovery, and Integration specification provides a common set of SOAP APIs that enable the implementation of a service broker. The UDDI specification was outlined by IBM, Microsoft, and Ariba to help facilitate the creation, description, discovery, and integration of Web based services. The motivation behind UDDI.org, a partnership and cooperation between more than 70 industry and business leaders, is to define a standard for B2B interoperability.

(see Resources for more information on all of the material above.)

These enabling technologies collectively reflect on the set of Web services technologies offered by IBM. Over time extensions and additions to this set will arise but all such changes will spawn from an ongoing reliance and cooperation with open industry efforts.

Web services emerge

I have listed the enabling technologies needed to implement a service-oriented architecture. I point to UDDI as a standard way for addressing the need for a repository or broker that will manage a directory of service interfaces. I mentioned the concept of a service IDL and the role of WSDL. But the overall intent of UDDI can only be achieved if a critical mass of service providers is created. They must deploy software resources for consumption over the Internet. Each software resource, known as a *Web service*, is a granular software component that can be used as a building block for distributed applications or for the assembly of business processes. A Web service can accept requests to perform a specific set of tasks and respond to such a request using open messaging standards to insure interoperability. Additionally, a Web service may itself be an aggregate of Web services.

Summary

My purpose with this article was to provide a cohesive overview of the technical landscape, commonly referred to in the industry as Web services, and to add some clarity to dynamic e-business.

We started at the thirty thousand foot level with a vision of dynamic e-business. We then descended down through a common architecture (SOA) and proceeded by outlining a set of open enabling technologies. Finally, we landed on the notion of a reusable and network-accessible software asset, a Web service.

Armed with the necessary tooling and products to create, deploy, and host Web services, businesses can attack the integration and infrastructure complexities of B2B and achieve the vision of dynamic e-business.

Resources

- Read the Web services Architecture Overview.
- Checkout real world adoption scenarios for dynamic e-business.
- Review the Extensible Markup Language.
- Learn about the Simple Object Access Protocol.
- Read about the Web Services Description Language.

- Learn more about Universal Description, Discovery, and Integration by visiting its homepage.
- Look to see who is part of the XML Protocol Workgroup.

About the author

A 13 year veteran of IBM, Dan Gisolfi holds a Masters Degree in Artificial Intelligence from Polytechnic University, and a BA in Computer Science from Manhanttanville College. Prior to 1999, his career was focused on software and product development ranging from Expert Systems, OS/2, and Secure Internet Payment Systems. As a member of the jStart (jump-Start) Emerging Technologies Team, he keeps his hands dirty in both the business and technical aspects of customer engagements. From Business Development Manger and Evangelist to Solution Architect and Contract Negotiator he gets to wear many hats. As jStart Team Lead for Web services, he is helping IBM drive the adoption of this emerging technology through real business solutions. You can reach him at gisolfi@us.ibm.com.