3.0 Web Services in Action: Case Studies From the Front Line

Enterprises that want to use IT to compete effectively should begin developing with Web services in 2003. Web services are a technology of the moment, but they are resilient and will thrive in the future. Understanding their impact on business will enable IT managers to experiment with them effectively. Separating promise from hype will enable business managers to exploit the capabilities of Web services for internal and external use.

Enterprises that are interested in the use of Web services can learn valuable lessons from the examples of early adopters. Such precedents can help enterprises to identify best practices or inspire them to come up with novel ideas for the use of Web services technology.

This chapter uses case studies to illustrate how enterprises can chart a strategy for introducing Web services in the enterprise. The analysis in this chapter is framed by the following Key Issues:

- What are Web services, and why do they matter?
- How are Web services being used today?
- What key factors should enterprises consider in their use of Web services?
- What key trends will shape enterprise use of Web services in coming years?

3.1 Understanding Web Services and Their Usefulness

Key Issue: What are Web services, and why do they matter?

Common-sense definitions of technology allow for some wiggle room in vendor claims while also creating a common frame of reference that all participants in a market may adopt. Gartner has heard many competing definitions of Web services. In one case, a major vendor representative referred to File Transfer Protocol as an early Web service, while stricter interpreters will only certify a facility as a Web service if it meets their own complex, alphabet-soup definition.

Gartner believes that the following definition is a useful and reasonable one:

- Web services are software components that employ one or more of three technologies — Simple Object Access Protocol (SOAP), Web Services Description Language (WSDL) and Universal Description, Discovery and Integration (UDDI) — to perform distributed computing. Use of any of these basic technologies constitutes Web services; use of all of them is not required.

Although it is somewhat oversimplified, this common-sense definition, like others before it, makes a trade-off between oversimplification and a wider and clearer understanding. This definition may change over time, but, for now, it suffices. (Note that the use of SOAP requires the use of XML.)

Why do Web services matter? They promise substantial improvement in the dynamic interaction among networked applications through the use of standards.

Action Items: Don't get hung up on definitions. Remember that Web services' usefulness lie in their simplicity and utility.

3.2 The Use of Web Services Today

Key Issue: How are Web services being used today?

3.2.1 Current Deployments

Strategic Planning Assumption: *Initial stages of Web services adoption will have neutral or negative impact on development efficiency and business innovation as enterprises grow accustomed to the new model through 2003; subsequent stages will result in substantial improvements in business innovation and development efficiency as enterprises gain proficiency in Web services through 2006 (0.7 probability).*

Web services are being deployed now to manage lightweight data integration and to provide improved data access across enterprise information webs.

Enterprises should expect the benefits of Web services to come slowly, but steadily, as they maneuver from learning about the specifications to dreaming up new uses for them. Any technology will follow a similar path, along which technologists and business managers scrutinize hype cycles and jump on board at the right time for their own project cultures.

Challenges to the progress of Web services adoption include:

- The inevitable and worsening vendor conflicts
- The current economic slowdown, which pinches budgets and draws all gazes to the bottom line
- General confusion over what Web services really are

In general, business and IS staffs should expect the first stage of an enterprise's consideration of Web services to produce few benefits in terms of reduced project execution time or business value, as both business and IS resources are consumed by learning the potential of the new business. As IS staff devotes more time than usual to the new development paradigm, business managers will benefit through improved data access and processes. Business innovation will stall as IS teams turn to development efficiency, but when an enterprise fully exploits a service-oriented architecture through Web services it will experience a renewal of business creativity.

Action Items: Set the initial education phase outside expectations for return on IT investment (ROI). In measuring ROI, consider both short- and long-term returns.

3.2.2 Case Studies

Enterprises that are interested in the use of Web services to accomplish the goals of particular projects, or that are considering Web services as the fundamental aspect of conversions to service-oriented architectures, should learn from the examples of early adopters. Such precedents can help enterprises to identify best practices or inspire them to come up with novel applications for Web services technology.

To demonstrate the increasing diversity of the technology's impact, Gartner recently sought out enterprises that have chosen to use Web services in projects. Case studies of seven such enterprises are provided in the following sections.

3.2.2.1 Case Study: Rotech Healthcare

Strategic Planning Assumption: *Web services will catalyze a 50 percent increase in data integration for analysis among Type A enterprises through 2004, and among Type B enterprises from 2004 through 2006 (0.7 probability).*

Tactical Guideline: *As always with projects that rely heavily on open-source or community-based development methods, standards remain immature, and projects should not be "shoe-horned" into a particular model. Although Rotech has 10 Web services projects in development or testing, it is not seeking to solve all business problems through a Web services model.*

Rotech Healthcare, based in Orlando, Florida, provides medical services through a distributed network of small healthcare companies in 650 locations. It maintains a lean IT staff and seeks to solve business problems through lightweight integration initiatives.

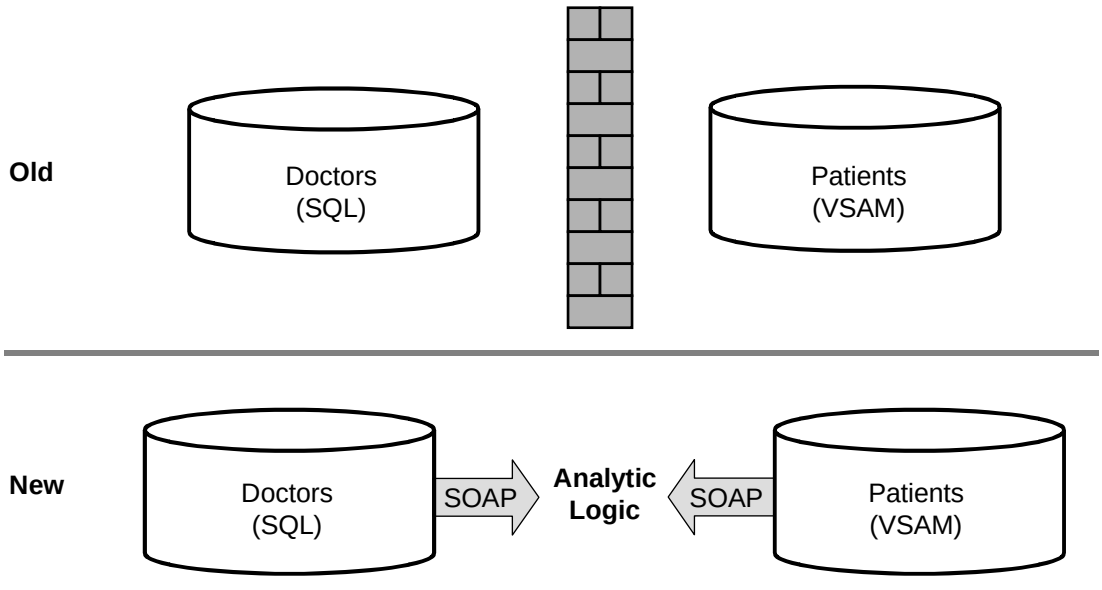
To better manage its sales resources, Rotech wished to compare data about the location of enlisted doctors' offices with the location of its patients. Incongruities or anomalies revealing that more doctors than necessary had been sold in low-patient areas, or the reverse, would aid Rotech in deploying sales representatives to even out the mixture. However, data concerning the patients Rotech serves resides in a Virtual Storage Access Method file management system, while data concerning relevant doctors resides in a Linux-based Structured Query Language database.

Instead of creating a new, relational database, Rotech established a SOAP link between the data sources to feed a homegrown analytical application (see Figure 3-1). The use of Web services enabled Rotech to establish the link more swiftly than it could have through conventional means. Rotech has spent very little on the project, devoting one or two developers per day for less than two weeks.

Aware that SOAP faces challenges in terms of its ability to transfer large amounts of data rapidly, Rotech runs the reports at night and periodically, instead of in real time.

Rotech believes the results have provided immediate benefits. Rotech can see more clearly how its sales resources should be deployed to improve the utilization

Figure 3-1: Rotech Uses SOAP to Bridge Disparate Data Sources



SOAP Simple Object Access Protocol
SQL Structured Query Language
VSAM Virtual Storage Access Method

Source: Gartner

of its core business services, including circumstances where its patient count implies that its doctor count is too low.

Action Item: Consider light data integration with an analytical engine as an early SOAP project.

3.2.2.2 Case Study: The Colorado Department of Agriculture

Colorado's agriculture department needed to modernize access to the multiple sources of data it uses in tracking the health of the state's captive elk population. Before the department's Web services project, employees had to manually combine data from multiple sources into a Macintosh HyperCard application for analyses. Defining, assembling and inspecting reports could take as long as six weeks.

When an elk is diagnosed with the disease, it is critical to determine what other elk it has had contact with, to trace back to any outbreak's "patient zero," as well as to evaluate the risk of contagion in other elk. Thus, the six-week wait was a significant detriment to rapid response. Additionally, the application within which the data was analyzed did not reside on the network and, therefore, was difficult to access.

The dominant technologic source for the data was a database that generated printed reports that employees in turn typed into the HyperCard application, where both the business logic and the data resided. The relational database consists of about 20 tables, with 14 to 30 fields per table. Lengthy waits to trace sick elks' previous contacts were common.

In the Web services solution, users place their requests via an Active Server Pages (ASP) Web interface, which passes requests via a dynamic link library to SOAP wrappers (see Figure 3-2). These, in turn, address a Component Object Model Plus (COM+) transaction layer, which interacts with the database. The results are passed back through the chain to the ASP presentation layer for the user to see. The SOAP layer for data transport was adopted to increase the installation's future flexibility.

Officials report that they can trace back an elk's contacts with other elk immediately, instead of waiting weeks for a report. The universal view of data the new system provides is critical for enabling unified strategies of public communication about outbreaks or diseased individual elk.

Development required one worker from Compuware and two workers from the state for 14 weeks. The project architect was full time. Two developers were cycled in and out of the project.

Developers learned that the Web services aspect must be tied closely to the data source. An initial attempt at a three-tier physical architecture failed, because interchange between the Web service and data source bogged down the Web service's response time. Placing the service and the database on a single physical server and placing the Web server on a separate server resolved the problem. The state learned that, although substantial benefits are to be gained from a Web services deployment, the technology's newness led to development "hiccups" that delayed the project somewhat.

Action Item: Through 2004, consider deploying Web services in visible but manageable projects that require no more than eight developers.

3.2.2.3 Case Study: MapPoint

Tactical Guideline: *Microsoft's experience with MapPoint .NET illustrates how effective delivery of Web services requires at least as much effort in operational deployment and commercial strategy as it does in technical specification and development. Enterprises should ensure that they have identified relevant stakeholders and process owners for all dimensions of a Web service deployment.*

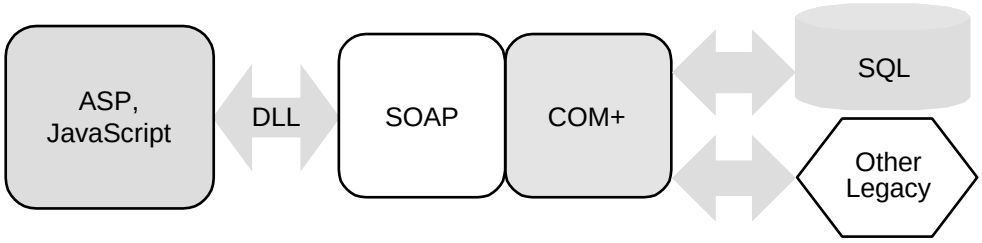
MapPoint shows the potential and the challenges of fully deployed Web services. MapPoint is a natural candidate for Web services use because mapping data has high value in many contexts, but isn't something most enterprises want to collect or manage for themselves. Map data appears static but in practice changes significantly all the time, particularly when complemented by data such as points of interest. MapPoint thus shows the potential for other owners of specialized data to distribute information at a transaction level via a Web service.

MapPoint comprises a substantial database of cartographic data and a program to search and display this data. The MapPoint application was conceived and

Figure 3-2: Colorado Agriculture Department Speeds Elk Health Analysis

Captive Elk Population Health Tracking System

- Reduced tracking time
- Improved interoperation
- Replaced legacy system



ASP	Active Server Pages
COM+	Component Object Model Plus
DLL	dynamic link library
SOAP	Simple Object Access Protocol
SQL	Structured Query Language

Source: Gartner

designed to run as a single-user PC application. It is also available as an ActiveX control with an extensive object interface, intended for embedding in other applications. The challenge for Microsoft was to turn MapPoint into a multiuser system, accessible over the Internet, and capable of integration with other systems requiring mapping data.

The initial step of defining a SOAP application programming interface (API) was fairly simple, since that could be wrapped onto the proven Component Object Model (COM) API from the ActiveX control. The second technical challenge was more substantial. It involved taking a query and rendering program designed for single users and turning it into something that could service high-volume concurrent access over the Internet. Microsoft had, however, already decided to deliver an Internet mapping service via MSN, and the implementation of a high-performance engine to access the cartographic database was, therefore, already planned.

With a SOAP API, a WSDL interface specification and a high-performance data access engine, the MapPoint Web service was "ready to fly." The more-substantial challenges involved creating a deployment infrastructure and commercial terms and conditions for use of the service.

A number of the initial deployments of MapPoint .NET are within MSN services (for example, Expedia and HomeAdvisor). Other early adopters are Dollar Rent-A-Car and the German hotel booking service hotel.de. TellMe uses MapPoint .NET to provide voice driving directions from a mobile phone.

A particularly innovative application of this Web service was developed by Marks & Spencer, which uses MapPoint .NET to combat credit card fraud. When the same account number shows up as having been used in multiple stores in a brief time window, this raises an alert. The system then sends the store locations to MapPoint .NET with the request to return a drive time. If the drive time is significantly more than the actual differential between purchases, the conclusion is that an instance of credit card fraud has likely occurred.

Several key factors contributed to the success of the MapPoint .NET initiative, including the following:

- Microsoft identified clear business objectives related to the intrinsic value of the established database.
- It created a deployment environment, including staging and managed operations, that allowed users to test and monitor systems that embed the Web service.

- Microsoft identified a commercial framework that reflects the natural usage pattern of applications using the Web service. This framework, which takes advantage of transaction-based pricing, is seen as equitable but not overly complex.
- MapPoint .NET leverages considerable prior investment. Creation of such a service from scratch would be much more difficult to justify.

Action Items: Enterprises should use an external service as a simple way to begin Web services training. Enterprises should press vendors to establish a SOAP consumption functionality.

3.2.2.4 Case Study: uBid

Strategic Planning Assumption: *Through year-end 2005, Web services will drive a 30 percent increase in interenterprise connections for data exchange and service exploitation in relationships where the dominant partner is a Type A enterprise (0.7 probability).*

Tactical Guideline: *Where interconnections between enterprises are critical to the provision of a service to a customer, Web services technologies offer a relatively standard, low-risk, low-cost, multiplatform approach. uBid's situation represented an optimal opportunity to use Web services, given the reasonableness of the interface, the limited number of partners, the nature of the business transaction and the contractual nature of the business relationship. Enterprises should look for opportunities that fit this profile as candidates for early interenterprise Web services projects.*

The complexity of the functionality available at most auction sites has increased significantly, and aggregators are becoming more important as their value and the value of online auctions are better understood. uBid needed a modern technological solution for supporting multiple aggregators. The most-important motivator was the threat that an aggregator might pull its support from a site if interacting with it was too hard.

uBid needed a solution that would enable aggregators to interact with its site in a reliable, secure and cost-efficient way. It wanted a solution that could be developed once, for all aggregators.

uBid discussed the situation with many of the aggregators and found that they all appeared ready to adopt SOAP interfaces to their systems. uBid's own Seller Interface, with its Microsoft COM and Distributed COM components, was also a likely candidate for conversion into a Web service. uBid and Tallán developed a synchronous XML- and SOAP-based interface enabling the flow of information between uBid's databases and third-party applications (see Figure 3-3).

Initially, the project team consisted of two people, who spent the first two weeks evaluating SOAP implementations and designing a solution. By the third week, the uBid project team had grown to 14 people, had a test site running and had produced a WSDL file for consumption by the aggregators. The project was completed in 14 weeks.

It now takes less than two weeks to connect a new aggregator, and the onus for the integration effort falls primarily on the aggregator. uBid simply supplies a WSDL file and helps test the integration.

uBid has seen the average number of items available for auction increase from approximately 500,000 to approximately three million. It is now connected to numerous aggregators and has seen the number of available items increase significantly, with a high of six-and-a-half-million items in one day.

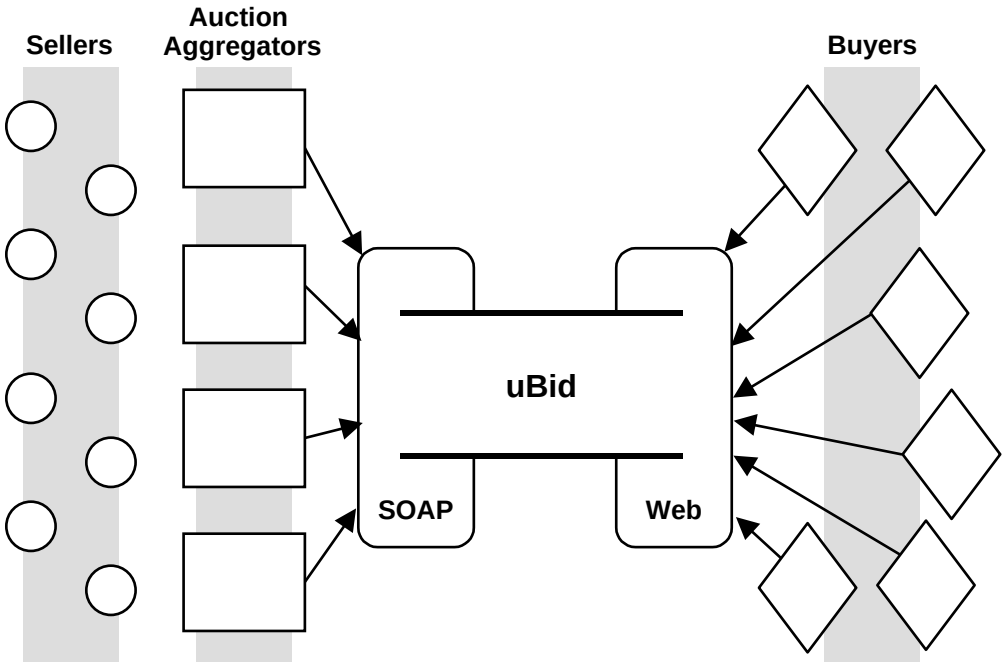
Action Item: Consider Web services as a way to allow tactical partners to integrate with a simple interenterprise data aggregation function. Where multiple future connections will be necessary, Web services can reduce investment in integration connectors.

3.2.2.5 Case Study: Eastman Chemical

Tactical Guideline: *Eastman has steered clear of trying to reinvent its online ordering system, and has also resisted the temptation to re-architect its business-to-business e-commerce system. Eastman has chosen instead to pursue a straightforward, comparatively low-risk Web services application, which could continue to function in a traditional Web server environment should problems arise.*

A global company with headquarters in Kingsport, Tennessee, Eastman Chemical has been very pragmatic

Figure 3-3: uBid Connects With Multiple Aggregators



Source: Gartner

SOAP Simple Object Access Protocol

in evaluating the potential of Web services to streamline Web-based e-commerce. After a careful analysis, Eastman concluded that shortening the "info to order" cycle was a realistic application of Web services at this early stage of its evolution.

This would mean putting deep, responsive product information systems in place to simplify and ultimately expedite the online product selection process. Typically, such systems assist the customer with appropriate and effective use of purchased products, using such tools as product configurators.

Eastman had migrated its configuration spreadsheets online as interactive product configuration wizards. Since these wizards were driven by server-side application logic and dynamically generated scripts, they were cumbersome to maintain and enhance. More important, given the breadth of Eastman's reseller network, this script-driven architecture made it difficult to provide customizable, value-added wizards that could be hosted by the company's specialized channel partners.

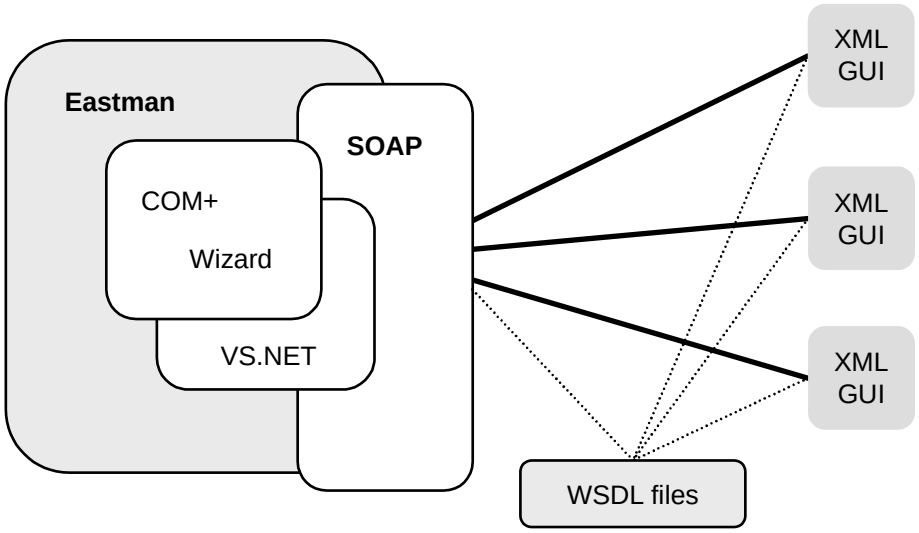
Therefore, Eastman chose a Web services approach. It has developed SOAP/XML wrappers for a number of wizards. Initially, the wizards were deployed directly to the desktops of its largest direct customers by using general-purpose XML-based, client-side graphical user interface (GUI) engines (see Figure 3-4). These engines can execute within a Web browser or directly on the desktop, and are designed to be powered on the back end by a presentation layer server operated by Eastman. The company used a combination of Microsoft's COM+ components and Visual Studio .NET development framework to create the wizards.

On the back end, Eastman uses WebMethod's WSDL generator to publish its wizard interface descriptions. These descriptions are manually uploaded to Grand Central Networks' service registry.

Initial testing show one- to two-second response times for the wizard, which falls well within the acceptable range for an interactive desktop application. Eastman is confident that this performance will scale satisfactorily, as most wizards should have less than 100 concurrent users at any given time.

Eastman has found that the distributed, rich-client GUI architecture enables it to build and deploy user interfaces that are far more sophisticated than those generally available through Web browsers. This approach has

Figure 3-4: Eastman Chemical Streamlines Customer Ordering



COM	Component Object Model
GUI	graphical user interface
SOAP	Simple Object Access Protocol
WSDL	Web Services Description Language
XML	Extensible Markup Language

Source: Gartner

allowed the company to create a user experience that enhances the intrinsic value of the company's product knowledge base.

Key to the success of this project was the choice of a highly distributed, XML-based GUI environment, and a relatively uncomplicated, Microsoft-based execution flow on the back end of the wizards. Thus, Eastman has wisely chosen to "keep it simple" from a process flow and execution standpoint.

Action Item: Enterprises should employ Web services as a relatively low-risk technology that can be used to implement a high-leverage business strategy.

3.2.2.6 Case Study: Google

Strategic Planning Assumption: *By year-end 2004, more than 5 percent of Google searches will be invoked through its SOAP API (0.7 probability).*

Tactical Guideline: *Applications accessible through the Internet may be swiftly divided into elements and exploited effectively with no major undertaking, if the application is useful, effective, trustworthy — and priced right.*

The popular search engine Google has published a SOAP interface. The uses made of this interface by a community of enthusiasts herald the new rules of the Web services era.

In early 2002, Google announced that it would provide access to its search index and particular functions through an interface using SOAP. The ensuing months brought a flurry of intriguing exploitations of the opportunity. Google has tallied a variety of user integrations to its index and search functions, from simple "because it's there" inclusion of a Google search box on Web journal pages to more-sophisticated use of its matching and relevancy algorithms.

Developers have created:

- A system that enables users to input the contents of their Web journals into a Google relevancy matcher, along with other homegrown analytic functions, to reveal other journal keepers who share their interests.
- A typographical-error generator that enables users to input a word, then see how it is most-often incorrectly rendered in written (Web-accessible) discourse.

- A Web journal publishing engine that analyzes an entry's title and incorporates in the entry an addendum of appropriate searches as identified by Google.

The enthusiasm with which users have incorporated search functionality into these other applications demonstrates how search is increasingly capable of being reduced to components. Like other applications, search engines were originally aspects of unitary products that included access, business logic and contents, as well as other support elements, such as repository format and user authentication. As those functions have been broken out and search applications have emerged into their own category, vendors have deconstructed these applications further.

The trend toward deconstruction is promoted and augmented by the growing interest in Web services as a method of accessing application interfaces in a granular, but standards-based, fashion. Google will use this step to provide its index to enterprises that require a look at the broad Web, or at their own internal content, but lack the sophistication to develop a full Web index or the commitment to establish search-and-retrieval logic for their own unstructured content.

Google is casting a wide net over a variety of business models, and access to this API is no exception. Gartner expects that, by year-end 2004, more than 5 percent of Google searches will be invoked through its SOAP API.

Action Item: Enterprises of all sizes should encourage developers to dream up SOAP-enabled interfaces for their most-valuable and least-accessible data repositories.

3.2.2.7 Case Study: Swiss Interbank Clearing

A few years ago, Swiss Interbank Clearing's (SIC's) electronic data interchange (EDI) implementation — based on EDI for Administration, Commerce and Transportation (EDIFACT) — was slow, complex, and difficult to maintain. EDI required expensive and redundant mechanisms to back up operational data transfers in an asynchronous communication mode.

Customer demands for faster fund clearance, along with a Web-based interactive interface to monitor transaction processing, could not be met by the established infrastructure. In addition, customer banks experienced problems late in the funds clearance process, potentially harming customer relationships, and resulting in high exception handling costs.

In 2001, SIC employed Otego AG, a Zurich-based consultancy, to develop and implement a Web-based system to replace its older, asynchronous EDI infrastructure. SIC decided to use a Web services approach, with the hope it would achieve lower development costs and reduced time to market if changes to the core application were handled on SIC's existing mainframe-based clearing system through an XML-based Web connector. The core application was based on scripting EDIFACT fund clearance processes to XML, and the use of SOAP messaging for connecting XML data to the Hypertext Transport Protocol (HTTP) transport layer (see Figure 3-5).

SIC clients can choose between different methods of access (via browser, Java client and end-to-end integration) on existing client devices, when necessary. Secure HTTP encryption with Secure Socket Layer is used to ensure the safe transfer of all data. Authentication is achieved by using smart cards operating digital certificates within an encrypted channel.

The new Web service solution differs from the old EDIFACT-based system in its synchronous data flow, use of Web-based specifications and extensible architecture. Response times have been reduced to a few seconds in a real-time environment. SIC customers receive immediate feedback to each action in the payment-clearing process, including potential payment-clearance problems.

Action Item: Traditional users of EDI in the financial sector should evaluate Web services projects to move operations to a real-time and open environment.

3.3 Web Services Implementation

Key Issue: What key factors should enterprises consider in their use of Web services?

3.3.1 Making the Web Services Decision

Remember the criteria that were used in early Internet projects and that were often settled on more by accident than by design. Don't select projects based on their ability

to generate big funding, for example. Major projects providing critical value to products in production should only be undertaken by "bleeding-edge" enterprises that are comfortable with risk, and that cannot find similar capabilities through non-Web-services systems. In general, any description of a project within an enterprise that is not an aggressive risk taker should end with a mention of Web services — not begin with the justification that it is a Web services project.

Enterprises should consider using Web services if:

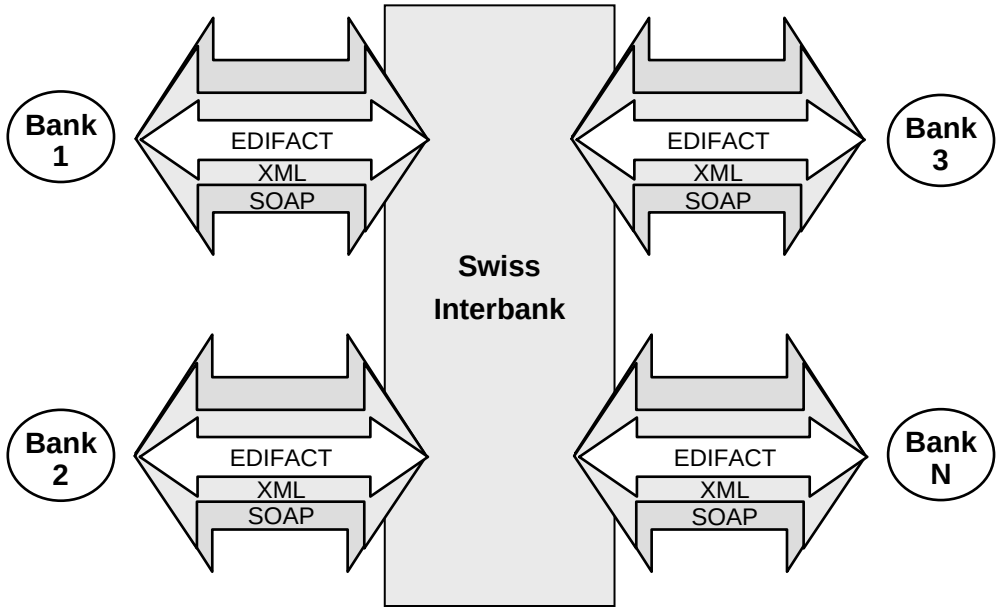
- What's moving is data, not goods.
- Many people and projects use the data.
- Dynamic relationships are the rule in your industry.
- The data is clean.
- The project has midrange visibility, and the benefits are eye-catching but tactical.
- IT staff expresses interest in Web services.

Enterprises should avoid using Web services if:

- The project is oriented toward operations.
- No one cares "what's in those numbers."
- People change suppliers as often as they buy new cars.
- The underlying data's value is suspect.
- It's a low-level sponsor's pet project.
- IT staff would prefer to "wait out a round or two."

Enterprises that comb their development project wish lists for nagging (but not strategic) problem areas will likely find suitable candidates for testing. Such problems would include those that aren't solvable through traditional development means or that need the unbroken attention of technologists. Examples include aggregation of data from obsolescent sources (such as proprietary or unpopular relational database management system formats) and the transfer by hand of the results of a

Figure 3-5: Swiss Interbank Clearing Takes Transactions to the Real-Time Level



EDIFACT Electronic Data Interchange for Administration, Commerce and Transportation
SOAP Simple Object Access Protocol
XML Extensible Markup Language

Source: Gartner

calculation from one piece of software or one networkable device to another.

Individual projects should require just a few weeks' time and a few developers' efforts, and, in most cases, will occur inconspicuously. Such projects may be coupled to achieve a larger goal, but the development methodology should not reach the level of overall project planning. Projects will probably be rolled out in a manner similar to early Internet technology projects — for example, departmental Web servers for intranet development.

Enterprises that approach the issue of Web services by looking for appropriate services outside the enterprise will be disappointed when they discover the paucity of such services and their relative immaturity. Despite the obvious benefits of reusing internal application elements and functions, some enterprises insist on seeing their major opportunity in Web services as one of locating external services (as was the case in the MapPoint example) and incorporating them into internal applications. Initially, such opportunities will be the exception, rather than the rule. For initial payback, enterprises should seek services in their own architectures.

Action Items:

- Carefully select projects using Web services. Begin by identifying a short list of internal development problems that can potentially be fixed through Web services technologies immediately.
- Populate small teams of developers with volunteers who will become well-seasoned to lead bigger and more-valuable Web services efforts later.

3.3.2 Addressing Key Challenges

Like other innovations, Web services mix the thrill of new capability with the disadvantages of complication. When first confronting Web services, most enterprises will find the technology daunting, because of its apparent (if not genuine) complexity, and the need to find like-minded partners with a technology development focus that matches that of the enterprise.

The biggest payoff from designing and implementing a Web services architecture will come in establishing controlled engagements among trading partners. The initial "baby steps" to reach that payoff, however, will likely begin within an enterprise's own walls. In September 2001, Gartner predicted: "More than 50 percent of enterprises' first experiences with Web services will be an internal deployment of a Web-services-enabled architecture (0.7 probability)." Gartner client feedback thus far appears to validate that prediction.

Enterprises that want to build service-oriented architectures must check internally, as well as troll the IT professional market, for two skill categories:

- Traditional developer skills, such as programmers in Visual Basic and COBOL, and new Java and C# developers
- New specialist skills that are needed to transform software architecture into service-oriented architecture

Skills used in designing and deploying reusable components in traditional languages and tools are equally applicable in service-oriented architectures. Component wrappers provide a layer of transparency to Web services developers, but because the legacy and packaged applications and databases were designed for traditional business transactional processing, reusing them in conjunction with Web services may require significant redesign. Also, to extend the legacy components into the new service-oriented-architecture paradigm, enterprises will need skills that can orchestrate the components into new services.

Enterprises that take the service-oriented-architecture path also will need specialists in new environments, such as Java 2 Enterprise Edition and Microsoft .NET, as well as programming languages, such as Java, C# or Visual Basic .NET. This will increase expense.

Action Item: Don't place want ads seeking "five years' experience in Web services." Recognize what enterprise talents and resources may be leveraged to take advantage of your established skills depth.

3.4 The Future

Key Issue: What key trends will shape enterprise use of Web services in coming years?

Web services are traveling a path to pure, standards-reliant implementations. They make it possible to render business elements of increasing complexity — initially

calculations, but ultimately, relationships and materials — as software. By 2007, they will have totally flooded the canals of interenterprise communication.

Today, Web services are pervasive, but shallow. Standards are robust, but usage and value expectations vary among enterprises. The development of a culture of reuse — and the ability to manage massive libraries and interaction streams — is just beginning.

By 2007, considerable progress will have been made in simplifying and standardizing integration efforts. Enterprises will have recognized the value of a consistent, ubiquitous mechanism for interacting internally and externally.

The question will be how much of an enterprise's internal workings should be exposed as Web services. Some believe enterprises should expose as much as possible, each becoming a Web services provider; however, given the interconnections, the cost of maintaining interfaces, the possible impact on connected parties and the risk of overexposure, enterprises must continue to carefully manage touchpoints.

One wild card is application vendors' plan to empower packaged applications with Web services interfaces. Although much of this work is already under way, the completeness of these interfaces, the degree to which service-oriented architectures are embraced, and the stability of advanced transactional capabilities have the potential to accelerate internal and external process connections. By 2007, with respect to simpler Web services, enterprises will have largely overcome light integration issues across heterogeneous environments because of the stability and maturity of core standards.

Enterprises' desire for basic Web services capabilities to drive improved interoperation and data access will butt up against vendor position jockeying and technology hype fluctuations through 2005. Unlike the era of Internet hype, today's technology over-promising does not have the fertilizer of overheated venture capital markets, but it does share other characteristics.

Enterprises that fail to determine the best strategies for executing Web services projects will be disappointed by projects that do not deliver expected value or return on investment. Enterprises that use Web services pragmatically, in concert with generally improved development attitudes, will discover the benefits of cautious dynamism. A foundation of dynamic relations, as enabled by XML, grants Web services and their users a resiliency that has not been available to previous generations of innovation.

Action Items:

- *Balance vendor and advocate hype with feedback from Web services users and developers.*
- *Expect Web services to be broadly available, but limited in scope.*

3.5 Recommendations

- Begin planning now for an internal SOAP deployment to pilot this year.
- Identify appropriate external partners to pilot an external SOAP project in 2004.
- Designate one to three developers to immediately experiment with SOAP interfaces to Google or other external products.