Executive E-mail

Feb. 3, 2005

Building Software That Is Interoperable By Design

Every day, businesses face an ongoing challenge of making a wide variety of software from many different vendors work together. It's crucial to success in streamlining business processes, getting closer to customers and partners, or making mergers and acquisitions successful. Whether you are connecting with partners' systems, accessing data from a mainframe, connecting applications written in different programming languages or trying to log on across multiple systems, bringing heterogeneous technologies together while reducing costs is today a challenge that touches every part of the organization.

Over the years, our industry has tried many approaches to come to grips with the heterogeneity of software. But the solution that has proven consistently effective – and the one that yields the greatest success for developers today – is a strong commitment to interoperability. That means letting different kinds of applications and systems do what they do best, while agreeing on a common "contract" for how disparate systems can communicate to exchange data with one another.

Interoperability is more pragmatic than other approaches, such as attempting to make all systems compatible at the code level, focusing solely on adding new layers of middleware that try to make all systems look and act the same, or seeking to make different systems interchangeable. With a common understanding of basic protocols, different software can interact smoothly with little or no specific knowledge of each other. The Internet is perhaps the most obvious example of this kind of interoperability, where any piece of software can connect and exchange data as long as it adheres to the key protocols.

Simply put, interoperability is a proven approach for dealing with the diversity and heterogeneity of the marketplace. Today I want to focus on two major thrusts of Microsoft's product interoperability strategy: First, we continue to support customers' needs for software that works well with what they have today. Second, we are working with the industry to define a new generation of software and Web services based on eXtensible Markup Language (XML), which enables software to efficiently share information and opens the door to a greater degree of "interoperability by design" across many different kinds of software. Our goal is to harness all the power inherent in modern (and not so modern) business software, and enable them to work together so that the whole is greater than the sum of the parts. We want to further eliminate friction among heterogeneous architectures and applications without compromising their distinctive underlying capabilities.

This may seem like an obvious approach, but the desire for interoperability is sometimes mixed up with other issues. For example, interoperability is sometimes viewed merely as adherence to a published specification of some kind, either from one or more vendors or a standards organization. But simply publishing a specification may not be enough, because it overlooks much of the hard work it takes to successfully develop interoperable products – namely, ensuring that the "contract" defined by a specification is successfully implemented in software and tested in a production environment.

Sometimes interoperability is also confused with open source software. Interoperability is about how different software systems work together. Open source is a methodology for licensing and/or developing software – that may or may not be interoperable. Additionally, the open source development approach encourages the creation of many permutations of the same type of software application, which could add implementation and testing overhead to interoperability efforts.

Working With What You Have

Wholesale replacement of existing technologies is a tough sell for most organizations. They simply have too much invested in a variety of systems from any number of vendors. So, making new software work alongside existing systems is an ongoing customer need. Because of this, Microsoft has consistently invested in helping customers integrate our platform and applications with a broad array of popular (and even not so popular) hardware, software and networks.

As a result of these efforts, Microsoft offers a comprehensive portfolio of interoperability software capabilities, from the operating system to individual applications. Our software works with a vast array of technologies in the marketplace, whether they shipped last week or decades ago. Microsoft software can talk to mainframes and minicomputers from IBM and other manufacturers; other operating systems such as the Mac OS and various UNIXes including Linux; NetWare or AppleTalk networks and native Internet protocols; dozens of programming languages, ranging from COBOL and RPG, through C++ and Java, to the latest experimental languages; hundreds of databases including Oracle, Sybase and DB2; popular business applications like SAP or Siebel; vertical industry standards like SWIFT or HL7; email systems; and infrastructure products providing message queues, directory, management and security.

Many of Microsoft’s products, such as Windows, Office, SQL Server, Exchange and Visual Studio, have significant functionality dedicated to interoperating with non-Microsoft products. Some of Microsoft’s server products are focused squarely on interoperability, such as Host Integration Server for mainframe connectivity, BizTalk Server for heterogeneous integration across multiple applications, or Identity Integration Server, which helps simplify user authentication and management across diverse systems. These resulted from many years of working to understand customer needs and their existing environments.

While our investments in interoperability are mostly focused on the design of our software, we are also involved in work that contributes to interoperability across the industry. Microsoft participates in many formal and informal industry standards organizations to help define the specifications that are a prerequisite for interoperability. We publish APIs, protocols and software development kits, and license our underlying intellectual property associated with this technology, to help others deliver interoperable software. And we collaborate and share technology with a wide array of industry participants, some of them direct competitors, to deliver interoperability solutions that work well with our products.

In fact, in a recent survey by Jupiter Research, 72% of IT managers rated Microsoft technologies as the most interoperable within their existing environments. Similarly, for enhancing interoperability in the financial industry via Web services, Microsoft .NET was recently named by Waters magazine as the best business development environment. This successful approach to interoperability stems in large part from Microsoft’s heritage as a personal computer company: we have always put a lot of emphasis on well-defined mechanisms for how different products from different companies interact, because of the incredible diversity of PC hardware and software. Without a commitment to interoperability, the industry, including Microsoft, would have been stopped in its tracks.

Using XML to Achieve "Interoperability by Design"

While Microsoft software supports an incredibly diverse array of interoperability mechanisms today, most of these are essentially unique efforts, each developed, tested and maintained individually to interoperate with a specific piece of hardware or software. The need to create individual solutions for each interoperability issue results in ever-increasing complexity. Customers and vendors – even companies of Microsoft’s size – face resource limitations as they struggle to keep up with the documentation, testing and minute technical details required by this approach.

To address this issue, Microsoft has been working with the industry to advance a new generation of software that is interoperable by design, reducing the need for custom development and cumbersome testing and certification. These efforts are centered on using XML, which makes information self-describing – and thus more easily understood by different systems. For example, when two systems exchange a purchase order, the attributes of that purchase order are described in XML, so any receiving system can use that description to translate and use the enclosed information. This approach is also the foundation for XML-based Web services, which provide an Internet-based set of protocols for distributed computing. This new model for how software talks to other software has been embraced across the industry. It is the cornerstone of Microsoft .NET and the latest generation of our Visual Studio tools for software developers.

This approach is also evident in the use of XML as the data interoperability framework for Office 2003 and the Office System set of products. Office documents, spreadsheets and forms can be saved in an XML file format that is freely available for anyone to license and use. Office also supports customer-defined XML schema beyond the existing Office document types. This means two things: first, by supporting data in XML, customers can easily unlock information in existing systems and act upon it in familiar Office applications. Second, information created within Office can be easily used by other business applications.

The XML-based architecture for Web services, known as WS-* ("WS-Star"), is being developed in close collaboration with dozens of other companies in the industry including IBM, Sun, Oracle and BEA. This standard set of protocols significantly reduces the cost and complexity of connecting disparate systems, and it enables interoperability not just within the four walls of an organization, but also across the globe. In mid-2003, Forrester Research said that up to a "ten-fold improvement in integration costs will come from service-oriented architectures that use standard software plumbing." Forrester believes those improvements are realistic today. However, the definition of well-designed protocol architecture is just part of the challenge. As part of this collaborative effort, Microsoft and other companies have invested significant resources to ensure that Web services implementations from different companies really are interoperable. This has involved industry workshops, extensive testing, revision of specifications in the face of experience, and even setting up an industry body known as WS-I to help ensure interoperability.

Microsoft's interoperability investments to date have yielded significant benefits to customers and the industry. And we are well aware that we can do even more to help customers and partners achieve greater interoperability to meet their business needs. The foundation we are building with XML is already yielding significant reductions in the time, skill and cost required to integrate systems.

We also see a tremendous opportunity for developers and IT professionals to help usher in a new generation of software that is interoperable by design. We have launched a new Web site (http://www.microsoft.com/interop) that provides more details on the interoperability capabilities of our software. Please take a few moments to visit; you'll find technical information, Webcasts and events intended to help you get the most from your Microsoft products in a heterogeneous software environment.

Bill Gates