

## **The Microsoft .NET Rollout**

### **Clients**

Microsoft .NET is Microsoft's platform for building the next generation of applications based on Web Services. .NET enables developers to build applications that run well on any device, including PCs, cell phones, or Personal Digital Assistants (PDAs). It contains all that's needed to build and run software based on the eXtensible Markup Language (XML), so applications can be built to run on any device connected to the Internet.

With a foundation based on XML and standard Internet protocols, .NET uses software building blocks to make a client smart. It enables a client to easily access critical client information, such as user identity, profile, notification, and data.

With any device, .NET can use XML to access, analyze, and act on data anywhere and at anytime. This makes clients smarter and more productive. It provides the ability to add services to other clients, easily share information among clients, and notify users about the connectivity of other clients, servers, and Web Services.

Applications and data can be optimally presented for input methods, and connectivity made appropriate for end-user interaction, based on connection, state, device type, and user preferences. Clients use Web Services with XML, Simple Object Access Protocol (SOAP), and Universal Description, Discovery and Integration (UDDI). With these applications, .NET gives users control of the information they see on any device. XML uses an open Internet communications protocol to make clients easy to customize.

Microsoft is currently working to make client software smarter and to deliver a personal experience, including Windows XP, Windows Me, and Windows CE, Windows Embedded, the .NET Framework, and the .NET Compact Framework. With this software, the next generation of applications can be built using programmable Web Services.

## **User Experiences**

Microsoft .NET is a strategy for creating the next generation of applications. A key part of developing these applications is providing a personalized and customizable user interface, the .NET user experience.

.NET experiences are user-centric and use identity-based building block services for user identification, preferences, notifications, and storage of user data. Users are always in control of their own data, ensuring accuracy, privacy, and availability of that data between different applications, services, and devices.

A key aspect of .NET is to deliver user experiences seamlessly across multiple devices. Instead of writing a different XML Web Service and a different .NET experience for every possible client, .NET applications can read the client characteristics and end-user selected preferences to deliver a personal user experience.

For example, Web Services such as Microsoft Passport provide users the means to manage their personal information and control access to that information. Other services built on the .NET Platform can also transcend device boundaries and make the most of the far-reaching connectivity of the Internet, providing users with a custom experience anytime, anywhere, on any device.

Microsoft is transitioning several popular products into .NET experiences. Microsoft Office XP is taking the first steps towards providing a .NET experience for knowledge workers by providing native support for XML and Web components, as well as with a feature called Smart Tags.

MSN, including the use of the MSN Explorer local client, is on the path to creating a consumer-focused .NET experience.

bCentral, Microsoft's small business portal, provides XML Web Services such as inventory management and online subscription services to help small businesses get online, improve marketing effectiveness, increase sales, and provide better customer service.

## **Web Services**

Web Services combine the best aspects of component-based application development with the broad reach and flexibility of the Internet. These services are a cornerstone of the Microsoft .NET platform and programming model.

A Web Service is a unit of application logic built using open, standards-based Internet protocols that provide data and services to other applications. Web Services help developers build .NET applications that seamlessly integrate features such as personalization into new and existing applications.

Applications can access Web Services through standard Internet protocols and data formats, such as XML, Hypertext Transfer Protocol (HTTP), and Simple Object Access Protocol (SOAP), without concern about the implementation details of the Web Service.

Since Web Services are standards-based, one XML Web Service can work with all devices, eliminating the need for developers to write a different version of the service for each device. A key strategy of Microsoft .NET is to reduce the amount of code that developers need to write and to extend the reach of the code they do write.

Web Services are provided in terms of the messages that a service accepts and generates, rather than how the service is implemented. By focusing on messages, the Web Services model is independent of the language, platform, and object model. A Web Service can be implemented using any programming language, object model, and platform.

Microsoft.NET and XML Web Services benefit not only developers, but individuals and businesses as well. .NET provides for new business models by enabling companies to commercialize their expertise in new ways. For example, a telecommunications company could use XML Web Services to expose access to voicemail and caller ID, thereby enabling users to get to these services from inside any messaging application, such as Microsoft Outlook.

Technology vendors can migrate current software packages to become XML Web Services, and then sell those services to third parties who require that functionality. Vendors can also provide services to .NET experience vendors who are building new software packages.

Microsoft .NET enables IT departments to tap into vendors' XML Web Services for expertise and outsourced services, reducing internal costs and expanding the capabilities they can deliver to their customers.

End users also benefit from the features of XML Web Services that send and receive messages. .NET allows users to move around in an intelligent, personalized Internet—one that remembers their preferences and delivers the appropriate data at the appropriate time to any device they choose.

## Developer Tools

To help developers build the next generation of applications and application components, such as XML Web Services, Microsoft has created Visual Studio.NET. It provides developers with the tools they need to build Web Services, as well as the capability to integrate Web Services into existing applications.

Visual Studio.NET is a multi-language suite of programming tools hosted in a powerful design and development environment. The development environment includes design features for both Windows forms and Web forms, provides contextual help by exposing MSDN information, and is customizable to provide company-specific coding guidelines directly to the developer.

Visual Studio.NET and applications built using the development tools, depend on the .NET Framework, the layer between the language and the application services. The .NET Framework is a set of programming interfaces designed and built to enable the development of Web Services using any .NET language.

The .NET Framework has several layers.

- The Common Language Runtime manages running code independently of the language used to write that code.
- The .NET Framework classes provide a common type system to all .NET languages.
- ADO.NET enables access to relational and unstructured data. It provides for data exchange and transformation using XML and standard Internet transfer protocols.
- ASP.NET provides personalized user experiences with Web Services.
- Support for any language that complies with the Common Language Specification.

Visual Studio.NET simplifies application development. Developers can build reusable XML Web Services that are easier to write and debug. Web Services collaborate through XML messaging and are independent, so modifying one won't break another. Services are reusable and interfaces are automatically generated, providing seamless deployment.

Because XML Web Services can be written in virtually any programming language (including C, C++, Visual Basic, COBOL, Perl, Python, and Java), developers can use the languages in which they are most productive, while retaining the ability to debug across services or components written in different programming languages.

## Infrastructure

The infrastructure for Microsoft .NET is provided by a suite of application services specifically designed for building, deploying, consuming, and operating XML Web Services. The infrastructure is built on Microsoft Windows 2000 and includes the .NET Enterprise Servers.

Key technologies include native support for XML, scale-up and scale-out capability, and business-process orchestration across disparate applications and services. The .NET Enterprise Servers include:

- Application Center 2000, which enables Web applications built to achieve mission-critical availability through software scaling, while reducing operational complexity and costs.
- BizTalk Server 2000, which offer a suite of tools and services, including business partner orchestration, that enable rapid implementation of secure, reliable trading partner relationships, independent of operating system, programming model, or programming language.
- Commerce Server 2000, which offers users a less complicated and less time-consuming way to build tailored, effective e-commerce solutions.
- Host Integration Server 2000 extends Windows to other systems by providing application, data, and network integration.
- Mobile Information Server 2001, which extends the reach of enterprise data and intranet content to the mobile user. It enables users to securely access their e-mail, contacts, calendar, tasks, or any intranet line-of-business application in real time.
- SQL Server 2000, which is Microsoft's Web-enabled database and data analysis package. SQL Server provides core support for XML and the ability to query across the Internet and beyond the corporate firewall.
- Exchange Server 2000, which supports a wide range of collaborative activities and provides access to information across geographic, organizational, and technology barriers.
- Internet Security and Acceleration Server (ISA) 2000, which provides secure, fast, and manageable Internet connectivity. It builds on Windows 2000 security and directory for policy-based security, acceleration, and management.

Through Windows 2000 and the .NET Enterprise Servers, Microsoft application services provide a solid and flexible infrastructure for building high-performance applications.