Microsoft Enterprise Development Strategy Series

Building Connected Systems:

.NET and the Windows Platform in Financial Services

February 2005

Published: February 2005
For the latest information, please see http://www.microsoft.com/financialservices/

The Microsoft Enterprise Development Strategy Series

The Enterprise Development Strategy Series of briefing papers from Microsoft are designed to deliver overview and technical information of Microsoft application development technologies to developers, architects and IT management. The papers in this occasional series describe strategic goals Microsoft is addressing and summary technical information on these technologies and how they address the needs of the enterprise.

This edition has been customized to highlight the usage of the Windows Platform and .NET in Financial Services.

Feedback is important to us. If you have comments on the series, please e-mail that information with a subject line of “Feedback” to entdevfb@microsoft.com.

Contents

Introduction

This document includes an overview of the Microsoft Platform and how .NET can be used to quickly develop and deploy enterprise level applications for an interconnected world. The document includes a drill down into the .NET Framework, Tools, Applications Services, Orchestration, the usage of Portals, Host Integration, Smart Clients, Smart Devices, Systems Management and the road to the future with the .NET Framework version 2.0 and Visual Studio.NET 2005.

In the ***Fast Fact*** section (beginning on page 29) you will find what the Analyst community is saying about .NET (Gartner, Forrester, Meta, The Middleware Company, Object Watch, and the Waters Group). In the ***Customer Experiences*** section (beginning on Page 31) you will find twenty-five live implementations of .NET and the Windows Platform in Financial Services (Allstate, Bank of Montreal, Barclays, CitiGroup, JPMorgan, Merrill Lynch, NASDAQ, Reuters and many more).

Web Services is a major enabler of the Power of the Microsoft Platform and will be highlighted throughout this document.

Information technology and its effective use to address changing business requirements are central to an organization's ability to compete. With the advent of new technologies and the use of the Internet to achieve tighter integration between an organization and its customers and business partners, the potential of IT has only grown. Successful organizations will be able to realize that potential by building a new generation of connected systems. Connected systems are applications that leverage the network to link the actors and systems that drive business processes. Connected systems pull together a constellation of services and devices, to more effectively meet modern day business challenges. Building connected systems requires a comprehensive enterprise software platform, but also a new service-oriented architectural approach to address the integration imperative. An enterprise software application platform suitable for building connected systems is much more than a traditional application server, and includes such elements as:

- Client and server operating systems.
- Application services such as transactions, messaging, web application support, and security infrastructure.
- A development technology and application runtime.
- Development tools.
- Business process orchestration.
- Pluggable backend servers providing packaged functionality such as database and portal services.
- An applications management environment.
- Enterprise design patterns and practices.

Most organizations utilize products from multiple vendors for these elements of their enterprise software development platform, and hence require that all elements be able to interoperate and integrate with elements supplied by different vendors. This paper describes the comprehensive Microsoft enterprise application development platform for building connected systems. It describes the core components of this platform, including the support for core standards and service-orientation that allow each element to interoperate with elements provided by different software vendors.

Application Platform Requirements, Connected Systems and Service Orientation

Microsoft's comprehensive enterprise application development platform for building connected systems is focused on meeting the following core customer requirements:

- Interoperability and integration
- Productivity
- Security
- Manageability
- Scalability

Microsoft's connected systems strategy centers on the .NET Framework development technologies as the common development framework spanning clients and servers, with a focus on Web Services and industry standards such as SOAP and Extensible Markup Language (XML) for interoperability with components and services provided by other vendors.

Before organizations can reap the benefits of reusing and enhancing their existing technologies, instead of simply upgrading existing systems or building new ones from scratch, they need to adopt modern development architectures centered on the notion of service-orientation. To the Microsoft Corporation, service orientation is a crucial prerequisite to creating connected systems. With the development of messaging standards based on Extensible Markup Language XML, service orientation is quickly becoming the mainstream approach for building connected systems.

The inherent challenge in connecting diverse systems is the translation of platform-specific information and procedural programming models. In an ideal world, we would have the following:

- A standard syntax, in which information from all systems could be unambiguously expressed.
- Standard semantic models so that organizations could express their business practices in a consistent language.
- Standard protocols so that information could be passed across boundaries between operating environments and between organizations.
- A standard means for binding behavior to business documents.

Web Service standards such as SOAP, XML, XSD, WSDL, UDDI and WS-* specifications, such as WS-Security and WS-Policy, are the first constructs of this growing common language. Without the interoperability provided by a platform based on broadly supported standards, service orientation is an arcane practice requiring significant expertise in protocol design, and questionable return on investment. Without Web services that connect enterprise capabilities across heterogeneous platforms, service orientation is significantly less valuable to an organization.

By interacting via standards such as Web Services, SOAP and XML, services spanning multiple platforms and implemented using development technologies from multiple vendors can easily be integrated into comprehensive, connected applications. SOA, as opposed to distributed object systems and/or process-oriented systems, is beneficial in enterprise applications because:

- Complexity is encapsulated. Any system has an inherent complexity the details of which are not important to the users of the system. Service-oriented architectures recognize this complexity, and provide a way to encapsulate it in order to hide it from the consumer of the service.
- More reusability of services across the heterogonous platforms is possible. Platform integration problems are reduced when the functions are defined as services based on interoperable standards such as SOAP and XML. The services can be reused by other components or services running on other platforms.
- Service-oriented architectures are less brittle. Typically, distributed object systems have been tightly coupled—upgrading one component of a distributed object system often requires changing other components, and causes interruption to ongoing application operations. Service-oriented applications can more readily designed for loose-coupling using technologies such as message queuing. This, along with the notion of contracts (typically defined in an XML schema) that separate implementation from interfaces for each service, allows individual services to be upgraded and versioned without requiring changes to other services or affecting ongoing operations of the applications that rely on these services.
- Developer roles are focused and decoupled. A service-oriented architecture allows applications to have many different layers. Developers who are working on the service layer must know transactions, reliability, and messaging, but the client developers only need to know their own programming language in order to connect to and utilize the service.
- Development efforts can be done in parallel. Having many application layers in a project means multiple teams can work on their own service component independently and in parallel after the architecture and design is complete. This solves many problems in enterprise-scale application development.
- The service definition supports multiple client types. Services and clients can be written in any language and deployed in any platform, as long as they can speak the standard languages and protocols that are used.
- More security can be included. By adding the additional service interface layer, it is possible to provide more security. For example, services can be deployed behind firewalls, and different services can be configured with differing levels of security as required, yet still be easily accessed by internal or external service-oriented components.

The Microsoft Enterprise Application Development Platform

The Microsoft enterprise application development platform is a comprehensive platform for building connected systems based on the .NET Framework. The .NET Framework provides a cohesive and comprehensive development environment, as well as a core runtime that enables effective deployment, operations and management of enterprise applications. The .NET Framework and the Microsoft enterprise application development platform, however, was not designed for a homogeneous enterprise. Microsoft based its application platform strategy on the support for industry standards and the understanding that enterprises need the ability to integrate with existing infrastructure and applications, and may desire to select elements of their platform from other vendors but have them easily integrate with .NET-based applications and packaged products from Microsoft. .NET fully enables Web Services, which are fundamentally designed around pluggable services that allow interoperability with other systems. This strategy not only allows companies to create robust, scalable and interoperable applications using current technologies, but positions users of the Microsoft enterprise application development platform to build service-oriented applications today while preserving their investments in existing platforms.

The core elements of the Microsoft enterprise application development platform are depicted below.

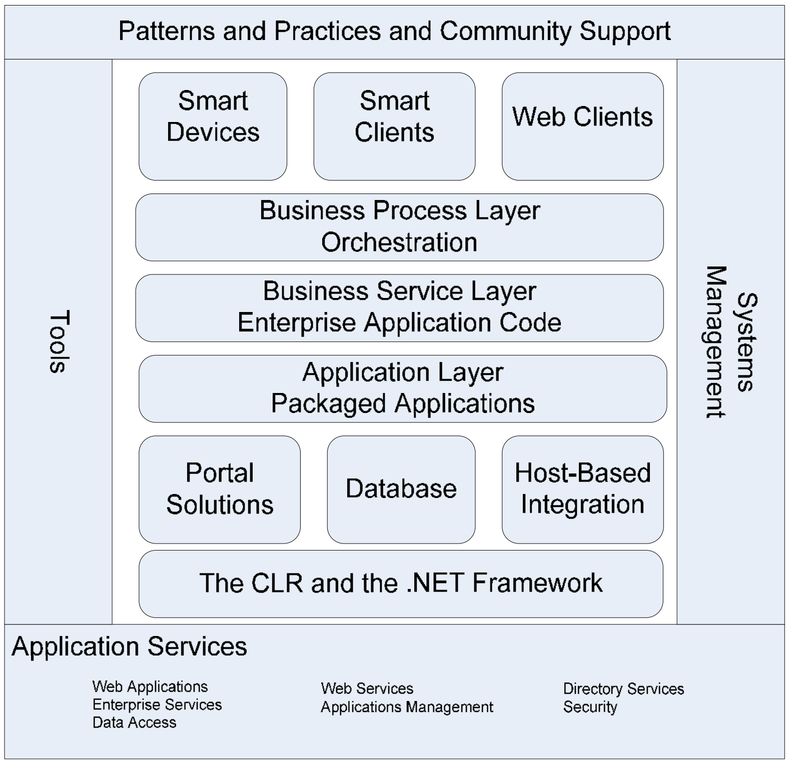

Figure 1: The Microsoft Enterprise Application Development Platform Core Layers

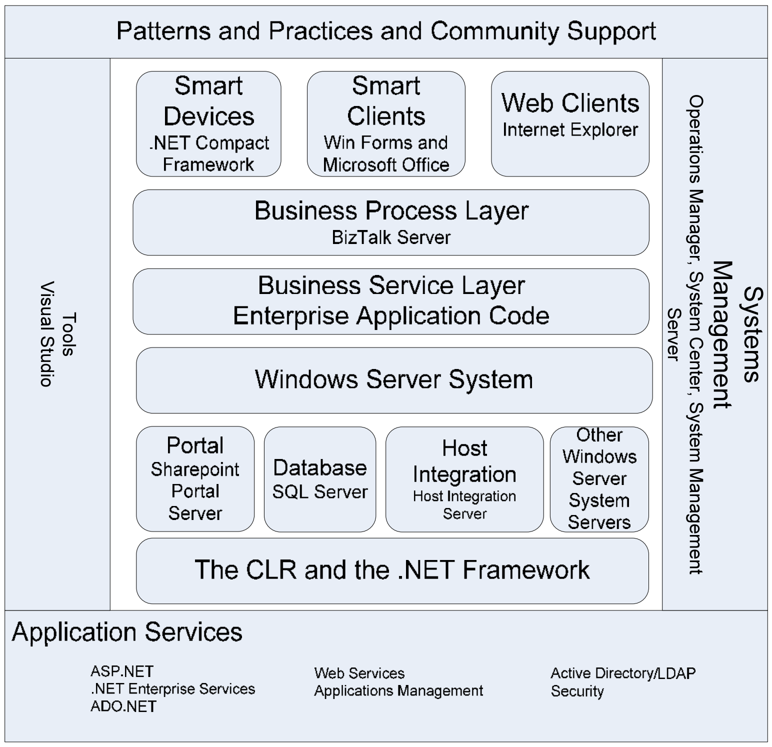

Figure 2: The Microsoft Enterprise Application Development Platform Core Technologies and Products

As depicted in the diagram, the Microsoft platform is comprehensive and integrated, with a solution at every level of the technology stack designed to be programmable via the common .NET Framework and Microsoft Visual Studio® development tool, and easily integrated via a service-oriented architecture. The platform supports all client types, inclusive of smart devices such as PDAs and smart phones, smart clients taking advantage of the processing power on desktop PCs, and thin browser-based clients based on HTML.

Below the client layer, the next layer of the platform includes business process orchestration services, provided by Microsoft BizTalk® Server, that optionally allow organizations to use higher-level business process management tools to aggregate and orchestrate different backend services into complete business processes. BizTalk Server tools allow architects to flowchart such business processes using diagramming tools integrated directly into Visual Studio. The process flows are then compiled automatically into a workflow process that is executed and managed in one or more BizTalk servers.

Individual services and application logic are defined at the business service layer, and these represent discrete custom application logic developed using the .NET Framework. It is important to note, however, that because the Microsoft application platform and .NET are based on industry standards for Web Services (SOAP and XML), clients and back-end services developed in alternative development technologies such as J2EE and running on heterogeneous platforms can be fully integrated into the Microsoft service-oriented architecture. In fact, the ease of such integration and the productivity provided by .NET provide one of the key advantages of the Microsoft application platform.

Beneath the service layer are pluggable back-end server products, provided as packaged applications as part of the Windows Server System™. These back-end servers provide a range of functionality, such as database and transaction processing, portal services, host integration services, collaboration services and the like.

The .NET Framework and the Common Language Runtime (CLR) provide the common development technology and application runtime underlying the complete architecture. The .NET Framework and the CLR are integrated directly into the Windows Server™ 2003 operating system, as are application services such as Web application support, Enterprise Services such as transaction processing and message queuing, Web Services, security, directory, data access and application management/monitoring services. Together, these integrated services combined with the highly productive .NET Framework provide comprehensive backend application server functionality without the need to buy (often) expensive and separate application server products, as is necessary with the use of J2EE. In short, Windows Server 2003 provides the development framework and all of the traditional services such as transactions, queuing, database connectivity, management tools and a managed execution environment as an integrated feature set without additional licensing costs.

Spanning all layers of the architecture is a common development environment for programming and integrating services using .NET: the Visual Studio development tool.

Basic Design Principles

When enterprises consider adopting an application development platform for creating their own applications and integrating with existing ones, there are several basic design principles, which they should look for the platform vendor to follow.

Interoperability and Integration with Application Services

A robust application platform will have a variety of application services which allow developers to create compelling applications. But developers shouldn't have to access each of the services with a different set of tools and with incompatible interfaces. Microsoft's developers can use the .NET Framework to access all of the core application services provided by Windows Server 2003 through a common, well defined interface. Moreover, developers can create new business services that can be exposed through industry standard interfaces (such as SOAP, UDDI and WS-I standards), which allow other enterprise applications using non-Microsoft platforms to interoperate seamlessly.

Productivity

Enterprise systems continue to increase in complexity, especially as applications today must reach out and connect to customers and business partners in ways that were not even possible before the advent of the Internet. The .NET Framework reduces complexity as was designed to dramatically streamline the development process. Developer productivity can also be measured by the number of tools a developer has to master before they become proficient at developing applications for a platform. The Microsoft platform allows developers to use a single tool (Visual Studio) and a single development framework (.NET) to configure a database, develop orchestration for business processes, write service-oriented code, and deploy and test the application. Visual Studio also allows developers to configure and programmatically access server components and application services from non-Microsoft platforms through .NET Framework technologies like web services as well as support for industry standard protocols like HTTP and TCP/IP. In short, developers and architects can use an integrated suite of tools to manage the lifecycle of- an enterprise application.

Security

Given that most new enterprise systems will have some means for connecting to external systems and services, it's critical that an application development platform have the capability to secure an application across every layer. In addition, developers should be able to take advantage of the security services with a minimum of additional knowledge or effort. Perhaps more importantly, administrators and operators should be able to configure and manage an application's security settings (users, groups, code permissions) without requiring developer interaction. The Microsoft enterprise development platform provides Code Access Security that allows application security to be configured at both a very granular level (a single line of code) and a very high level (a business process) with a common set of security objects. Because security is integrated throughout the application stack, administrators and operators can manage permissions in one place, and all the applications running on the Microsoft enterprise development platform can optionally use a common credential store—Active Directory® utilizing role-based security. .NET applications can also interact with application on other platforms using security standards such as X.509 certificates, Kerberos, PKI and WS-Security. In all cases, .NET provides an integrated security framework with comprehensive threat-analysis and written guidelines available through the Microsoft Developer Network (MSDN®) that allow IT managers and developers to design and implement a security infrastructure that fits their needs.

Manageability

For many enterprise applications, the cost of deploying and supporting the application will dwarf the initial development or acquisition costs. Much of these costs can be associated with the tools, technologies, processes and best practices for managing applications. Manageability includes not only the ability to use common tools to monitor, configure and troubleshoot applications but also the ease with which new or updated applications can be deployed. The Microsoft enterprise development platform supports common cross-platform standards to allow an enterprise to manage resources (database, security, network connections, etc.) across multiple applications developed and deployed using application services from different vendors. This allows an enterprise to select the best management tools from any vendor to manage its application services or have the option of purchasing third-party management tools. The Microsoft enterprise development platform allows organizations to realize a reduced TCO by including effective systems management as a core design principle.

Scalability

Over the past several years, scalability considerations have focused on an application server's capability to serve increasingly demanding workloads and user loads. As the industry moves toward service orientation, the determining factor in the viability of an application platform will be its ability to deliver consistent application performance from the Business Process and Business Services layers. And these clients won't be just web pages, but Windows™ applications, WAP and Smart Phones, Smart Devices and even applications running on other platforms in other companies. To help businesses accomplish these goals, the Microsoft enterprise platform is able to scale up (running on 32-bit and 64-bit SMP computers with up to 64 processors) and to scale out (application clustering software allows .NET applications to be deployed across multiple computers for scalability and failover purposes).

In the remaining sections of this white paper, we'll examine each of the core elements of the Microsoft enterprise application development platform in more detail.

The .NET Framework: Common Language Runtime and the .NET Framework Class Libraries

Microsoft .NET development technologies that are integrated directly into Windows Server and Windows desktop operating systems include two core components: the Common Language Runtime which is the managed execution environment for .NET applications, and the .NET Framework Class Libraries which provide a unified development API for a rich set of application services such as data access, security, web-based protocols, user interface, transactions and the like. Microsoft's .NET Framework and its Common Language Runtime (CLR) were designed from the ground-up to enable developers to easily create applications with scalability and security that not only work well with other .NET services and applications, but leverage existing investments in other technologies as well. .NET works with non-Microsoft technologies through its built-in support for creating and consuming Web services based on SOAP and XML, and it works with existing Microsoft technologies by providing a layer to interoperate with COM components written using previous Microsoft development technologies. These features help preserve investments in prior technologies and ensure interoperability with other platforms.

The Common Language Runtime

The Common Language Runtime (CLR) is a managed execution environment that can host applications written in many different programming languages. Microsoft provides C#, Visual Basic.NET, C++ .NET and J# (java syntax) tools and compilers as part of Visual Studio. Third parties such as Borland and Fujitsu and others today provide tools and compilers for over 30 other .NET languages including COBOL, Pascal (Delphi), PERL and many others. The CLR manages memory, thread execution, code execution, code safety verification, compilation, and other system services. These features are intrinsic to the managed code that runs on the common language runtime, and no matter what language is used to program an individual component or service, it is compiled into the same executable .NET intermediate language (IL), shares a common .NET type system, and is fully reusable by .NET services and components written in other languages (including binary interoperability as well as cross-language inheritance between objects written in different languages).

With regards to security, managed components are awarded varying degrees of trust, depending on a number of factors that include their origin (such as the Internet, enterprise network, or local computer). This means that a managed component might or might not be able to perform file-access operations, registry-access operations, or other sensitive functions, even if it is being used in the same active application.

The runtime enforces code access security in conjunction with the deployment of configured security policies for different classes of applications and services. For example, users can trust that an executable embedded in a Web page can play an animation on screen or sing a song, but cannot access their personal data, file system, or network. The security features of the runtime thus enable legitimate Internet-deployed software to be exceptionally feature-rich.

The runtime also enforces code robustness by implementing a strict type-and-code-verification infrastructure called the common type system (CTS). The various Microsoft and third-party language compilers generate managed code that conforms to the CTS. This means that managed code can consume other managed types and instances, while strictly enforcing type fidelity and type safety.

In addition, the managed environment of the runtime eliminates many common software issues. For example, the runtime automatically handles memory management as well as object layout and object references, releasing them when they are no longer being used. Developers no longer need to track individual objects via pointers, as was the case with C++. Automatic memory management and the elimination of pointers resolve the two most common application errors, memory leaks and invalid memory references.

The runtime also accelerates developer productivity. The .NET framework eliminates the need to program low level infrastructure, since developers simply reuse the framework classes themselves which provide for fundamental services such as data access, security, transactions and the like. Also, programmers can write applications in their development language of choice, yet take full advantage of the runtime, the class library, and components written in other languages by other developers. Any compiler vendor who chooses to target the runtime can do so. Language compilers that target the .NET Framework make the features of the .NET Framework available to existing code written in that language, greatly easing the migration process for existing applications.

The runtime is designed to enhance performance. Although the common language runtime provides many standard runtime services, managed code is never interpreted. A feature called just-in-time (JIT) compilation enables all managed code to run in the native machine language of the system on which it is executing. Meanwhile, the memory manager removes the possibilities of fragmented memory and increases memory locality-of-reference to further increase performance.

Finally, the runtime can be hosted by high-performance, server-side applications, such as Microsoft® SQL Server™ and Internet Information Services (IIS). This infrastructure enables you to use managed code to write your business logic, while still enjoying the superior performance of the industry's best enterprise servers that support runtime hosting.

The .NET Framework Class Libraries

The .NET Framework class library is a collection of reusable services that tightly integrate with the common language runtime. The class library is object-oriented, providing types from which your own managed code can derive functionality. This not only makes the .NET Framework types easy to use, but also reduces the time associated with learning new features of the .NET Framework. In addition, third-party components can integrate seamlessly with classes in the .NET Framework, and the Framework can be extended by ISVs and corporations with custom-developed functionality. For example, the .NET Framework Windows Form Controls implements a set of interfaces that developers can use to develop their own Windows Form Controls. These controls will blend seamlessly with the controls already supplied in the .NET Framework.

As one would expect from an object-oriented class library, the .NET Framework types enable developers to accomplish a range of common programming tasks, including tasks such as string management, data collection, database connectivity, and file access. In addition to these common tasks, the class library includes types that support a variety of specialized development scenarios. For example, you can use the .NET Framework to develop the following types of applications and services:

- Console applications
- Windows graphical user interface (GUI) applications (Windows Forms)
- ASP.NET Web applications
- Web services based on the SOAP industry standard

- Windows services

While the .NET Framework and CLR are integrated into Windows Server 2003 and Windows XP (SP1); developers can also download and install the latest .NET Framework runtime and SDK from the Microsoft site at no charge. Once installed, the .NET Framework includes everything that’s necessary to create, compile, deploy and run applications. In fact, you can develop a Windows™ Forms or Web Application using Notepad or any other text editor! But the real power of the .NET Framework is exposed by a robust set of graphical development tools called Visual Studio™.

Tools

Visual Studio provides the single unified development environment for all layers of the Microsoft enterprise development platform inclusive of devices, Web and Windows clients, business process orchestration, and backend services and applications. With intuitive visual designers, high-performance data access tools, server-side visual designers, native support for the Microsoft .NET Compact Framework, and integrated support for XML Web services, Visual Studio .NET delivers a highly productive development experience.

Visual Studio exposes the richness of the entire .NET Framework to developers and system designers. Developers can use the same tool to develop class libraries, Windows ™ services, XML Web Services, Web Applications, Smart Client Applications or any other kind of application or component required. And since other .NET technologies like orchestration, portal services and database services also use or are integrated with the .NET Framework, you can use Visual Studio to develop components that utilize these technologies. One of the biggest benefits of this level of integration is the debugging support in the Microsoft enterprise development platform. Using Visual Studio, a developer can debug a complete system line by line including Web Application code, business process components, data access components, web services and even stored procedures in the database.

The platform on which future service-oriented applications will be built requires extensive support for current Internet standards and, more importantly, emerging XML Web Services standards. Visual Studio is built from the ground up to enable integration through XML Web services. By allowing applications to share data using Internet standards and protocols, XML Web services enable developers to assemble applications from new and existing code, regardless of platform, programming language, or object model. The rich Web Services support makes Visual Studio an excellent tool for integrating services and components written in other languages, such as J2EE.

For enterprise development, Microsoft provides additional functionality in the Enterprise Architect edition of Visual Studio .NET (VSEA). VSEA includes additional capabilities for designing, specifying, and communicating application architecture, development best practices, and application functionality. These capabilities include the ability to:

- Visually model applications, databases, and business processes. Clearly define application functionality and architecture for XML Web services and applications, and visually orchestrate business processes.
- Create sound architectural frameworks and best practices guidelines. Increase application development efficiency through starting application frameworks, sharing best practices guidelines, and simplifying development and management of complex applications.
- Build on a scalable and dependable platform for distributed applications. Create applications with security, reliability, and high performance using the Visual Studio .NET integrated development environment (IDE) and the Windows Server 2003 programming model.

Application Services

According to Gartner, an application server is “system software that resides between the operating system on one side, the external resources – such as DBMS, communications and Internet services – on the other side, and the users’ application on the third side.” In addition, modern application servers include both “MOM (message oriented middleware [sic]) and limited remote database access middleware.” The application services layer of the Microsoft enterprise development platform includes all of these capabilities in an integrated set of services based on the .NET Framework.

Web Applications

Web application services are provided by ASP.NET. ASP.NET exposes a set of rich classes to provide complete functionality for creating, deploying and managing dynamic, web-based applications. ASP.NET directly integrates with Microsoft’s high-performance Web Server—Internet Information Server. ASP.NET combines unprecedented developer productivity with performance, reliability, and deployment. Developers can be more productive with ASP.NET because its reusable server controls enable an HTML-like style of declarative programming that let you build great pages with less code than other tools. Furthermore, displaying data, validating user input, uploading files and other common web application tasks are much simpler to accomplish because ASP.NET has classes which natively support such functionality. The .NET Framework offers classes that encapsulate rich functionality like XML, data access, file upload, regular expressions, image generation, performance monitoring and logging, transactions, message queuing, SMTP mail, and many more. In fact, ASP.NET solves one of the most common enterprise development problems—multiple browser support—by providing native support for pages to work in all modern browsers including Netscape, Opera, AOL, and Internet Explorer. This browser support even extends to browser applications running on standard cellular telephones that can use WAP or MHTML to render pages. And since ASP.NET provides built in support for session state across web farms, developers don’t have to create their own state management solutions.

ASP.NET lets you serve more users on the same hardware because it dramatically increases performance and scalability. Performance is enhanced because ASP.NET dynamically compiles and caches web applications. In addition, ASP.NET output caching and a rich cache API can dramatically improve the performance and scalability of Web applications. ASP.NET applications are more reliable because they can automatically detect and recover from errors like deadlocks and memory leaks to ensure your application is always available to your users. And since ASP.NET applications are based on the .NET Framework, the memory management, code security, type safety and other reliability benefits of the .NET Framework extend to ASP.NET applications automatically.

ASP.NET also dramatically lowers the cost of supporting and deploying web applications. First, ASP.NET dramatically simplifies installation of your application. With ASP.NET, you can deploy an entire application as easily as an HTML page: just copy it to the server. There is no need to register server components using the Windows Registry, and configuration settings can be stored in a single XML file deployed with the application. Second, ASP.NET now lets you update compiled components without restarting the web server. With ASP.NET, you simply copy the component over the existing component (called “XCOPY Deploy”)--and ASP.NET will automatically detect the change and start using the new code without any interruption of uptime for the application. This is possible because .NET will automatically load the new component into memory and start using it for new requests at the same time continuing to service in-progress requests for the existing component until all such requests are completed. Even session state is preserved during such updates.

Data Access

The application services layer of the platform provides extensive support for accessing data via the integrated ADO.NET data access technology. This support includes not only data coming from an enterprise database, but also from XML stores or from data warehouses. The .NET Framework also includes an extensive set of classes for creating and manipulating XML documents and streams. This includes support for standards such as XSLT, XSD and XPath. There are also classes which allow developers to manipulate XML documents using simple data access verbs while preserving the structure and content of the XML document. This extensive XML support allows an application or service to be both a creator and a consumer of complex XML documents as part of an overall enterprise architecture based on XML as a standard protocol. Support for XML is also built into the Visual Studio tools. This includes support for creating well-formed and valid XML documents in the editor and tools to create XML schema files and data sets that conform to them.

In addition to the rich XML support, the application services layer provides extensive support for relational data. Developers can use ADO.NET to access data from all major database systems such as Oracle, DB/2, Microsoft SQL Server, and Sybase. While Microsoft provides ADO.NET drivers for Oracle, SQL Server and DB/2, both IBM and Oracle also provide their own .NET data providers and even tools that integrate within the Visual Studio development tool. The .NET Framework also provides common services like database connection pooling for all managed providers and transaction management for local and distributed transactions via its integrated Enterprise Services classes.

Enterprise Services

The application services layer of the Microsoft enterprise development platform natively support the ability to deliver enterprise scale applications by providing key services including transaction management and message queuing. .NET Enterprise Services provides both of these capabilities via the .NET Enterprise Service Classes. Some of the services available via .NET Enterprise Services include just-in-time (JIT) activation, synchronization, object pooling, message queuing and transactions. In all cases, the underlying infrastructure code is handled by COM+, so developers can focus on the business logic. And in all these cases, COM+ services are exposed to the developer via a set of .NET Framework classes.

Transaction Management

The primary goal of any transaction management system is to ensure the consistency of data across different databases. It is not uncommon to need to update two different systems at once; for example, updating the order entry and inventory systems when a new order is placed. If the order entry system is a commercial package using Oracle and the inventory system is a custom application using SQL Server, you need a way to update both databases as a single transaction; failure to update one database should mean that changes to the other database are rolled back.

Transactions are managed using Distributed Transaction Coordinator (DTC), which is used by .NET Enterprise Services. .NET Enterprise services rely on COM+ transaction management services to coordinate transactions. .NET Enterprise Services offers automatic enrollment in transactions, including management of transactions spanning heterogeneous resources such as message queues and databases; or two different databases running on different machines (for example, a SQL Server database on Windows Server 2003 and an Oracle database running on UNIX). COM+ transactions can also operate over the XA protocol.

Messaging

Enterprise applications have another major interoperability concern: messaging. It is often necessary or desirable to pass messages between disparate systems and create loosely-coupled systems. In situations in which you want to provide messaging services across heterogeneous platforms or to systems that may only be occasionally available, you need an infrastructure in place that is easily integrated with technologies from other companies and easily accessible to the developers of your internal applications.

Microsoft Message Queuing (MSMQ) technology, built into the application services layer of the Microsoft enterprise development platform, enables applications to communicate via message passing. This offers the ability to easily integrate applications on disparate platforms, and also offers the advantage of “loose coupling” such that one subsystem or network may go offline, but the application/user interface continues to function for users. Messages created while the receiving subsystem is offline have guaranteed delivery via MSMQ, and will be processed once the receiving system becomes available again. Microsoft MSMQ provides features such as guaranteed message delivery, efficient routing, security, and priority-based messaging.

Microsoft also provides support for interoperability with other messaging systems provided by other vendors. For example, with the MSMQ-MQSeries Bridge, message queuing can be used to communicate with IBM WebSphere MQ systems. Sending applications can route messages to destination queues, referred to as foreign queues, in the foreign messaging system as if they were direct destination queues within the originating message queuing system.

Web Services

Web Services are a cornerstone of the Microsoft .Net technology and the overall Microsoft enterprise development platform. Support for standards-based Web Services over SOAP and HTTP protocols have been inherent capabilities within the .NET Framework since its first release. Web Services allow disparate systems to be bridged easily, no matter the platforms, development technologies or programming languages used within the bridged systems.

Web Services allow developers to expose business logic and business objects directly over intranets and the Internet via XML. In this way, the business logic exposed can be directly invoked by other applications (for example, a business partner’s application running across the Internet). With support for Web Services across a wide range of application server products such as IBM WebSphere and BEA WebLogic, .NET and J2EE applications can be easily integrated. Microsoft, IBM and others have sponsored the creation of the Web Services Interoperability Organization (WS-I). WS-I provides a forum for the common interpretation of the Web service standards, so technology customers can be confident that different vendors products can fully interoperate via the Web Services standards as specified by the W3C and OASIS standards bodies.

In addition, Microsoft and other vendors are working within the standards process to develop additional advanced Web Service protocols. These enhancements include specifications for additional Web Service security (WS-Security), cross-boundary Web Service transactions

(WS-AtomicTransactions), and reliable messaging between Web Services (WS-Messaging), among others. Microsoft provides support for draft specifications of these developing standards via the Web Services Enhancements toolkit, available for free download from http://msdn.microsoft.com. As these standards are formalized and specifications completed by the standards bodies, Microsoft is fully committed to supporting them as part of the integrated .NET Framework.

Management

Without effective management tools and technologies, there is no ability to either monitor existing applications or effectively deploy new ones. The Microsoft enterprise development platform provides integrated management features and tools to deploy and manage production applications within the data center, as well as desktop resources located outside the data center. The cornerstone of the Microsoft applications management strategy is the Dynamic Systems Initiative (DSI). The Dynamic Systems Initiative (DSI) is an industry effort led by Microsoft to enhance the Microsoft Windows platform and deliver a coordinated set of solutions that dramatically simplify and automate how businesses design, deploy, and operate distributed systems. Microsoft is investing heavily in software research and development and working with partners to deliver end-to-end offerings integrated across application development tools, operating systems, applications, hardware, and management tools that will result in reduced costs, improved reliability, and increased responsiveness throughout the entire IT life cycle.

Support for management and ongoing operations of .NET applications are accomplished via two mechanisms:

- DSI features and management APIs directly embedded within the .NET Framework and Windows Server 2003
- DSI tools that provide management support of the data center

.NET and Windows Server 2003 Management Features

.NET and Windows directly support consuming and emitting Windows Management Instrumentation (WMI) events and allow developers to easily integrate WMI support into their applications. WMI is a standard way to access monitor and manage Windows systems, allowing a standard access method to collect performance and operational information. It is an implementation of the Web Based Enterprise Management standard (WEBM)

.NET applications can also take full advantage of system level monitoring tools like event logs and performance counters that enable IT managers to monitor ongoing operations of .NET applications. The .NET Framework provides objects which wrap all of these system level resources, making it simple for developers to add code that allows their applications to be monitored and also develop sophisticated systems monitoring tools.

.NET also allows dynamic application deployment and updating that can be as simple as copying new application code over the old application code. .NET manages the lifetime of currently created objects and the transition to new ones automatically, so that live updates can be easily rolled out across application clusters with no application downtime. .NET applications running on the server are fully process isolated from other applications, and .NET provides features such as failover and automatic component restarts at the process level, application clustering for sub-second failover at the middle-tier physical machine level, and Microsoft Clustering Services for establishing failover and redundancy at the data tier level. Finally, .NET also includes functionality that makes it easy to automatically update client desktops with new or upgraded functionality with central deployment of smart-client applications from a central server location.

Other DSI features embedded directly within Windows Server 2003 include:

- Network load balancing features balance incoming IP traffic across nodes in a cluster.
- Windows Server clustering provides high availability and scalability for critical applications.
- Windows System Resource Manager enables the allocation of resources, including processor and memory resources, among multiple applications based on business priorities.
- Virtual Disk Service provides a vendor-independent API for identifying and configuring storage devices from multiple vendors in a unified way.
- Automated Deployment Services (ADS) provides rapid server provisioning capabilities and the ability to administer large numbers of Windows Servers from one central location.
- Integrated support for the .NET Framework and ASP.NET provides a fully managed and feature-rich application execution environment for Web-based applications and XML Web services.
- IIS 6.0 is a full-featured Web server with a new fault-tolerant process model that increases the reliability of Web sites and applications.
- Windows Management Instrumentation (WMI) tools provide administrators with unified and direct access to the management functions of local and remote systems.
- Software Update Services enable management of critical-patch releases from Microsoft to automatically deliver them to target computers in an organization from a single intranet.

Directory Services

The Microsoft enterprise application development environment provides a robust, integrated and extensible set of directory services based on Windows Active Directory. The directory services manage users, groups, resources and other system objects and also create and manage the permissions that govern their accessibility. The directory services are exposed to developers through a set of classes in the .NET Framework. This allows developers to easily create directory-aware applications, relieving them from the responsibility of creating their own independent user, group and permission management systems. Moreover, all of the technologies in the platform can use these same permissions to govern access to their resources, including applying role-based security. .NET also fully supports the Web Services directory protocol UDDI.

By providing this level of integration, the platform provides a central management store for credentials and permissions that all applications and technologies can consume.

But the Microsoft enterprise development platform can also fully interoperate with other directory services, such as LDAP. There is several integration mechanisms built into the directory services: .NET applications can use common LDAP interfaces to directly integrate with LDAP services running on other platforms, and administrators can manage and synchronize directory information from multiple directory stores.

Security

The Microsoft enterprise application development platform provides native security services to applications. Security within the platform extends to the code and the Common Language Runtime (CLR)-managed execution environment, which provides application logic sandboxing. To help protect computer systems from malicious code and to provide a way to enable mobile code to run more safely, the .NET Framework provides a security mechanism called Code Access Security (CAS). CAS is a .NET security feature that applies to all .NET-managed code including Web applications, Windows applications, components, web services. Through .NET, administrators can assign a predefined set of permissions to an application. These permission sets vary based on the level of trust accorded to an application. By default, applications receive a level of trust dependent upon the evidence presented about the code's digital signature, origin, and the location of the application. This allows an administrator to control which code is allowed to execute and helps prevent hackers or malicious users from running unauthorized code or substituting a signed component that has appropriate permissions with one that doesn't have the appropriate permissions.

Assuming that the code has the appropriate permissions to execute, it can consume other resources using industry standard mechanisms such as SSL, Kerberos, and X.509 digital certificates. The .NET Framework also provides extensive mechanisms for integrating these standards with emerging Web Services standards in a simple, object-oriented fashion that's integrated with the Visual Studio tools. For example, there are class libraries which allow a developer to build a SOAP header, require an SSL connection to pass it, and then set the user ID and password necessary to authenticate with a remote resource using basic authentication over a secure channel. As new standards emerge, the security system can be extended easily. Developers who want to create interoperable systems based on the latest WS-Security standards can download and install the Web Services Enhancements (WSE) Toolkit to add these advanced Web Services security standards to their services and applications.

Windows Server 2003 and Internet Information Server also provide tight security controls that automatically lock down Web applications from unauthorized access. ASP.NET integrates with these security features and provides an automated authentication mechanism that prevents unauthenticated access to pages so designated within an application. Using a simple configuration file, the security mechanism can be configured to use Windows integrated security or a database of registered users. Users attempting to access an authenticated page are automatically redirected to a designated logon page, and once authentication for a user session is complete, redirected to the requested page—all without any programming required.

One of the most important aspects of Microsoft's security efforts is the campaign to educate developers and system administrators on security and help them create and deploy applications by applying Microsoft-published design patterns and practices. Complete security guides and threat assessment models can be found at http://www.microsoft.com/security/guidance/. The security center for developers and administrators can be found at http://msdn.microsoft.com/security/.

Orchestration

No application or application platform is an island. Even though many are still created with an internal focus, the reality is that connecting applications together has become the norm. Yet connecting software is about more than just exchanging bytes. As organizations move toward a service-oriented architecture (SOA), the real goal—creating business processes that unite separate applications into a coherent whole across application platform boundaries—comes within reach.

Microsoft BizTalk Server provides a development and execution environment for integrating disparate services and applications---including components running on different platforms—into automated business processes. BizTalk tools integrate directly within the Visual Studio IDE, and allow developers and system analysts to build flowcharts of business processes that span organizational boundaries, and then 'bind' these business processes to technical components (including Web Services) that perform the various tasks within the business process. The resulting model is then compiled into an executable workflow that is hosted within the BizTalk server or BizTalk server cluster. BizTalk uses XML and messaging to integrate disparate services running on heterogeneous platforms, and includes connectors and XML schema mapping tools for enterprise application integration. Connectors for SAP, SOAP-based Web Services and IBM WebSphere MQ are built in.

The orchestration services enable enterprises to connect diverse applications, and then to graphically create and modify business processes that use the services that those applications provide. The services provide a mechanism for specifying business rules, and facilities to manage and monitor the running business processes. These orchestration services also include support for information workers. These include a group of Business Activity Services (BAS), a Business Activity Monitoring (BAM) Framework for analyzing running business processes, support for business process provisioning and configuration, and services that enable information workers to set up and manage interactions with business partners.

Because of the increasing demand for integrated Web Services support, the orchestration services have native support for communicating through Web services. In addition, the orchestration services have the ability to define Web services-based business processes by using the Business Process Execution Language for Web Services (BPEL4WS, commonly called "BPEL"). And the orchestration services are built completely around the .NET Framework and Visual Studio.

Biztalk Accelerator for SWIFT

The Microsoft BizTalk Accelerator for SWIFT (A4SWIFT) extends the BizTalk Server platform to provide the most comprehensive, reliable and secure delivery of financial messaging using the SWIFT format and network for customers in banking, capital markets, payments, and corporate finance.

Financial institutions are faced with a daunting challenge: a myriad of applications and systems that need to be integrated with customers, partners, and external financial networks. However, financial institutions can no longer afford integration projects with long delivery cycles and drawn-out returns. Gone are the days of massive, enterprise wide integration implementations, the current economy has changed both the rationale and the techniques used in defining, purchasing, and implementing integration solutions.

(http://www.microsoft.com/biztalk/evaluation/swift/default.asp)

Portal

Support for portal development is provided in a standard set of services based on the .NET Framework, Windows SharePoint Services, and Microsoft SharePoint Portal Server. Windows SharePoint Services is a Windows Server 2003 component that helps organizations increase individual and team productivity by enabling them to create Web sites for information sharing and document collaboration. Sites based on Windows SharePoint Services, called SharePoint sites, take file storage to a new level and help create communities for team collaboration. Users can collaborate on documents, tasks, and events and easily share contacts and other information. In addition, Windows SharePoint Services makes it easy for managers of teams and sites to manage site content and user activity. The environment is designed for flexible deployment, administration, and application development based on ASP.NET and the .NET Framework. Windows SharePoint Services is a free downloadable component available on the Microsoft Web site.

Microsoft SharePoint Portal Server 2003 uses Microsoft Windows SharePoint Services to create portal pages for people, information, and organizations. The portal also extends the capabilities of Microsoft Windows SharePoint Services sites with organization and management tools, and enables teams to publish information in their sites to the entire organization. Microsoft SharePoint Portal Server 2003 enables enterprises to develop an intelligent portal that seamlessly connects users, teams, and knowledge so that people can take advantage of relevant information across business processes to help them work more efficiently. SharePoint Portal Server 2003 provides an enterprise business solution that integrates information from various systems into one solution through single sign-on and enterprise application integration capabilities, with flexible deployment options and management tools. The portal facilitates end-to-end collaboration by enabling aggregation, organization, and search capabilities for people, teams, and information. Users can find relevant information quickly through customization and personalization of portal content and layout, as well as by audience targeting. Organizations can target information, programs, and updates to audiences based on their organizational role, team membership, interest, security group, or any other membership criteria that can be defined. Microsoft SharePoint Portal Server 2003 is fully programmable via the .NET Framework and Visual Studio to enable developers to create custom portal solutions that integrate with other enterprise applications.

Host Integration

An estimated 70 percent of all corporate data is stored on host systems, such as IBM mainframe and AS/400 computers. Yet, increasingly, organizations rely on personal computers together with Web-based and Windows–based applications for everyday productivity and line-of-business solutions. Companies have discovered that Web and Windows solutions often are easier to learn and quicker to implement than comparable host-based applications. To preserve their time and capital investments in host technology, organizations must either migrate all of their host-based resources to the Windows platforms, which can be expensive and time-consuming, or integrate their host-based resources with Windows-based and Web-based solutions.

The Microsoft enterprise application development platform supports the ability to integrate not only host data, but also to integrate directly at the transaction layer with host systems such as CICS environments. Using the Enterprise Services capabilities built into the application services layer, .NET applications can enlist other transactions from a host and can enforce the same transaction and database update integrity over them that a native host program (like CICS or IMS) could enforce over a mainframe transaction.

The host integration functionality is provided through Microsoft Host Integration Server (HIS). Host Integration Server 2004 features and technologies, including network integration, host access with enhanced security, and application integration, enable Windows developers to publish business processes in IBM mainframe and AS/400 applications as XML Web Services, which helps to bring their Host applications and processes into a services oriented architecture.

Smart Clients

Over the last few years, most new enterprise applications have been developed using Internet technologies. These decisions weren't based on the needs of the user, but because the organization recognized cost savings based on the inherent distribution and management advantages of Web applications. As long as the enterprise client has a browser installed, it can consume Web-based applications. But thin-client applications fail to take advantage of the processing power and storage provided (and paid for!) by the enterprise client. So as the limitations of a browser-based interface have become more obvious and the cost of deploying and supporting applications that can take advantage of local processors and storage begin to disappear, a new breed of client – a "Smart Client" has begun to reemerge as an option for building productive applications that take full advantage of the desktop computer. The Microsoft enterprise application development platform provides extensive support for Smart Client development and deployment. Smart Client applications can be divided into two general categories – Windows Forms Applications and custom .NET applications that integrate directly with Microsoft Office as the client.

Windows Forms Applications

The .NET Framework reduces the distribution costs normally associated with Smart Client applications. Once installed on the enterprise desktop, the .NET Framework can automatically download, install, update and execute Smart Client applications from a central server location. This feature is called 'no touch deployment'. Developers can use the .NET Windows Forms class libraries to create rich user interfaces. And the design tools built into Visual Studio allow for rapid application creation and testing of Windows Forms applications. The object oriented nature of the .NET Framework allows developers to create reusable Windows Forms assets which can serve as the basis for a standard enterprise Smart Client framework, increasing the consistency and usability of enterprise Smart Client applications. Windows Forms applications have a development model consistent with Web Form applications based on ASP.NET, and can use native Internet transport protocols such as http, as well as standards-based XML data models to communicate with backend servers. Windows Forms applications, like ASP.NET Web Forms applications can be easily integrated with backend Web Services developed in either Microsoft .NET or other development frameworks such as J2EE.

Office Applications Using Managed Code

Microsoft Visual Studio Tools for the Microsoft Office System is a new technology that brings the power and productivity of Visual Studio .NET and the Microsoft .NET Framework to business solutions built on the current versions of Microsoft Word and Microsoft Excel. With this technology, developers using Visual Studio .NET 2003 can use Microsoft Visual Basic® .NET and Microsoft Visual C#® .NET to write code behind Microsoft Office Word- and Microsoft Office Excel-based applications. These tools also bring the power of Web services to Office 2003 solutions by enabling developers to discover and integrate Web services into their Office solutions by leveraging the support for XML Web Services that's built into the .NET Framework.

Microsoft Office InfoPath™ 2003 is a new offering in the Microsoft Office system that supports the use of schematized forms as an interaction model with back-end services. InfoPath has proven itself particularly useful in structured collaboration scenarios, from human resource enrollment to contract negotiation. Another new Office component that provides access to information via Web services is the Microsoft Office Information Bridge Framework (IBF). IBF is an add-in to Visual Studio .NET that enables developers to build Web-services-based solutions that access enterprise business data such as sales numbers, inventory figures, customer information and more. This information can be viewed directly within the 2003 versions of Word, Excel and Outlook. IBF solutions enhance information worker productivity by enabling them to retrieve and act upon information without leaving the familiar Office applications.

Smart Devices

There are over 100 million devices in use today, including phones, PDAs, and specialty devices like the Blackberry. As cellular networks continue to increase coverage and bandwidth, the number and sophistication of these devices will increase dramatically over the next five years. Only a small percentage of the current devices are Smart Devices. A “Smart Device” is a small form-factor, instant-on device with local processing power, local storage and the ability to create and distribute programs to it. The Microsoft enterprise development platform provides extensive support for Smart Devices, including full .NET programmability for a wide range of devices ranging from the PocketPC PDA to a variety of smart phones that have the .NET Compact Framework directly embedded. Support for core standards such as WAP and WM within ASP.NET also enable .NET–based server applications to directly support virtually any type of cell phone on the market.

Visual Studio provides a common development platform for building .NET applications for such devices, and includes Mobile Controls that provide a visual development environment for building logic and user interface for such devices. This support includes not only editing and debugging support but also database management, web services and XML support. Because the Smart Device support is based on a scaled down version of the .NET Framework called the .NET Compact Framework, developers can easily transfer their .NET development skills to begin creating Smart Device applications. The .NET Compact Framework is able to consume Web Services, enabling enterprises to also create Smart Device applications that can interoperate with backend Web Services programmed in other environment such as J2EE.

System Management

In addition to the Dynamic Systems Initiative management features embedded directly within Windows Server 2003 and the .NET Framework, Microsoft provides core enabling products centered on managing enterprise software environments. These products, also part of the Dynamic Systems Initiative, include Microsoft Operations Manager and Microsoft Systems Management Server.

Microsoft Operations Manager

Microsoft Operations Manager (MOM) 2005 helps provide the knowledge to avoid the avoidable–reducing the complexity associated with managing today's IT infrastructure environment and lowering the cost of operations. MOM 2005 provides manageability as part of the design and implementation of Windows Server System technologies. By delivering operational knowledge and subject expertise directly from the application developers, MOM 2005 helps simplify identification of issues, streamlines the process for determining the root cause of the problem, and facilitates quick resolution to restore services and to prevent potential IT problems. Core features include comprehensive event monitoring and alert management for data centers. Administrators can set, monitor and capture system-level and performance monitor events within any node of their data center, and program automatic responses to handle these events. For example, an alert indicating a node in a cluster has exceeded its available memory or CPU capacity might trigger an alert that would automatically bring a new server into the cluster and/or automatically restart the existing server. Any system level event including all performance monitor counters can be tracked and managed from a central console and integrated into the event/alert system. Microsoft Operations Manager also provides extensive reporting and trend analysis for the data centers operating enterprise applications on the Windows Server 2003 platform.

Systems Management Server

Systems Management Server (SMS) 2003 provides a comprehensive solution for change and configuration management for the Microsoft platform, enabling organizations to provide relevant software and updates to users quickly and cost-effectively. SMS 2003 provides the following key capabilities:

- Application Deployment
- Asset Management
- Security Patch Management
- Mobility
- Windows Management Services Integration
- Integrating Operations and Technology

Community

Microsoft is focused on delivering the guidance necessary to build enterprise applications via Microsoft Patterns & Practices. Extending the traditionally strong guidance for developers available from MSDN, the largest developer community in the world, Microsoft is offering architectural guidance in the form of technical events, books, white papers, reference applications and pattern libraries. Microsoft's Patterns and Practices portal (http://www.microsoft.com/practices/) is the access point for architectural guidance, from information design to solution architecture to the modeling of solutions for deployment into the enterprise datacenter.

Microsoft has also worked hard to develop independent communities which provide the mechanisms for users to support each other and for Microsoft to distribute the most up-to-date information and guidance on particular technologies. For example, communities like www.GotDotNet.com provide workspace functionality where Microsoft product managers, developers, testers and other technical employees can interface directly with corporate end users to discuss and support the guidance provided by the Patterns and Practices team. There are several Framework-specific sites designed to foster interaction between and among product teams and their hard core users. These include sites dedicated to web development (www.asp.net), Windows® Forms development (www.windowsforms.com) and Smart Device Development (http://msdn.microsoft.com/mobility) .

Microsoft also provides support for local resources. There are hundreds of local .NET Developer users groups around the country and the world (www.ineta.org). In addition, Microsoft provides newsgroups where developers and engineers can post questions about the technologies for fast, reliable support.

Microsoft provides a Community for Financial Services Developers. This community includes a quarterly newsletter that highlights the latest technology updates and practical implementation in Financial Services. The community also highlights events including webcasts that are applicable to Financial Services organizations. The community is accessible via the Financial Services Developer Connection at (http://www.financialdevelopers.com)

Windows in Financial Services is a publication that highlights the usage of the Windows Platform in Financial Services. The publication is available at (http://www.windowsfs.com)

Finally, Microsoft's Partner Program is the largest in the industry, and consists of software vendors and system integrators trained and certified on using the .NET Framework and the Microsoft enterprise application development platform to build custom business solutions for any size organizations. Organizations including EDS, Accenture, and IBM Global Services and hundreds of others are available for custom engagements involving development using the .NET Framework.

Road to the Future: .NET Framework 2.0 and Visual Studio 2005

.NET Framework 2.0

The next major release of the .NET Framework is the .NET Framework version 2.0, slated for release in 2005 and currently in beta testing. The Microsoft .NET Framework 2.0 will bring new features and functionality to customers while providing a turnkey upgrade path which in most cases is simply a recompile and run. In addition, the .NET Framework 2.0 can be installed and used side-by-side with the .NET Framework 1.1 on the same machine without conflict, allowing organizations to upgrade easily at their own pace.

The .NET Framework 2.0 core infrastructure enhancements include:

- .NET Framework and CLR enhancements including full 64-bit support.
- New features for ASP.NET that will enable developers to reduce coding required by an estimated 40%-60% with the addition of new prepackaged and reusable server controls.
- Enhanced caching features that enable developers to create middle tier object caches that are automatically refreshed when backend database content changes.
- Enhancements and new productivity features for data access and XML manipulation with ADO.NET 2.0.
- Many more enhancements.

For more information, or to download the beta, see http://msdn.microsoft.com/netframework/.

Visual Studio 2005

Visual Studio 2005 is set to release at the same time as .NET Framework 2.0, and will fully support the new .NET Framework 2.0 feature set. Visual Studio 2005 will also introduce new enterprise team development features with Visual Studio 2005 Team System. Microsoft Visual Studio 2005 Team System provides tools to support the entire software development team:

Architects: Visual Studio 2005 Team Architect Edition includes integrated and productive tools for visually constructing service-oriented solutions that are designed from the onset for their deployment environments. For more information, see Visual Studio 2005 Team System: Designing Distributed Systems for Deployment.

Developers: Visual Studio 2005 Team Developer Edition equips developers with advanced static analysis, code profiling, code coverage, and unit testing tools that enable teams to design for quality, early and often throughout the life cycle. For more information, see Visual Studio 2005 Team System: Building Robust and Reliable Software.

Testers: Visual Studio 2005 Team Test Edition builds on the developer offering to better equip testers with the tools they need to manage and run a wide assortment of tests, including unit tests, manual tests, web tests and advanced load testing tools that enable teams to verify the performance of applications prior to deployment. For more information, see Visual Studio 2005 Team System: Enabling Better Software through Better Testing.

Project Managers: Visual Studio 2005 Team Foundation delivers a series of project management tools based on the software project managers already know: Microsoft Excel, Microsoft Project, and Windows SharePoint Services. With Microsoft Office integration, project managers no longer need to manually map data from these applications onto the data used by the engineering team. A project site provides a dashboard view of project status and drill-down capability to stakeholders. Rich reports open up the metrics collected throughout the natural workflow of the team. A customizable project process, based on industry-proven practices, drives the life cycle. For more information, see Visual Studio 2005 Team System: Software Project Management.

Team Development: Visual Studio 2005 Team Foundation also provides team collaboration tools that enable organizations to effortlessly manage and track the progress and health of software projects. The Portfolio Explorer integrates these same project work products found on the project site into the Visual Studio IDE for effective team access. Visual Studio 2005 Team Foundation also provides an extensible work item tracking system and enterprise-class source code control. For more information, see Visual Studio 2005 Team System: Enterprise-Class Source Control and Work Item Tracking.

For more information, visit http://msdn.microsoft.com/vs2005.

Fast Facts

Microsoft .NET adoption is now exceeding J2EE adoption as the preferred development platform for large organizations. While the two technologies will continue to co-exist, in less than three years since its initial release, the .NET Framework has quickly gained critical mass according to several recent analyst reports. Forrester Research reports in their independent study that .NET is now preferred by 56% to 44% over J2EE in North American firms as their primary development platform. In a Gartner Research study, scientifically constructed based on random sampling of Central IT within large US-based corporations, Microsoft and .NET technologies also surpass J2EE usage for mission-critical applications. This study also shows that Windows Server is now the most widely used server operating system for hosting mission critical business applications. Finally, Gartner also reports in September 2004 that Microsoft leads in Web Services vision and ability to execute, placing Microsoft .NET as the overall winner of its Web Services Magic Quadrant analysis.

Forrester Report: The State of Technology Adoption

(http://www.microsoft.com/forrester)
Forrester Research recently surveyed 878 IT decision-makers at North American enterprises. This research determined that Microsoft .NET has the edge when it comes to choosing a development platform. In interviewing these enterprise software decision-makers, Forrester found that 56 percent were using .NET as their development platform.

Gartner Custom Research Enterprise Application Tracker (PDF)

(http://download.microsoft.com/download/1/9/b/19bc8aa7-05fa-4e86-a612-c2cc181e4ee6/GartnerMissionCriticalApplicationSurvey.pdf)
In 2004, Gartner surveyed workers at a number of large enterprises to determine the platform and development environments for their mission critical applications. This presentation shows their findings. (PDF Download).

Gartner Magic Quadrant for Web-Services-Enabled Software, 3Q04

(http://mediaproducts.gartner.com/gc/webletter/microsoft4_enterprise/article23/article23.html)
Microsoft and IBM continue to lead the Web-services-enabled software market, with Microsoft being placed as the overall Web Services leader according to Gartner in this latest research report.

Developers are more productive with .NET

(http://msdn.microsoft.com/vstudio/java/compare/ibmwebsphere/default.aspx)
In September 2004, The Middleware Company published a detailed study comparing .NET 1.1 on Windows Server 2003 to IBM WebSphere 5.1 on RedHat Linux. The study compares developer productivity, application performance, manageability and reliability of the two platforms. As such, the study is the most comprehensive hands-on comparison published on these two platforms to date, and was unique in its design. To complete the study, Middleware created an application specification for a connected system known as ITS, an automated facilities management application with both B2C and B2B integration via Web Services and Message Queuing, web application front ends and mobile device integration. The Middleware Company then assembled two teams of senior developers (three developers on each team), one to develop the ITS application with WebSphere using J2EE, one to develop the application with Microsoft Visual Studio.NET. Each team was similarly skilled on their respective platforms. The full Middleware Company report can be downloaded from:

http://www.middlewareresearch.com/endeavors/040921IBMDOTNET/endeavor.jsp

Interoperability Projects Benefit from Microsoft-Based Service-Oriented Architectures

(http://www.microsoft.com/windowsserversystem/facts/analyses/objectwatch.mspx)
This ObjectWatch white paper, entitled "Interoperability Through Service-Oriented Architectures (SOAs)," provides a set of criteria for evaluating SOA technologies. It includes the author's interviews with companies that are successfully building SOAs, as well as lessons learned that are applicable to any company interested in leveraging SOAs. The service-oriented architecture approach to interoperability delivers excellent scalability, better ability to leverage existing systems and applications, lower IT costs, and improved user productivity.

Microsoft .NET Development Platform Delivers 25% Lower Development and Support Costs than J2EE/Linux

(http://www.microsoft.com/windowsserversystem/facts/analyses/msnetdev.mspx)
This report by John R. Rymer, vice president of Giga Research, titled "The Total Economic Impact of Developing and Deploying Applications on Microsoft and J2EE/Linux Platforms" analyzes real-world custom application development projects and pinpoints areas where Microsoft tools save money: lower product costs, lower labor costs due to simplified development processes, and lower maintenance costs. The study found that the cost advantage to develop, deploy, support, and maintain custom applications on the Microsoft .NET platform was: • 28.2% less cost for large enterprises and • 25.0% less cost for large- to medium-sized organizations.

Microsoft .NET is Named Best Program Development Platform in Financial Industry by Waters Magazine

(http://www.microsoft.com/presspass/press/2004/dec04/12-07WatersMagBestPR.asp)
"This year, the editors of Waters chose Microsoft .NET as the best program development platform in the financial industry. Microsoft's improved infrastructure provides support, manages patches and upgrades, and addresses security issues promptly," said Phil Albinus, editor of Waters magazine. "By adopting .NET, organizations can leverage their previous investment in technology and move forward with the latest technology."

Customer Experiences

The following documented case studies highlight the Power of the Microsoft Platform and how it is being used with major Financial Services organizations today. Each case study is highlighted with a brief description of the solution and a link to access the entire case study. With the link you will have access to download the case study and in many cases view a video where the customer is sharing their experiences with the Windows Platform and .NET.

Allstate Uses Web Services to Quickly Create Insurance Policy Management Solution
Customers of Allstate Corporation, one of the largest insurance companies in the United States, can manage their property and vehicle policies through the company's consumer Web portal, the Allstate.com Customer Care Center. To also provide customers with access to their life and annuity insurance policies, Allstate took advantage of an existing Web portal—accessAllstate.com—that enabled financial representatives to manage those policies. Allstate used Web Services Enhancements version 2.0 for Microsoft .NET to enable customers to access policies managed through accessAllstate.com while remaining within the familiar Customer Care Center user interface. Allstate built the new feature in two months, saved millions by extending an application instead of duplicating it, and created the foundation for a service-oriented architecture that will accelerate future integration projects.
http://www.microsoft.com/resources/casestudies/CaseStudy.asp?CaseStudyID=16352

Allstate Connects with Countrywide Producer Network in Seven Months Using Microsoft Visual Studio .NET and the .NET Framework
Allstate Financial Group wanted to extend its five existing policy management systems for access by the company's countrywide network of independent producers. Using Microsoft Visual Studio .NET and the Microsoft .NET Framework, Allstate built a solution combining .NET technologies and J2EE to put Allstate Financial information at producers' fingertips. Allstate attributes the success of its project and rapid time to market—seven months from the start of coding to production rollout—to the significant gains in performance, scalability, reliability, and developer productivity provided by Visual Studio .NET and the .NET Framework.
http://www.microsoft.com/resources/casestudies/CaseStudy.asp?CaseStudyID=13648

Bank of America Division Automates Forecasting, Cuts Time on Reports by 98 Percent
The Consumer Real Estate (CRE) Division of Bank of America provides mortgage loans to homebuyers, and adjusts its staffing levels according to fluctuating interest rates and real estate. But manual data collection and analysis were slow and inaccurate, leading to either high employee costs or lowered service quality. To automate its loan forecasting, the company installed OutlookSoft Everest, a Business Performance Management (BPM) solution based on Microsoft® SQL Server™ 2000, Microsoft SQL Server 2000 Analysis Services, and Microsoft Office Excel 2003. As a result, Bank of America has achieved a return on its investment and reduced the number of hours it takes to create reports by 98 percent. The bank projects annual savings of U.S.$4 million to $5 million. Link to Case Study:
http://www.microsoft.com/resources/casestudies/CaseStudy.asp?CaseStudyID=16280

Bank of Montreal Increases Employee Productivity by 23 Percent
Established in 1817 as Bank of Montreal, the Bank of Montreal (BMO) Financial Group is a highly diversified North American financial services organization. BMO provides a broad range of retail banking, wealth management, and investment banking products and solutions. BMO wanted a solution to improve employee communication and knowledge sharing. To

remedy this situation, the financial group created BMO Central—a Web portal in which employees can share information, ideas, documents, and knowledge. Since the creation of BMO Central, employees have experienced a 23 percent increase in productivity.
http://www.microsoft.com/resources/casestudies/CaseStudy.asp?CaseStudyID=16377

Bank of Montreal Selects Microsoft .NET and Smart Client Architecture for Mission-Critical Customer Service Application
BMO Financial Group selected Microsoft Visual Studio .NET and the Microsoft .NET Framework to build a smart client application that uses XML Web services to access the company's host systems. The .NET-connected solution will replace the company's mission-critical customer service application—used by 18,000 retail branch, call center, and back-office personnel to service millions of customer transactions per day. Selecting .NET and taking advantage of its rich support for Web services and smart client applications improves BMO's ability to execute its core IT strategy: improving its ability to service customers by decoupling user desktops from the complexity of back-end systems and accelerating the delivery of smart client solutions that are engineered to user roles.
http://www.microsoft.com/resources/casestudies/CaseStudy.asp?CaseStudyID=14030

Barclays Predicts Significant Returns Thanks to Standardized Microsoft Technologies and Collaboration Software
Barclays Bank promised and delivered shareholders cost reductions of £1 billion (U.S. $1.8 billion) and needed to find new ways of streamlining business processes and increasing efficiency to meet this target. As part of this exercise, the bank chose to standardize its desktops on Microsoft® software, and adopt the Microsoft Office System, such as Microsoft Office SharePoint® Portal Server 2003. Barclays chose Microsoft technology, over the alternatives, for its high-security, integrated, and easy-to-maintain infrastructure in order to manage its vast network of over 41,000 internal users. Enhanced tools enable employees to safely collaborate on business-critical documents regardless of location and speed up the decision-making processes. In addition, improved information access will speed up response times and improve customer service. Barclays Bank expects to see significant financial returns through increased productivity, helping it to quickly achieve further cost-savings targets.
http://www.microsoft.com/resources/casestudies/CaseStudy.asp?CaseStudyID=15737

Bear Stearns Extends Stock Order Functionality to Its Developers and Beyond
A leading investment banking and securities firm, Bear Stearns needed to extend stock order processing functionality residing on an AS/400 to its developers. Access to this functionality is critical in the development of the client applications that employees use to conduct their daily business. Bear Stearns extends applications to external clients as well. Using Microsoft Visual Studio .NET and the .NET Framework, the company built and deployed a set of XML Web services that provide an easy way for Bear Stearns developers to produce powerful, value-adding applications. With the Microsoft .NET Framework, Bear Stearns has exceeded its performance goals, increased developer productivity, and built a solution that adheres to Internet protocols and standards. In addition, the company saved an estimated $250,000 over alternative solutions.
http://www.microsoft.com/resources/casestudies/CaseStudy.asp?CaseStudyID=13655

Broadspire - Migration off Mainframe Reduces Operational Costs by 98 Percent, Yields Two-Month ROI
Third-party insurance administrator Broadspire wanted to migrate the database for its mission-critical claims-processing solution from an IBM S/390 mainframe running DB2 on z/OS to a less expensive platform. After evaluating Oracle on UNIX and Microsoft SQL Server 2000 running on Windows Server 2003, the company chose Microsoft software because it

presented 50 percent lower implementation costs and 60 percent lower long-term costs than the UNIX option. A visit to the Microsoft Technology Center validated that SQL Server 2000 could handle the load, support future growth, and improve application responsiveness. The migration finished 25 percent ahead of schedule and significantly under budget, paying for itself in two months and reducing long-term costs by 98 percent compared to what Broadspire was paying for mainframe hosting services.
http://www.microsoft.com/resources/casestudies/CaseStudy.asp?CaseStudyID=15949

Citigroup CitiVision Integrates 270 Different Sources of Information for 12,000 Global Users in Investment Banking
Citigroup is the largest financial services group in the world, offering a diverse array of corporate and personal financial products and the greatest global distribution capacity of any financial firm. Citigroup has 120 million clients, served by 280,000 employees in 103 countries in the Americas, Europe, Asia and the Pacific, the Middle East, and Africa. More than 90 percent of Citigroup employees are local to the territory that they serve, which makes information delivery and distribution an especially difficult yet vital task for the organization worldwide. Citigroup's investment bankers in particular need up-to-date information aggregated from many diverse sources.
http://www.microsoft.com/resources/casestudies/CaseStudy.asp?CaseStudyID=14014

Corillian Corp. Helps Banking Solutions Provider Surpass Key Benchmark Results
Corillian provides online banking solutions to major financial institutions across the United States, helping them offer services that consumers can use to pay bills, check account balances, transfer funds, and more. Its flagship product is Voyager, an online banking solution built on the Microsoft environment. To demonstrate to major banks how its Microsoft Windows-based software can push the upper limits of scalability and performance across multiple user types and lines of business, Corillian teamed with Microsoft Services Labs. During a three-week session, with the assistance of Microsoft engineers, Corillian achieved benchmark results that set new standards for scalability in online banking services.
http://www.microsoft.com/resources/casestudies/CaseStudy.asp?CaseStudyID=15811

Countrywide Financial – Balboa Insurance Group selected BizTalk Server 2004 to manage over 100,000 documents daily to track information on 14 million loans.
“We use the rules engine for our business rules, making our decisions about how we apply our transactions, but we also use the rules engine for our work-flow routing. One thing we've learned over time is that in this particular application, the business needs to change the work-flow routing continuously. They need to be able to move quickly as events happen externally, as clients come on, as we see things like natural events, like a hurricane hits certain parts of the country, we need to be able to change the order and the way we process things. And by using the rules engine to make these work-flow adjustments, it puts the work flow in control of the business area and gives us complete flexibility.”
http://arsenalcontent/ContentDetail.aspx?ContentID=58919

Dell Sales Tool Can Reduce Dell Sales Call Times by 10 Percent or More, Substantially Improve Profitability
In the past, sales representatives who work at Dell call centers used as many as 40 different information systems to understand customers and meet their needs, which limited their ability to deliver an optimal customer experience. To address that challenge, Dell built the Integrated Dell Desktop (IDD), a smart client solution based on Microsoft software. IDD provides everything that Dell sales representatives need to assist customers and sell the company's products within a single, easy-to-use desktop application—one designed from the ground up to support the sales process and optimize call center operations. Currently rolled out to more

than 8,000 desktops at nine locations, the IDD smart client is helping deliver decreases in average call duration of 10 percent or more, a 45 percent decrease in training time, and improved profit per sale.
http://www.microsoft.com/resources/casestudies/CaseStudy.asp?CaseStudyID=16276

Equifax Sees 14 Percent Lower TCO Deploying Supercomputers on Windows Instead of Linux
Equifax, a global leader in transforming data into intelligence, has migrated many of its key operations from mainframes to Microsoft® Windows®–based Intel processor clusters to enhance the speed and performance of its marketing services capabilities. Selecting Windows over Linux, Equifax has deployed massively parallel supercomputer clusters to streamline operations, reduce costs, and improve customer service. Financial models that once took days to run on the mainframe can now be done in hours, providing customers with faster time to value. Equifax is now able to seamlessly aggregate and analyze data from disparate sources and help customers compile and manage numerous types of credit, financial, public record, demographic, and marketing information in real time. Using Microsoft Windows Server System™, the company has seen 14 percent in cost savings over Linux.
http://www.microsoft.com/resources/casestudies/CaseStudy.asp?CaseStudyID=15528

Farmers Insurance - Insurance Company Streamlines Claims Processing with Web Services–based Solution
The Farmers Insurance Group of Companies, the third largest home and auto insurer in the United States, wanted to expedite automobile claims processing and improve the customer's experience after an accident. Although Farmers had implemented many standards and efficiencies in its claims-settlement process, some aspects were still manual, paper-based, and complex. Farmers turned to Microsoft Certified Partner ProcessClaims and its material-damage management software, ClaimsPort. ClaimsPort integrates seamlessly with Farmers claims-processing system through use of the Microsoft .NET Framework. The automated, Web-based solution gives Farmers the ability to remotely monitor repair facilities, eliminate manual handoffs, reduce cycle time, and make the claims-settlement process more satisfactory for customers.
http://www.microsoft.com/resources/casestudies/CaseStudy.asp?CaseStudyID=15507

JPMorgan Chase Boosts Development Productivity by Building FIX.NET Based Automated Trading System
JPMorgan Chase wanted to build an automated trading system for the bank's European equities traders as quickly as possible to increase the capabilities of the equity trading desk. Traditionally a Java house, the bank chose to build the system using the Microsoft .NET Framework and base it on the FIX.NET solution from Microsoft partner SolutionForge. The system was developed and delivered into production in under four months, proving the agility of .NET development
http://www.microsoft.com/resources/casestudies/CaseStudy.asp?CaseStudyID=14137

London Stock Exchange Sub second Delivery of Stock Market Information with Windows Server 2003 and Visual Studio .NET 2003
Stock market information depreciates faster than almost any other product in the world so speed of delivery is vital to the market professionals who need it to transact business worth billions of dollars every day. When the London Stock Exchange aimed to substantially increase information sales by providing the most current value-added price and trading data available, it wanted to process 500 messages a second with average latency of less than 300 milliseconds. To build a system that could do this, the Exchange worked with Accenture, Microsoft Windows Server 2003, and the Microsoft Visual Studio .NET 2003 integrated development environment (IDE).

http://www.microsoft.com/resources/casestudies/CaseStudy.asp?CaseStudyID=13911

Merrill Lynch & Co. High-Performance Database Helps Brokerage House Position for Future Growth
If, as the saying goes, money makes the world go around, then financial brokerage houses serve as the earth's axis. Understanding that businesses seek peace of mind as much as prosperity, brokerage house Merrill Lynch & Co. needed to enhance its institutional trading business with a database technology that could keep pace with the company's increasing amount of daily transactions data. Merrill Lynch chose Microsoft SQL Server 2000 to provide it with more efficient database storage. As a result of the new solution, Merrill Lynch found a significant increase in its processing performance, including a threefold increase in processing transactions. More importantly, the company now has a larger database capable of matching the company's future business growth potential. At this brokerage house, the earth now spins a little faster.
http://www.microsoft.com/resources/casestudies/CaseStudy.asp?CaseStudyID=15488

NASDAQ Major Stock Market Site Boosts Reliability, Security, Performance, While Reducing Operating Costs
NASDAQ.com, the public Web site of the world's largest electronic stock market, serves millions of investors with market summaries; stock, option, and mutual fund pricing information; and a variety of personal investment tools. With security and reliability a constant concern and challenged by difficult market conditions that demanded better resource utilization, NASDAQ.com decided to upgrade its existing Web server farms to Microsoft Windows Server 2003 operating system with IIS 6.0. Since the switch, NASDAQ.com has been able to reduce the number of Web server farms it uses by half, lower operating costs, decrease downtime by 50 percent, and achieve 99.998 percent uptime.
http://www.microsoft.com/resources/casestudies/CaseStudy.asp?CaseStudyID=14814

National Savings Bank of Serbia (NSBS) Goes from Planning to Launch in Three Weeks
National Savings Bank of Serbia (NSBS), a new retail bank in the former Yugoslav republic, needed to implement a core banking system in just three weeks, to take on 3.5 million customer accounts. The new bank was part of a government rehabilitation strategy for financial services in Serbia after the collapse of 12 banks. Pexim (a Microsoft® certified partner) provided the core banking module and NSBS chose an operating system based on the Microsoft .NET Framework running with Microsoft SQL Server™ 2000 Enterprise Edition to handle customer databases from the 12 failed banks. The solution enabled operations to start within three weeks. Two years on, NSBS is delivering a wider range of products to an ever growing number of customers with no interruptions to the service. Staff productivity has increased because the bank delivers services faster than its competitors.
http://www.microsoft.com/resources/casestudies/CaseStudy.asp?CaseStudyID=15466

Reuters Global Information Provider Lowers Cost of Great Service with Web Services
Reuters wanted to provide innovative products along with outstanding service via a secure, cost-efficient, and flexible integration platform for a new generation of information and analytics products for financial services organizations. The company is building this next generation of enterprise integration capabilities, called Reuters Architecture for Product Integration and Delivery (RAPID) using Microsoft Windows Server 2003, Web services, and Microsoft Visual Studio .NET. Through this move, Reuters was able to get to market about 50 percent faster with a security enhanced integration platform that will support a new generation of products that are quickly and easily developed, deployed, and integrated into the enterprise to create superior value for its customers.
http://www.microsoft.com/resources/casestudies/CaseStudy.asp?CaseStudyID=13933

SunGard Improves Developer Productivity and Customer Solutions Using Microsoft .NET
This paper analyzes how SunGard, a global leader in integrated solutions for the financial services industry, uses Microsoft Visual Studio .NET to enhance developer productivity, and to create business solutions for their customers more rapidly than the competition. SunGard was able to demonstrate significant increases in developer efficiency by bringing there developers from both ends of the language spectrum Microsoft Visual Basic® and C++ under a common language – Microsoft Visual C# --that is approachable by VB developers, while feeling comfortable to C++ developers. SunGard was further able to realize significantly faster time to market due to rapid application development features in Visual Studio .NET, and significant enhancements in the build process and compiler error checking. The solution is based on Microsoft .NET, software that connects information, people, systems, and devices. Microsoft .NET includes the Microsoft Windows .NET Framework programming model, tools such as Microsoft Visual Studio .NET, and a set of servers including Microsoft Windows Server 2003, Microsoft SQL Server, and Microsoft BizTalk Server. These tools and servers integrate, operate, and manage Web services and applications.
http://www.microsoft.com/resources/casestudies/CaseStudy.asp?CaseStudyID=14078

TSYS Credit Card Processor Gains Flexibility, Cuts Development Time with Reporting Solution
TSYS, one of the world's largest third-party processors of credit card transactions, created a new reporting solution that helps it better respond to customer needs—including providing reports based on real-time business intelligence. TSYS ProphIT Reporting Services, based on Microsoft SQL Server 2000 Reporting Services, has helped the company slash the time required to create customer reporting solutions—from days or weeks to a matter of hours. SQL Server 2000 Reporting Services has helped TSYS reduce development costs for internal applications by 75 percent. The three-tier architecture is deployed by using the Microsoft Windows Server 2003 Enterprise Edition and the Windows Server 2003 Web Edition operating systems. SQL Server 2000 Reporting Services is helping TSYS to offer reporting solutions that couldn't have been created with the company's previous mix of reporting tools.
http://www.microsoft.com/resources/casestudies/CaseStudy.asp?CaseStudyID=15557

UnumProvident Boosts Productivity with Web-Based Data System
Each new insurance policy sold by UnumProvident has to pass through 12 to 13 different functional areas for processing – from sales and underwriting to billing and enrollment. But each of these departments had its own system for storing the customer information, which meant redundant data entry, manual transmission of information, inconsistencies and errors. So, the company turned to HP to develop a centralized, Web-based data system which incorporates all of these departments' processes and unifies common customer information onto a single Microsoft SQL Server database. This solution has significantly increased productivity for the company, reducing employee time on task by 50% and eliminating costly errors due to incorrect enrollment materials. These savings generated a return on investment within 12 to 18 months, which has led UnumProvident to expand the solution to more lines of business.
http://www.microsoft.com/resources/casestudies/CaseStudy.asp?CaseStudyID=15009

Wells Fargo Identity Management Solution Saves $350,000, Speeds Development 25 Percent
At Wells Fargo, granting system and service access to new employees was a difficult procedure that sometimes took weeks to complete. Likewise, changing employee access rights as they changed roles was also a challenge. Beyond the drag on employee productivity, the lack of an effective identity management solution was eating into IT budgets because several applications needed custom integration to Human Resources data, resulting

in duplication of effort and inefficient use of corporate resources. To solve these problems, Wells Fargo is deploying a Microsoft-based identity management solution that is on track to save U.S.$350,000 in lower total cost of ownership in its first year of operation. The solution is expected to pay for itself in the first year, enhance productivity and security by making program access reflect employee status almost immediately, and increase agility by bringing new programs online 25 percent faster.
http://www.microsoft.com/resources/casestudies/CaseStudy.asp?CaseStudyID=15331

XRT Treasury Management Suite Cashes In with the .NET Framework and Web Services
XRT designs, develops, and supports a comprehensive suite of cash and treasury management solutions. XRT had a traditional client-server product line, and wanted to build a new Web-based application using a "self-service" treasury model. After considering Java/J2EE, XRT built a financial value chain management application suite with Microsoft Visual Studio .NET and the .NET Framework. Web services allow the application to easily integrate with enterprise resource planning (ERP) and banks.
http://www.microsoft.com/resources/casestudies/CaseStudy.asp?CaseStudyID=15987

Conclusion

With the advent of new technologies and the use of the Internet to achieve tighter integration between an organization and its customers and business partners, the potential of IT has only grown. Successful organizations will be able to realize that potential by building a new generation of connected systems. Connected systems are applications that leverage the network to link the actors and systems that drive business processes.

Microsoft’s comprehensive enterprise application development platform for building connected systems is focused on meeting the following core customer requirements:

- Interoperability and integration
- Productivity
- Security
- Manageability
- Scalability

This paper has provided a high-level overview of the comprehensive Microsoft enterprise application development platform for building connected systems for Financial Services organizations. In so doing, it has touched on core principles such as support for industry standards and service-orientation that allow each element of the Microsoft platform to interoperate with elements provided by different software vendors.

Microsoft®

Your potential. Our passion.™