Introduction

Over the years, our computing lifestyles have changed. Today, everyone sees the value of the Internet, and our computing lifestyle is becoming more and more dependent on Web-based services. Personally, I love to shop, get traffic conditions, compare products, buy tickets, and read product reviews all via the Internet.

However, I'm finding that there are still many things I'd like to do using the Internet that aren't possible today. For example, I'd like to find restaurants in my area that serve a particular cuisine. Furthermore, I'd like to be able to ask if the restaurant has any seating for, say, 7:00 p.m. that night. Or if I had my own business, I might like to know which vendor has a particular item in stock. If multiple vendors can supply me with the item, I'd like to be able to find out which vendor offers the least expensive price for the item or maybe which vendor can deliver the item to me the fastest.

Services like these don't exist today for two main reasons. The first reason is that no standards are in place for integrating all this information. After all, vendors today each have their own way of describing what they sell. The emerging standard for describing all types of information is Extensible Markup Language (XML). The second reason these services don't exist today is the complexity of developing the code necessary to integrate such services.

Microsoft has a vision in which selling services is the way of the future—that is, companies will offer services and interested users can consume these services. Many services will be free; others will be available through a subscription plan, and still others will be charged per use. You can think of these services as the execution of some business logic. Here are some examples of services:

- Validating a credit card purchase
- Getting directions from point A to point B
- Viewing a restaurant's menu
- Booking a flight on an airline, a hotel room, or a rental car
- Updating photos in an online photo album
- Merging your calendar and your children's calendars to plan a family vacation
- Paying a bill from a checking account
- Tracking a package being shipped to you

I could go on and on with ideas for services that any company could implement. Without a doubt, Microsoft will build some of these services and offer them in the near future. Other companies (like yours) will also produce services, some of which might compete with Microsoft in a free market.

So how do we get from where we are today to a world in which all these services are easily available? And how do we produce applications—HTML-based or otherwise—that use and combine these services to produce rich features for the user? For example, if restaurants offered the service of retrieving their menu, an application could be written to query every restaurant's menu, search for a specific cuisine or dish, and then present only those restaurants in the user's own neighborhood in the application.

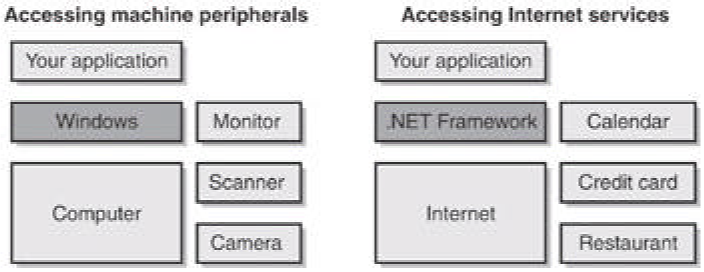
Note

> To create rich applications like these, businesses must offer a programmatic interface to their business logic services. This programmatic interface must be callable remotely using a network, like the Internet. This is what the Microsoft .NET initiative is all about. Simply stated, the .NET initiative is all about connecting information, people, and devices.

Let me explain it this way: Computers have peripherals—mouse, monitor, keyboard, digital cameras, and scanners—connected to them. An operating system, such as Microsoft

Windows, provides a development platform that abstracts the application's access to these peripherals. You can even think of these peripherals as services, in a way.

In this new world, the services (or peripherals) are now connected to the Internet. Developers want an easy way to access these services. Part of the Microsoft .NET initiative is to provide this development platform. The following diagram shows an analogy. On the left, Windows is the development platform that abstracts the hardware peripheral differences from the application developer. On the right, the Microsoft .NET Framework is the development platform that abstracts the XML Web service communication from the application developer.



Although a leader in the development and definition of the standards involved in making this new world possible, Microsoft doesn't own any of the standards. Client machines describe a server request by creating specially formatted XML and then sending it (typically using HTTP) over an intranet or the Internet. Servers know how to parse the XML data, process the client's request, and return the response as XML back to the client. *Simple Object Access Protocol* (SOAP) is the term used to describe the specially formatted XML when it is sent using HTTP.

The following figure shows a bunch of XML Web services all communicating with one another using SOAP with its XML payload. The figure also shows clients running applications that can talk to Web services and even other clients via SOAP (XML). In addition, the figure shows a client getting its results via HTML from a Web server. Here the user probably filled out a Web form, which was sent back to the Web server. The Web server processed the user's request (which involved communicating with some Web services), and the results are ultimately sent back to the user via a standard HTML page.