**What Is Windows Communication Foundation?**

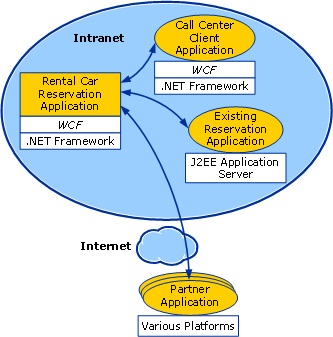
The global acceptance of Web services, which includes standard protocols for application-to-application communication, has changed software development. For example, the functions that Web services now provide include security, distributed transaction coordination, and reliable communication. The benefits of the changes in Web services should be reflected in the tools and technologies that developers use. Windows Communication Foundation (WCF) is designed to offer a manageable approach to distributed computing, broad interoperability, and direct support for service orientation.

WCF simplifies development of connected applications through a new service-oriented programming model. WCF supports many styles of distributed application development by providing a layered architecture. At its base, the WCF channel architecture provides asynchronous, untyped message-passing primitives. Built on top of this base are protocol facilities for secure, reliable, transacted data exchange and broad choice of transport and encoding options.

The typed programming model (called the *service model*) is designed to ease the development of distributed applications and to provide developers with expertise in ASP.NET Web services, .NET Framework remoting, and Enterprise Services, and who are coming to WCF with a familiar development experience. The service model features a straightforward mapping of Web services concepts to those of the .NET Framework common language runtime (CLR), including flexible and extensible mapping of messages to service implementations in languages such as Visual C# or Visual Basic. It includes serialization facilities that enable loose coupling and versioning, and it provides integration and interoperability with existing .NET Framework distributed systems technologies such as Message Queuing (MSMQ), COM+, ASP.NET Web services, Web Services Enhancements (WSE), and a number of other functions.

## Problem Example

The following example illustrates some of the problems that WCF addresses. A car rental company decides to create a new application for reserving cars. The creators of this rental car reservation application know that the business logic it implements must be accessible by other software running both inside and outside their company. Accordingly, they decide to build it in a service-oriented style, with the application’s logic exposed to other software through a well-defined set of services. To implement these services, and thus communicate with other software, the new application will use WCF.



Over its lifetime, the rental car reservation application will likely be accessed by a range of other applications. When it is designed, however, the architects of the rental car reservation application know that its business logic will be accessed, as shown in the preceding figure, by three other kinds of software:

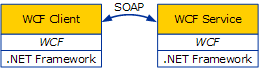
* A call center client application running on the Windows desktops that are used by employees in the organization’s call center. Created specifically for the new reservations system, this application will also be built using the Microsoft .NET Framework and WCF. This application is not truly distinct from the new rental car reservation application, because its only purpose is to act as a client for the new system. From a service-oriented perspective, it is just another client for the reservation system’s business logic.
* An existing reservation application built on a J2EE server running on a non-Windows system. Due to a recent merger with another car rental firm, this existing system must be able to access the new application’s logic to provide customers of the merged firms with a unified experience.
* Partner applications running on a variety of platforms, each located within a company that has a business arrangement with the car rental firm. Partners might include travel agencies, airlines, and others that have a business requirement to make car rental reservations.

The diverse communication requirements for the new rental car reservation application are not simple. For interactions with the call center client application, for instance, performance is important, while interoperability is straightforward, because both are built on the .NET Framework. For communication with the existing J2EE-based reservation application and with the diverse partner applications, however, interoperability becomes the highest goal. The security requirements are also quite different, varying across local Windows-based applications, a J2EE-based application running on another operating system, and a variety of partner applications coming in across the Internet. Even transactional requirements might vary, with only the internal applications being allowed to make transactional requests. How can these diverse business and technical requirements be met without exposing the creators of the new application to unmanageable complexity?

WCF is designed for this diverse but realistic scenario and is the default technology for Windows applications that expose and access services. This topic provides an introduction to WCF, examining what it provides and showing how it is used. Throughout this introduction, the scenario just described will serve as an example. The goal is to make clear what WCF is, show what problems it solves, and illustrate how it solves those problems.

## Addressing the Problem

The foundation for new Windows-based applications is the .NET Framework. Accordingly, WCF is implemented primarily as a set of classes on top of the .NET Framework CLR. Because it extends their familiar environment, WCF enables developers who create object-oriented applications using the .NET Framework today to also build service-oriented applications in a familiar way.



The figure shows a view of a WCF client and service. The two interact using SOAP, the WCF native message representation, so even though the figure shows both parties built on WCF, this is not required. WCF is built on .NET Framework 2.0.

As the scenario described earlier suggests, WCF addresses a range of challenges for communicating applications. Three things stand out, however, as the most important aspects of WCF:

* Unification of existing .NET Framework communication technologies.
* Support for cross-vendor interoperability, including reliability, security, and transactions.
* Explicit service orientation.

#### Unification of Microsoft Distributed Computing Technologies

In the absence of WCF, the development team that implements the rental car application would need to choose the right distributed technology from the multiple choices offered by the .NET Framework. Yet given the diverse requirements of this application, no single technology would fit the requirements. Instead, the application would probably use multiple existing .NET Framework technologies, such as the following:

* ASP.NET Web services (ASMX). An option for communicating with the J2EE-based existing reservation application and with the partner applications across the Internet. Given that basic Web services are supported today on most platforms, this was the most direct way to achieve cross-vendor interoperability before the release of WCF.
* .NET Framework remoting. An option for communication with the call center application, because both are built on the .NET Framework. Remoting is designed expressly for tightly coupled .NET-to-.NET communication, so it offers a seamless and straightforward development experience for applications in the local network.
* Enterprise Services. Used by the rental car reservation application for managing object lifetimes and defining distributed transactions. These functions could be useful in communicating and integrating with any of the other applications in this scenario, but Enterprise Services supports only a limited set of communication options.
* WSE. Could be used along with ASMX to communicate with the J2EE-based reservation application and with the partner applications. Because it implements more recently defined Web services agreements, known collectively as the WS-\* specifications, WSE allows for more flexible Web services security, as long as all applications involved support compatible versions of these new specifications.
* Microsoft Message Queuing (MSMQ). Used to communicate with Windows-based partner applications that require guaranteed data delivery as well as decoupling of workloads and application lifetimes. The durable messaging that Message Queuing provides is typically the best solution for intermittently connected applications.

Built on .NET Framework, the rental car reservation application must use more than one of these communication technologies to meet its requirements. Although this is technically possible, the resulting application would be complex to implement and challenging to maintain.

With WCF, the solution is much easier to implement. As the figure shows, WCF can be used for all the situations previously described. Accordingly, the rental car reservation application can use this single technology for all of its application-to-application communication. The following shows how WCF addresses each of these requirements:

* Because WCF can communicate using Web services, interoperability with other platforms that also support SOAP, such as the leading J2EE-based application servers, is straightforward.
* You can also configure and extend WCF to communicate with Web services using messages not based on SOAP, for example, simple XML formats like RSS.
* Performance is of paramount concern for most businesses. WCF is developed with the goal of being one of the fastest distributed application platform developed by Microsoft. For a high-level performance comparison between WCF and other Microsoft .NET distributed communication technologies, see <http://go.microsoft.com/fwlink/?LinkId=94274>.
* To allow optimal performance when both parties in a communication are built on WCF, the wire encoding used in this case is an optimized binary version of an XML Information Set. Messages still conform to the data structure of a SOAP message, but their encoding uses a binary representation of that data structure rather than the standard angle-brackets-and-text format of the XML 1.0 text encoding. Using this option makes sense for communicating with the call center client application, because it is also built on WCF, and performance is an important concern.
* Managing object lifetimes, defining distributed transactions, and other aspects of Enterprise Services are now provided by WCF. They are available to any WCF-based application, which means that the rental car reservation application can use them with any of the other applications it communicates with.
* Because it supports a large set of the WS-\* specifications, WCF helps provide reliability, security, and transactions when communicating with any platform that also supports these specifications.
* The WCF option for queued messaging, built on Message Queuing, allows applications to use persistent queuing without using another set of application programming interfaces.

The result of this unification is greater functionality and significantly reduced complexity.

## Interoperability with Applications Built on Other Technologies

While WCF introduces a new development environment for distributed applications, it is designed to interoperate well with the non-WCF applications. There are two important aspects to WCF interoperability: interoperability with other platforms, and interoperability with the Microsoft technologies that preceded WCF. The following section describes both.

#### Interoperability with Other Web Services Platforms

Enterprises today typically have systems and applications that they purchased from a range of suppliers. In the rental car application, for instance, communication is required with various other software applications written in various languages and running on various operating systems.

Because WCF’s fundamental communication mechanism is SOAP-based Web services, WCF-based applications can communicate with other software running in a variety of contexts. An application built on WCF can interact with all of the following:

* WCF-based applications running in a different process on the same Windows machine.
* WCF-based applications running on another Windows machine.
* Applications built on other technologies, such as J2EE application servers, that support standard Web services. These applications can be running on Windows machines or on machines running other operating systems.

To allow more than just basic communication, WCF implements Web services technologies defined by the WS-\* specifications. All of these specifications were originally defined by Microsoft, IBM, and other vendors working together. As the specifications become stable, ownership often passes to standards bodies, such as the World Wide Web Consortium (W3C) or the Organization for the Advancement of Structured Information Standards (OASIS). These specifications address several areas, including basic messaging, security, reliability, transactions, and working with a service’s metadata. For more information, see [Interoperability and Integration](http://msdn.microsoft.com/en-us/library/ms730017.aspx). For more information about advanced Web services specifications, see http://go.microsoft.com/fwlink/?LinkId=86603.

Grouped by function, those specifications cover:

* Messaging: SOAP is the foundation for Web services and defines a basic envelope that contains header and a body sections. WS-Addressing defines additions to the SOAP header for addressing SOAP messages, which frees SOAP from relying on the underlying transport protocol, such as HTTP, to carry addressing information. Message Transmission Optimization Mechanism (MTOM) defines an optimized transmission format for SOAP messages with large binary data contents based on the XML-binary Optimized Packaging (XOP) specification.
* Metadata: The Web Services Description Language (WSDL) defines a standard language for specifying services and various aspects of how those services can be used. WS-Policy allows specification of more dynamic aspects of a service’s behavior that cannot be expressed in WSDL, such as a preferred security option. WS-MetadataExchange allows a client to directly request descriptive information about a service, such as its WSDL and its policies, using SOAP.
* Security: WS-Security, WS-SecureConversation, WS-Trust, and WS-Federation all define additions to SOAP messages for providing authentication, data integrity, data privacy, and other security features.
* Reliability: WS-Reliable Messaging defines additions to the SOAP header that allow reliable end-to-end communication, even when one or more Web services intermediaries must be traversed.
* Transactions: Built on WS-Coordination, WS-Atomic Transaction allows coordinating two-phase commit transactions in the context of Web services conversations.

The rental car reservation application would likely use several of these more advanced technologies. For example, WS-Addressing is essential whenever SOAP is used over a transport mechanism other than HTTP, which might be the case for communication with the .NET Framework-based call center client application. WCF relies on WS-Policy and WS-Metadata Exchange to discover whether the system it is communicating with is also using WCF and for other things. Reliable communication is essential for most situations, so it is likely that WS-Reliable Messaging would be used to interact with many of the other applications in this scenario. Similarly, you might also use WS-Security and the related specifications for securing the communication with one or more of the applications, because all would require some kind of protection against unauthorized access or message modification and interception. For the applications that require transaction integration with the rental car reservation system, WS-Atomic Transaction would be essential. Finally, MTOM could be used whenever an optimized wire format for binary data is necessary (for instance for pictures of fleet examples), and both sides of the communication supported this option.

The key point is that WCF implements interoperable Web services complete with cross-platform security, reliability, transactions, and other services. To provide maximum throughput, WCF-to-WCF communication can be significantly optimized, but all other communication uses standard Web services protocols. In fact, it is possible for a single application to expose its services to both kinds of clients.

#### Interoperability with Microsoft Technologies

Many Microsoft customers have made significant investments in the .NET Framework technologies that WCF includes. Protecting those investments was a fundamental goal of WCF’s designers. Installing WCF does not break existing technology, so there is no requirement that organizations change existing applications to use it. A clear upgrade path is provided, however, and wherever possible, WCF interoperates with those earlier technologies.

For example, both WCF and ASMX use SOAP, so WCF-based applications can directly interoperate with those built on ASMX. Existing Enterprise Services applications can also be wrapped with WCF interfaces, allowing them to interoperate with applications built on WCF. And because persistent queuing in WCF relies on MSMQ, WCF-based applications can interoperate directly with non-WCF-based applications built using native MSMQ interfaces. In the rental car reservations application, software built using any of these earlier technologies could directly connect to and use the new system’s WCF-based services.

Interoperability is not always possible, however. For example, even though WSE 1.0 and WSE 2.0 implement some of the same WS-\* specifications as WCF, these earlier technologies implement earlier versions of the specifications. Version 3.0 of WSE does allow interoperability with WCF, but earlier versions do not. For more information about interoperability, see [Migrating WSE 3.0 Web Services to WCF](http://msdn.microsoft.com/en-us/library/ms732008.aspx).

## Interoperability with Other XML Protocols

The future of the Internet is not predictable and the technologies used today may evolve or be replaced. Today, a popular trend in building Web-centric applications (called by many "Web 2.0"), is an application model based on communication using only simple XML formats that are not SOAP-based and exclusively rely on HTTP as a transport and as an application protocol. For example, the Representational State Transfer (REST) architectural style has no notion of user-defined operations for dealing with data. Instead, application state is associated with HTTP URLs and HTTP methods (such as PUT, POST, DELETE, and GET). This approach is in contrast to the creation of user-defined procedures or functions that most developers are familiar with in an enterprise environment. However, the REST approach is of value in scenarios where services must function as the back end of Web 2.0 applications.

REST is just one example of an evolving Web 2.0 technology. In this environment of experimental programming models and ongoing reinterpretation and refinement of standards, flexibility is required to cope with unforeseeable changes. WCF is flexible. For example, while WCF uses SOAP as an underlying structure, it is not bound to using SOAP for wire communication. In fact, WCF can be configured to process "plain" XML data that is not wrapped in a SOAP envelope. WCF can also be extended to support specific XML formats, such as ATOM (a popular RSS standard), and even non-XML formats, such as JavaScript Object Notation (JSON). This flexibility ensures that code written today will be valid in the future, even if protocols change or are replaced. Therefore, WCF was designed for the present and the future.