

Introduction

The Windows Communication Foundation, which was code-named *Indigo*, is a technology that allows pieces of software to communicate with one another. There are many other such technologies, including the Component Object Model and Distributed Component Object Model, Remote Method Invocation, Microsoft Message Queuing (MSMQ), and WebSphere MQ. Each of those works well in a particular scenario, not so well in others, and is of no use at all in some cases. The Windows Communication Foundation is meant to work well in any circumstance in which a Microsoft .NET assembly must exchange data with any other software entity. In fact, the Windows Communication Foundation is meant to always be the very best option. Its performance is at least on par with that of any other alternative and is usually better; it offers at least as many features and probably several more. It is certainly always the easiest solution to program.

Concretely, the Windows Communication Foundation consists of a small number of .NET libraries with several new sets of classes that it adds to the Microsoft .NET Framework class library, for use with the second version, the 2.0 version, of the .NET Common Language Runtime. It also adds some facilities for hosting Windows Communication Foundation solutions to the 5.1 and later versions of Internet Information Services (IIS), the web server built into Windows operating systems.

The Windows Communication Foundation is distributed free of charge as part of a set that includes several other technologies, including the Windows Presentation Foundation, which was code-named *Avalon*, Windows CardSpace, which was code-named *InfoCard*, and the Windows Workflow Foundation. Prior to its release, that group of technologies was called *WinFX*, but it was renamed the *.NET Framework 3.0* in June 2006. Despite that name, the .NET Framework 3.0 is still primarily just a collection of classes added to the .NET Framework 2.0 for use with the 2.0 version of the .NET Common Language Runtime, along with some enhancements to the Windows operating system, as shown in Figure I.1.

One can install the .NET Framework 3.0 on Windows XP Service Pack 2, Windows Server 2003, and Windows Server 2003 R2. The runtime components are pre-installed on Windows Vista. On the successor to Windows Server 2003, which is code-named *Windows Server "Longhorn,"* one can add the .NET Framework 3.0 via the Server Configuration Wizard. Only a very small number of features of the .NET Framework 3.0 will be available exclusively on Windows Vista and later operating systems.

This book does not serve as an encyclopedic reference to the Windows Communication Foundation. Instead, it provides the understanding and knowledge required for most practical applications of the technology.

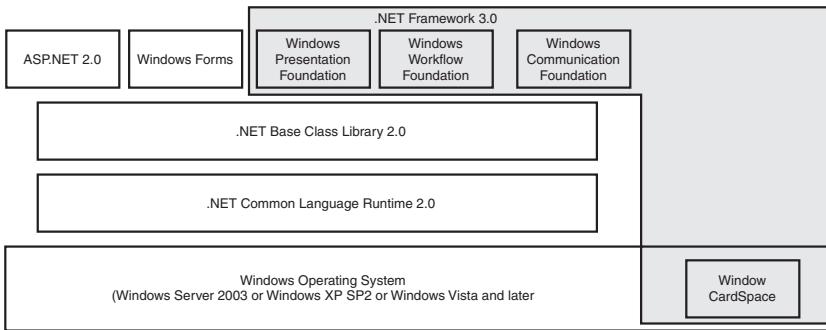


FIGURE I.1 The .NET Framework 3.0.

The book explains the Windows Communication Foundation while showing how to use it. So, typically, each chapter provides the precise steps for building a solution that demonstrates a particular aspect of the technology, along with a thorough explanation of each step. Readers who can program in C#, and who like to learn by doing, will be able to follow the steps. Those who prefer to just read will get a detailed account of the features of the Windows Communication Foundation and see how to use them.

To follow the steps in the chapters, one should have installed any version of Visual Studio 2005 that includes the C# compiler. Free copies are available at <http://msdn.microsoft.com/vstudio/express/>. One should also have IIS, ASP.NET, and MSMQ installed.

The .NET Framework 3.0 is also required, as one might expect. One can download it from <http://www.microsoft.com/downloads/>. The instructions in the chapters assume that all the runtime and developer components of the .NET Framework 3.0 have been installed. It is the runtime components that are preinstalled on Windows Vista, and which can be added via the Server Configuration Wizard on Windows Server “Longhorn.” The developer components consist of a Software Development Kit (SDK) and two enhancements to Visual Studio 2005. The SDK provides documentation, some management tools, and a large number of very useful samples. The enhancements to Visual Studio augment the support provided by IntelliSense for editing configuration files, and provide a visual designer for Windows Workflow Foundation workflows.

To fully utilize Windows CardSpace, which is also covered in this book, one should install Internet Explorer 7. Internet Explorer 7 is also available from <http://www.microsoft.com/downloads>.

Starting points for the solutions built in each of the chapters are available for download from the book’s companion page on the publishers’ website, as well as from <http://www.cryptmaker.com/WindowsCommunicationFoundationUnleashed>. To ensure that Visual Studio does not complain about the sample code being from a location that is not fully trusted, one can, after due consideration, right-click the downloaded archive, choose Properties from the context menu, and click on the button labeled Unblock, shown in Figure I.2, before extracting the files from the archive.

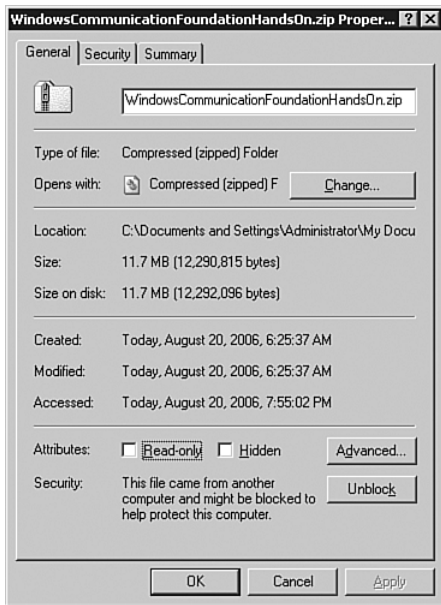


FIGURE I.2 Unblocking a downloaded source code archive.

Note that development on the Vista operating system will be supported for Visual Studio 2005 with an update that will ship after Service Pack 1. Developers working with an earlier version of Visual Studio 2005 on the Vista operating system should anticipate some compatibility issues. To minimize those issues, they can do two things. The first is to disable Vista's User Account Protection feature. The second is to always start Visual Studio 2005 by right-clicking on the executable or the shortcut, selecting Run As from the context menu that appears, and selecting the account of an administrator from the Run As dialog.

This book is considerably different from its predecessor, *Windows Communication Foundation Hands On!* Naturally, the text and samples are up to date with the object model of the final released version of the Windows Communication Foundation. There are also several new chapters, though.

Most significantly, there are two chapters dedicated to the Windows Workflow Foundation and to how to use it with the Windows Communication Foundation. The authors have found that the requirement to have the two technologies work together is commonplace, but know that accomplishing the feat is still quite challenging in the .NET Framework 3.0. Making it simple to do is an important objective only for the next release, which is currently referred to within Microsoft as the *.NET Framework 3.5*.

Whereas the earlier book had just one chapter on Windows CardSpace, this book sports two. Their author, Nigel, once described CardSpace in an internal email as *a sleeping giant of a technology*. It is sleeping because we find that it is the least well-known of the facilities

included in .NET Framework 3.0. Yet it is a giant because it is not only the one technology of which consumers are most likely to become aware, but it is also the one that promises to have the most impact by tangibly improving most people's computing experience and further accelerating the growth of electronic commerce.

In terms of coverage of the Windows Communication Foundation itself, several new chapters have been added. Those chapters provide better coverage of the security features and the extensibility points, as well as offering a passel of recommendations for designing and building Windows Communication Foundation applications gleaned from the authors' work with early adopters.

Many people contributed to this book. The authors would like to thank Joe Long, Eric Zinda, Angela Mills, Omri Gazitt, Steve Swartz, Steve Millet, Mike Vernal, Doug Purdy, Eugene Osvetsky, Daniel Roth, Ford McKinstry, Craig McLuckie, Alex Weinert, Shy Cohen, Yasser Shohoud, Kenny Wolf, Anand Rajagopalan, Jim Johnson, Andy Milligan, Steve Maine, Ram Pamulapati, Ravi Rao, Mark Garbara, Andy Harjanto, T. R. Vishwanath, Doug Walter, Martin Gudgin, Marc Goodner, Giovanni Della-Libera, Kirill Gavrylyuk, Krish Srinivasan, Mark Fussell, Richard Turner, Ami Vora, Ari Bixhorn, Steve Cellini, Neil Hutson, Steve DiMarco, Gianpaolo Carraro, Steve Woodward, James Conard, Nigel Watling, Vittorio Bertocci, Blair Shaw, Jeffrey Schlimmer, Matt Tavis, Mauro Ottoviani, John Frederick, Mark Renfrow, Sean Dixon, Matt Purcell, Cheri Clark, Mauricio Ordonez, Neil Rowe, Donovan Follette, Pat Altimore, Tim Walton, Manu Puri, Ed Pinto, Erik Weiss, Suwat Chitphakdibodin, Govind Ramanathan, Ralph Squillace, John Steer, Brad Severtson, Gary Devendorf, Kavita Kamani, George Kremenliev, Somy Srinivasan, Natasha Jethanandani, Ramesh Seshadri, Lorenz Prem, Laurence Melloul, Clemens Vasters, Joval Lowy, John Justice, David Aiken, Larry Buerk, Wenlong Dong, Nicholas Allen, Carlos Figueira, Ram Poornalingam, Mohammed Makarechian, David Cliffe, David Okonak, Atanu Banerjee, Steven Metsker, Antonio Cruz, Steven Livingstone, Vadim Meleshuk, Elliot Waingold, Yann Christensen, Scott Mason, Jan Alexander, Johan Lindfors, Hanu Kommalapati, Steve Johnson, Tomas Restrepo, Tomasz Janczuk, Garrett Serack, Jeff Baxter, Arun Nanda, Luke Melton, and Al Lee.

A particular debt of gratitude is owed to John Lambert for reviewing the drafts. No one is better qualified to screen the text of a book on a programming technology than an experienced professional software tester. Any mistakes in the pages that follow are solely the fault of the writers, however.

The authors are especially grateful for the support of their wives. They are Marta MacNeill, Kathryn Mercuri, Sylvie Watling, and Libby Winkler. Matt, the only parent so far, would also like to thank his daughter, Grace.