Professional WCF 4

WCF 4高级编程

(美) Pablo Cibraro
Kurt Claeys
Fabio Cozzolino
Johann Grabner
著

吴文国
译

清华大学出版社

WCF 4 高级编程

(美) Pablo Cibraro
Kurt Claeys
Fabio Cozzolino
Johann Grabner 著
吴文国 译

清华大学出版社
北 京

Pablo Cibraro, Kurt Claeys, Fabio Cozzolino, Johann Grabner
Professional WCF 4
EISBN：978-0-470-56314-4

图书在版编目(CIP)数据
WCF 4 高级编程/ (美) 西布拉奥(Cibraro, P.)，克莱斯(Claeys, K.)，科佐利诺(Cozzolino, F.)，格拉布纳(Grabner, J.) 著；吴文国 译. —北京：清华大学出版社，2011.10
书名原文：Professional WCF 4
ISBN 978-7-302-26699-0
Ⅰ. W… Ⅱ. ①西… ②克… ③科… ④格… ⑤吴… Ⅲ. 网络服务器—程序设计 Ⅳ. TP368.5

中国版本图书馆 CIP 数据核字(2011)第 184678 号

责任编辑：王　军　李维杰
装帧设计：牛艳敏
责任校对：胡雁翎
责任印制：何　芊
出版发行：清华大学出版社　　地　　址：北京清华大学学研大厦 A 座
http://www.tup.com.cn　　邮　　编：100084
社　总　机：010-62770175　　邮　　购：010-62786544
投稿与读者服务：010-62776969，c-service@tup.tsinghua.edu.cn
质　量　反　馈：010-62772015，zhiliang@tup.tsinghua.edu.cn
印 刷 者：北京密云胶印厂
装 订 者：三河市金元印装有限公司
经　　销：全国新华书店
开　　本：185×260　印　张：27.5　字　数：669 千字
版　　次：2011 年 10 月第 1 版　　印　　次：2011 年 10 月第1 次印刷
印　　数：1～4000
定　　价：58.00 元

产品编号：041151-01

致　谢

感谢我的夫人 Romina，感谢她在我写本书时的宽容。也特别感谢我的好朋友兼 Tellago 公司的首席技术官 Jesus Rodriguez。他的鼓励和对我自始至终的支持是本书完成的重要保证。最后，非常感谢 Wiley 出版公司的编辑和 Kurt Caleys 给我这个机会，让我能编写一本关于我最喜欢的技术主题的书，即有关 WCF 的书。

——Pablo Cibraro

感谢所有在编写本书的过程中给我提供支持的朋友。没有他们的支持，本书无法完成。感谢我的夫人 Marijke，她允许我没日没夜地躲在计算机旁工作；感谢我的孩子 Milan 和 Timon，在写书的间隙，是他们带给我的快乐，让我的生活充满乐趣。虽然你们年纪尚小，但是我感觉到你们知道我在忙什么：努力工作，实现一个梦想。也感谢我的父母几年前对我的信任，当时我想用计算机做点事情，想要学习信息学。特别感谢 Robert(Bob) Elliott，给我提供编写本书和组建编写队伍的机会。他非常了不起，总是激励处于逆境的人们。感谢我在 Ordina 公司的同事，他们给了我大量的帮助，并给我提供非常有价值的反馈意见。当然，最重要的还是要感谢我的同事 Pablo Cibraro、Fabio Cozzolino 和 Johann Grabner，他们为本书付出了辛勤的劳动和汗水。最后，感谢微软比利时公司的全体同仁，感谢他们珍贵的友谊。

——Kurt Claeys

感谢其他三位合作者，即 Kurt、Pablo 和 Johahn，他们愿意和我一起工作。特别感谢 Pablo 对我和我工作的信任。也感谢 DotNetSide User Group 团队的成员 Tiziana、Vito、Mario、Leo 和 Andrea，他们总是鼓励我参加社区的活动。感谢我的全体同仁，特别是 Alessandro、Benedetto、Mimmo 和 Vincenzo，感谢他们给予我精神上的支持。非常感谢 Claudio，他是我的研究和开发经理，谢谢他的鼓励、支持和关心。

特别感谢 Wiley 公司的编辑们，感谢 Robert Elliot 和 Kelly Talbot，为了组织和协调本书的工作，我与他们交换了大量的 E-mail；感谢 Tricia Liebig、Jeff Riley 和 Doug Holland，他们审查本书的内容并付出了艰辛的劳动。他们的反馈意见是本书成功的关键因素。编写本书需要渡过无数个不眠之夜，特别感谢我父母和我妹妹对我的宽容。

最后，感谢我的未婚妻 Tiziana，在我们准备结婚时，她鼓励和支持我继续本书的工作。千言万语无法表达我对她的感谢，假如没有她的支持，我无法完成这一切，我的生活也会完全不一样。

——Fabio Cozzolino

首先感谢我的家人和父母，感谢他们在感情和经济上给予我的支持、信任和激励。如果没有他们的支持，没有他们提供的衣食保证，不用几个星期我就会放弃这项工作。

作为奥地利南部.NET Usergroup 的发起人之一，我特别感谢社区的成员和微软同事，他们提供了建设性的意见并进行了生动的讨论。

当然我还要感谢 Kelly 和 Bob，他们控制工作进度，保证每人按时完成任务，最重要的是感谢 Kurt Claeys 和 Pablo Cibraro，与他们的合作非常愉快。

我希望本书能为读者提供大量有用的知识，希望他们可以把这些知识应用到实际生活中。对于希望全面、深入地理解 WCF 服务编程的读者来说，本书是不错的起点。

——Johann Grabner

前　言

本书讨论基于.NET 4.0 架构的 WCF(Windows Communication Foundation)服务编程技术。WCF 是.NET 架构中的技术，用来创建面向服务的应用程序、交换不同通信方案中的消息，以及执行由服务操作生成的工作流。通过学习本书，读者将掌握面向服务的基本原理，学习通信模式和发现如何显式定义业务流程。读者还将学习如何使用这些技术的不同部分来实现上述功能，并清晰地理解 WCF 4 的不同组件是如何相互支撑、协作，成为一个综合框架的，进而支持企业级分布式应用程序的各个方面。除了介绍 WCF 技术外，本书还采取了一种实用的讲法，介绍了 3 个案例(面向服务、通信和业务流程)并逐步实现了它们。本书将引导读者如何将 WCF 和 Visual Studio 工具用于实际开发的各个方面。

在构建 WCF 4 知识体系的过程中，读者将掌握如何有效地利用 Visual Studio 2010，以构建能够最大化利用 WCF 4 新增功能的解决方案。

读者将学习如何创建应用程序(作为开发人员和架构师)，如何把这些应用程序集成为 WCF 4 中新的编程范式。读者将学习如何用建立在 WCF 技术和.NET 服务之上的新架构模式创建一个解决方案，将学习如何解决在实现 WCF 4 项目所需要的新编程范式和新结构风格的过程中遇到的实际问题。本书中介绍的实例决不是简单的“hello world”示例，而是引导读者如何建立架构正确的解决方案，并向读者展示最佳的编程风格。

本书 4 位作者都具有丰富的实际开发经验。在日常的开发工作中他们都曾遇到过这些问题，并用最佳的指导和实践提出了切实可行的解决方法。他们把各自的实际经验融入到了本书中。

0.1　本书读者对象

本书专门针对中级.NET 开发人员和解决方案架构师。他们对用 WCF 4 创建面向服务的应用程序、实现可靠的通信、宿主业务流程、在云计算中实现安全及可扩展的集成应用比较感兴趣。

0.2　本书主要内容

本书将介绍以下内容：

- 在一个可靠、正确的架构中设计服务、使用通信和工作流模式
- 实现不同的 WCF 绑定
- WCF 4 消息增强带来的好处

- 如何实例化服务，如何使用代理
- 保护访问服务操作安全的不同方法
- 如何在面向服务的方法中实现 WCF
- 如何用工作流服务组织业务流程，如何创建长时间运行的业务流程
- 如何建立基于云和.NET 服务的集成应用
- 如何创建 RESTful 服务，如何将这些服务应用于轻量级的客户端

0.3 本书组织结构

根据当前具有的 WCF 4 知识，读者可以从任何一章开始阅读本书，但如果读者是初次接触 WCF 4，则建议按顺序阅读本书。第 2 章介绍了一个汽车租赁服务实例，它是本书其他实例的基础，因此不管读者属于哪种情形，在开始钻研其他章节之前必须熟悉第 2 章的内容。

第 1 章介绍了与面向服务、集成和业务流程有关的一些原理和模式，并讨论了它们与 WCF 的关系。本章还讨论了如何用 WCF 实现模式。本章的目的是为 WCF 将要应用的领域提供架构性的背景知识。

第 2 章~第 10 章讨论技术本身、用于开发应用程序的 API 以及它们的配置方法。这 9 章讨论了 WCF 技术的各个方面：绑定、客户端、实例化、工作流流程、安全和.NET 服务。这 9 章旨在为开发人员提供编写 WCF 4 程序所需要的基础知识。

第 11 章~第 13 章向读者介绍如何在 Visual Studio 2010 中实现一个完整的解决方案。这 3 章采用了一种更为实用的方法，介绍了如何利用前 10 章的知识开发一个完整的解决方案。在这 3 章中，读者要用 Visual Studio 完成一个项目，它是某种特定情形的一个解决方案。这些解决方案的代码可以从 www.wrox.com 网站上下载。

每一章都针对一个案例：

- SOA 案例(第 11 章)
- 通信与集成案例(第 12 章)
- 业务流程案例(第 13 章)

本书的最后一章专门介绍服务的托管方法。这一章介绍了如何在 IIS/WAS 和云中托管服务，以及如何用 Windows Server AppFabric 跟踪和管理终结点，另外还解释了路由服务。

0.4 阅读本书的要求

为了学习 WCF，完成本书的示例，读者需要安装 Visual Studio 2010 专业版和.NET 4.0。Visual Studio 2010 必须运行在 Windows XP Service Pack 3(初学版除外)、Windows Vista Service Pack 1 (初学版除外)、Windows 7、Windows Server 2003 SP2、Windows Server 2003 R2、Windows Server 2008 SP2 或 Windows Server 2008 R2 平台上。读者的计算机至少需要 1GB 内存，建议配置更高内存。

0.5　源代码

读者在阅读本书代码时，既可以亲自输入所有代码，也可以使用随书提供的代码文件。本书所有代码均可从 http://www.wrox.com 网站下载。进入该网站后，请根据本书的书名查找本书(读者既可以使用搜索框进行查找，也可以使用书名列表进行查找)，然后单击本书详细内容页面上提供的 Download Code 链接，就可以下载本书提供的所有代码。

注意：

由于许多书籍的名称与本书类似，因此建议读者通过本书的 EISBN 进行查找，本书的 EISBN 为 978-0-470-56314-4。

下载完代码后，读者可以利用压缩工具将代码解压。此外，读者还可以通过访问网站 http://www.wrox.com/dynamic/books/download.aspx 中提供的 Wrox 代码下载页面来获取本书提供的代码，也可以下载 Wrox 出版的其他书籍提供的代码。

0.6　勘误表

为了避免本书文字和代码中存在错误，我们已经竭尽全力。然而，人无完人，错误在所难免。如果读者在我们编写的书籍中发现了诸如拼写错误或代码缺陷等问题，那么请告诉我们，我们对此表示十分感谢。利用勘误表反馈错误信息，可以为其他读者节省大量时间，同时我们也能够受益于读者的建议，这有助于我们编写出质量更高的专业著作。

如果读者需要参考本书的勘误表，请在网站 http://www.wrox.com 中通过搜索框或书名列表查找本书。然后在本书的详细内容页面上，单击 Book Errata 链接。在随后显示的页面中，读者可以看到与本书相关的所有勘误信息，这些信息是由读者提交，并由 Wrox 的编辑们加上的。通过访问 www.wrox.com/misc-pages/booklist.shtml，读者还可以看到由 Wrox 出版的其他所有书籍的勘误表。

如果在 Book Errata 页面上找到您发现的错误，那么请转到页面 www.wrox.com/contact/techsupport.shtml，针对您发现的每一项错误填写表格，并将表格发给我们，我们将对表格内容进行认真审查，如果确实是我们书中的错误，那么我们将在该书的 Book Errata 页面上标明此错误信息，并在后续版本中改正相关错误。

0.7　关于 p2p.wrox.com

如果读者希望与作者进行讨论，或希望能够参与读者的共同讨论，那么请加入 p2p.wrox.com 论坛。这个论坛是一个基于 Web 的系统，读者可以在论坛上发表与 Wrox 出版的书籍有关的技术信息，并与其他读者和技术用户进行讨论。论坛提供了订阅功能，可以将与读者所选主题相关的新帖子定期发送到读者的电子邮箱。Wrox 的作者、编辑、业界专家以及其他读者都会参与论坛中的讨论。

读者可以在 http://p2p.wrox.com 参与多个论坛的讨论，这些论坛不仅能够帮助读者更好地理解本书，还有助于读者更好地开发应用程序。如果读者希望加入论坛，那么请按照以下步骤执行操作：

(1) 进入 http://p2p.wrox.com 页面，单击 Register 链接。

(2) 阅读使用条款，然后单击 Agree。

(3) 填写必要的信息(必要时还需要填写可选信息)，然后单击 Submit。

(4) 随后读者会收到一封电子邮件，邮件中说明了如何验证账号并完成整个加入过程。

注意：

要阅读论坛信息，读者无需加入 P2P 论坛。但是如果需要发表主题或发表回复，那么就必须加入论坛。

成功加入论坛后，读者就可以发表新主题了。此外，读者还可以回复其他主题。读者在任何时间都可以阅读论坛信息。如果读者希望论坛将新的信息发送到自己的电子邮箱中，那么可以单击论坛列表中论坛名称旁的 Subscribe to this Forum 图标，完成该功能设置。

如果读者需要获得更多与 Wrox P2P 相关的信息，请阅读 P2P FAQ，这样可以获得大量与 P2P 和 Wrox 所出版书籍相关的具体信息。阅读 FAQ 时，请单击 P2P 页面上的 FAQ 链接。

目　录

第 1 章

设计原理与设计模式

本章主要内容：

- 探讨服务与 SOA
- 理解通信与集成模式
- 使用业务流程模式

本章介绍与面向服务、集成和业务流程有关的一些原理和模式。通过本章的学习，读者将掌握这些原理与 WCF 之间的关系，以及如何用 WCF 实现这些模式。

1.1 SOA 简介

SOA (Service-Oriented Architecture，面向服务架构)既是一种编程模式，也是软件开发的一种架构方法。根据这种架构方法，应用程序是由具有一定行为(称为“服务”)的功能单元组成的。

服务是一组具有相同要求和功能目标的方法。根据它们的结果(如数据、运算结果等)，其他部分可以调用相应的服务来执行其逻辑。这些函数都有一个明确定义且公开的签名，因此其他代码(服务的客户端程序)可以把这些服务中的函数当作黑盒子那样调用。服务的具体操作是不可见的——它们与用户之间没有直接的交互，而是根据输入参数的要求来运行。SOA 架构允许用户以一定的方式组织分布式应用程序。分布式是指服务的使用者(服务的客户端程序)可以运行在与服务不同的计算机上。这使得业务逻辑与用户界面可以集中处理，而使用者则可以分布在网络的各个位置。为了在 SOA 中实现这个功能，需要在计算机之间传送包含数据的结构化消息。

SOA 的基本思想是构建一个松散耦合的系统，在这个系统中，服务的使用者与服务的实现唯一共同拥有的东西，就是公开的服务操作列表和参数的结构定义。

客户端只知道用来描述服务函数的名称输入参数名称及类型，以及函数返回类型的签

名。除此之外，不存在其他依赖性，不管是客户端还是服务，都可以采用不同的开发平台和编程语言。

显而易见，这种依赖性只是功能上的，而并不基于技术基础架构，这使得不同的开发平台可以很容易与服务进行交互。SOA 架构的技术基础是 SOAP 标准。SOAP 用 XML 语言来定义一个服务操作所发送和接收消息的内容。

消息是由参数的值或返回值形成的，而且数据需要转换成 SOAP 格式。每个开发平台都有一个 SOAP 栈，因此服务的调用会得到很多平台的支持，支持多平台是 SOA 架构的主要目标。

这种方法使得用户可以通过服务来创建系统。服务是应用程序的组成模块，从服务操作可以生成应用程序。不管是终端用户应用程序还是另一个服务，都可以利用这些组成模块。有了 SOA 架构，就可以定义业务流程的工作流，在业务流程中可以调用服务操作。

在这种架构中，实现一个应用程序就是使代码和代码的功能行为可以在未来不可知的情景中重用。由于解除了业务逻辑与用户界面的某种技术之间的耦合，因此对于未来使用更新界面创建技术的客户端来说，仍然可以访问这些服务操作。

不需要考虑业务逻辑与界面之间的关系，这本身就是一种优势。当为某个项目组建一个开发团队时，可以给服务边界的两侧分配不同的小组或成员。

其中一个小组只考虑建立用户界面，不需要考虑业务逻辑和数据访问。UI 小组接收服务接口，根据服务接口设计代码；而另一个小组只需要实现已定义好的服务接口，不需要考虑用户界面(UI)。这意味着一个开发人员不需要负责一个项目的全部过程，包括用户界面、业务逻辑和针对某个特定需求的数据访问。其结果是，每个开发人员都可以把自己的专业知识应用于整个项目开发中的某一个层面。

不需要考虑依赖关系也意味着 UI 和服务的开发可以同时进行，即直接从经双方同意的服务接口的公布日期开始。这带来的最大好处是，可以把 UI 的开发工作外包给其他公司，而实际的业务逻辑则由公司内部完成。

SOA 是构建分布式系统的一个方法。在这种分布式系统中，自治逻辑是通过松散耦合的消息和严格定义的接口调用的。

每个服务接口都需要一个可靠的定义，只有当服务契约被多方认可，且在开发过程中不再发生变化，SOA 架构的优点才能表现出来。业务和需求分析要对系统的功能十分明确。这需要用到功能结构，它在技术层面上定义接口。这一切并不容易，甚至是不可能的，因为对于大多数情形，业务需求总是经常变化的。为了解决这个矛盾，最好经过一个持续时间为 1~4 周的业务迭代过程。每次迭代，都要清楚地分析服务接口未发生变动和已发生变动的部分，并把修改结果报告给开发小组。

当应用程序和软件系统越来越大、越来越复杂时，就需要一个严密的开发架构。这种架构通过大量可重用的组件，来实现更好的可维护性。在当前看到的比较分散的环境中——大多应用程序是在不同的平台中实现的，十分需要一个简单的支持互连的开发方法。

对于仅采用面向对象方法通过对不同组件集成无法解决的大型系统，必须使用 SOA

架构。对现有组件的集成，需要一个经过深思熟虑且适用于整个行业的模式，这就是 SOA 架构。

1.2　SOA 架构的 4 条原则

为了使读者对 SOA 架构有一个更深入的理解，下面详细地讨论 SOA 架构的几条基本原则。在软件行业中，基本原则就是执行准则。在 SOA 架构中，面向服务有 4 条原则。

这 4 条原则是：

- 边界是显式定义的
- 服务是自动的
- 多个服务共享模式和契约，而不是共享类
- 服务的兼容性是基于策略的

1.2.1　边界显式定义

在使用 SOA 架构开发产品时，必须显式定义用户访问实现时必须跨越的边界。服务运行时所处的进程和内存空间必须独立于引用服务的客户端程序所在的进程和内存空间。定义边界时需要有超前的思想，并且必须与每个可能的参与者进行通信。边界可以用契约和地址的方式来定义，服务通过这些地址可以被访问到。这些信息非常重要，必须要容易访问。

没有契约和地址就无法执行服务中的逻辑。只能通过一种办法执行服务中的逻辑，即通过调用契约，因此也称契约为边界。边界必须是显式定义的，是指客户端程序只需要知道服务中存在的函数，通过契约运行这些函数即可。这条原则也意味着，必须对所有可能发生的异常事件进行描述，一个方法只有当以显式获知的数据结构形式，或者以一个包含异常事件详细信息的结构的形式，给出所需要的答案时才会停止执行。没有明确的允许，数据既不可以进入服务操作，也不可以离开服务操作。

1.2.2　服务自动化

服务是不依赖于其他服务的行为的独立程序模块。服务不需要显式实例化就可以直接引用。服务必须经过部署，而且每个服务的版本相互独立，安装了某个服务的新版本不会影响其他服务的行为。类在编译后的可执行文件中是相互耦合的，而服务则不同。服务使用一种松散耦合的架构。

1.2.3　服务共享的是模式和契约，而不是类

模式是对服务操作的定义，它以一种独立于平台的方式来描述签名：函数的名称、参数类型和返回值类型。契约是服务的元数据，是服务作为黑盒子的对外接口，模式则是对参数结构的定义。这条原则明确表示，类本身(不管是代码意义上的类，还是 UML 意义上的类)不可以被服务及服务的客户端共享。