HZ BOOKS
华章科技

由经验丰富的.NET专家和架构师亲自执笔，技术专家和技术社区联袂推荐！

以实践为导向，循序渐进地讲解了企业级.NET应用架构与设计的流程、方法、原则、模式与最佳实践，极具实践指导意义。

實戰

汪洋 著

Architecture and Design for .NET: Principles, Patterns, and Practices

.NET应用架构设计
原则、模式与实践

机械工业出版社
China Machine Press

實戰

Architecture and Design for .NET: Principles, Patterns, and Practices

.NET应用架构设计
原则、模式与实践

汪洋 著

机械工业出版社
China Machine Press

国内首本讲解企业级.NET应用架构与设计的专著，由国内经验丰富的.NET专家和架构师亲自执笔，权威性毋庸置疑。本书的重点不在于架构与设计的理论，而是从实践的角度出发，结合大量示例和一个完整的项目案例循序渐进地讲解了.NET应用架构与设计的方法、流程、原则、模式和最佳实践，实践指导意义极强。本书在写作方式上打破了传统的知识灌输方式，而是用逐步演进的方式去引导和启发读者的抽象思维和宏观思想，从而让读者快速掌握架构与设计的精髓。

全书一共分为三个部分：第一部分首先介绍了企业应用架构与设计的流程和核心概念，然后讲解了应用架构中常用的设计模式和设计原则，以及常用的几种设计方法；第二部分的主题是架构与设计的方法和最佳实践，既对架构分层的相关知识进行了详细的阐述，又用大量实战案例对业务层、服务层、数据访问层、数据存储层、显示层的原理和设计进行了深入的剖析；第三部分以一个真实的项目案例（已上线）演示了企业级应用的架构与设计的流程和方法，旨在帮助读者将前面所学的知识融会贯通，从而真正达到能动手实践的目的。

图书在版编目（CIP）数据

.NET应用架构设计：原则、模式与实践 / 汪洋著. —北京：机械工业出版社，2012.1

ISBN 978-7-111-36536-5

Ⅰ.N…　Ⅱ.汪…　Ⅲ.计算机网络－程序设计　Ⅳ.TP393

中国版本图书馆CIP数据核字（2011）第240410号

机械工业出版社（北京市西城区百万庄大街22号　邮政编码　100037）
责任编辑：杨绣国　　秦　健
北京京师印务有限公司印刷
2012年1月第1版第1次印刷
186mm×240mm·26.5印张
标准书号：ISBN 978-7-111-36536-5
定价：69.00元

凡购本书，如有缺页、倒页、脱页，由本社发行部调换
客服热线：（010）88378991；88361066
购书热线：（010）68326294；88379649；68995259
投稿热线：（010）88379604
读者信箱：hzjsj@hzbook.com

前　言

为什么要写这本书

自从进入IT行业之后，我就一直被“架构师”这个名词所吸引。架构师有扎实的技术能力、强大的组织与沟通能力，是一个极具挑战性的角色！微软巨人比尔·盖茨不习惯称自己为CEO，而称自己为首席架构师！架构师像电影《盗梦空间》中的造梦者，让无数的程序员向往和追求。

一路走来，听过也见过不少朋友兴致勃勃地谈论与架构相关的话题。每次我在社区（ITPUB、博客园、51CTO等）发布架构设计相关的文章，也都会引来大批读者的回帖和跟帖。和很多做技术的朋友一样，我自己也在一直苦苦地追求着“架构之梦”，四处寻找与架构相关的资料，甚至希望有真正的架构师能够为自己指点一二，以便打开自己的“任督二脉”！

在学习的过程中，我一次又一次地将自己之前的“原以为”推翻，然后又一次次地重新厘清观念，同时给自己一个新的目标，继续前行，这是一个非常痛苦的过程！原以为，架构师的核心能力在技术上，于是不断地通过学习和看书来了解相关的技术。后来发现只看书是不够的，架构师应该实战经验很丰富，于是又不断地尝试做各种项目。随后我又发现，架构师的设计能力应该非常强，可以将整个项目在头脑中实现，于是我又开始学习设计，在脑中设计，然后用代码实现。再后来，我发现架构师的业务分析能力要强，于是我又开始在每次的项目中积极主动地学习业务知识，大胆地与团队、客户沟通和协调。再后来……

直到今天，我依然在架构之路上摸索着，不过总算也积累了一些心得和失败的经验，希望能与大家分享。

在摸索和学习的过程中，我发现架构设计的资料非常少，尤其是在.NET领域，而且很多

架构设计的书籍都写得比较高深，无法让人一下理解，就算勉强理解了，也不会在项目中使用，有时候，就算生搬硬套地用了，思维层次上却没有得到提升。

同时，我在写博客和回帖的过程中了解到：用演化的方式去讲述知识比较容易让人理解和接受，并且可以让读者的思维和作者一起展开，所以我觉得应注重思想的改变，而不是生硬地记忆。

另外，我认为实用的知识能进一步强化和巩固我们的所学。很多书上的示例就仅仅是例子而已，其作用类似于“Hello World”，在实际项目中意义不大！读者看完之后，可能确实能明白这到底是什么，但是“如何用”还得自己摸索。

基于以上原因，本书选择了从实战和分析的角度来撰写，希望可以从思维层次上提升读者的能力，帮助大家走上架构设计之路。

读者对象

本书的读者可分为以下几类：

- 有经验，并且想学习架构设计的开发人员。
- 想学习设计方法、设计模式、架构模式的开发人员。
- 想学习如何将 .NET 中各种技术结合使用的开发人员。
- 想学习 .NET 技术内核的开发人员。
- 想学习如何使用 .NET 中各种开源组件、框架的开发人员。

另外，高级工程师、架构师也可以参考本书。

如何阅读本书

本书分为三大部分，每一部分都是从实战的角度出发，其中第三部分为项目实战，是对第一部分和第二部分的全面总结和巩固！

本书的每一个部分都涵盖一些高端的话题，而且从不同的角度阐述了相关知识。

第一部分涉及架构与设计的原则和模式，主要介绍了架构与设计的流程和核心概念、模式、设计原则和方法。

第二部分是架构与设计的方法和最佳实践，其中对架构分层、业务层、数据访问层、服务层、数据存储层、显示层进行了详细的剖析，并包含大量示例和最佳实践。

第三部分为项目实战，这一部分实现了一个完整的项目案例，能指导读者如何把前面两个部分的知识综合起来运用！

附录是与 Content Delivery Network （CDN）相关的知识，书中有关性能的讨论会涉及 CDN 知识。

勘误和支持

由于作者水平有限，编写时间也很仓促，书中难免会存在一些错误或不准确的地方，恳请读者批评指正。

同时，我特意将本书第三部分的项目案例部署到了网上（地址：http://www.agilesharp.com）。一方面是希望大家对书中的案例有一个清晰的认识，能实实在在地看到项目的结果；另一方面是希望通过这个项目实体让大家亲自动手去实践。

大家可以将书中的错误发布在 Bug 勘误表页面。另外，如果大家在学习或项目中遇到任何问题，也可以访问 Q&A 页面，我将尽量在线上为大家提供最满意的解答。书中的全部源文件都会发布在这个网站上，我也会及时提供相应的功能更新。如果大家有更多的宝贵意见，也欢迎发送邮件到我的邮箱，邮箱地址为：yangyang4502@hotmail.com，我很期待听到你们真挚的反馈。

致谢

首先要感谢微软开创了 .NET 平台，造就了一个新的技术时代，它成为了我的“衣食父母”。

感谢机械工业出版社的编辑杨福川，在这一年多的时间里他始终支持我的写作，他的鼓励和帮助引导着我顺利完成全部书稿。感谢一直给我审稿的编辑杨绣国，为了这本书，她一次又一次地与我交流和沟通！还要感谢机械工业出版社的其他朋友们，你们为此书付出了很多，辛苦了！

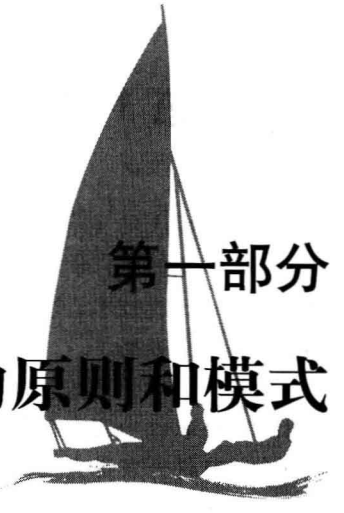
我特别感谢我的爱人潘燕。在写书的这段时间内，我一直没有时间顾及家里的事情，也没有时间陪她。感谢她的体谅与支持！每次我写完一章，她都会耐心地检查和指正，她是这本书的第一个读者！

最后我要感谢我的爸爸、妈妈、岳父、岳母，感谢他们将我和我的爱人培养成人，是他们给了我们信心和力量！

谨以此书，献给我最亲爱的家人，以及众多热爱技术、追逐梦想的朋友们。

汪洋

目　录



第一部分

架构与设计的原则和模式

第 1 章　架构与设计的流程和核心概念

很多开发人员（不管处于哪个阶层）对架构设计特别着迷，甚至达到了痴迷和神往的地步。其实，相对而言学习编程可能是一件比较容易的事情（只要有好的学习态度和编程习惯，然后掌握一定的技术，很快就可以成为一个比较出色的开发人员），若想要练就架构设计的能力就不是那么容易了。

架构，近似于艺术，有着对美的要求和对完美的追求，但它又和艺术不一样，比如：架构设计可以在软件中实实在在地体现出来，一个合适的架构会让系统有很好的灵活性和伸缩性，并且会使软件开发进入良性循环；相反，一个不合适的架构则很容易导致软件在变化的需求中崩溃，甚至会让开发进入“死亡行军”的状况中。就因为如此，架构才会显得如此神秘，同时又让人难舍对它的追求。

本书将会和大家一起探讨有关架构的话题，不过，架构相关内容覆盖面很广，想要通过一本书就把它讲清楚几乎是不可能的。本书从实践角度出发，着重讲述架构设计中可以采用的原则和模式，以及一些思想，希望在大家的架构之路上能够献上一点绵薄之力。

本章是全书中理论最多的一章，主要是希望大家对架构师、架构设计有一定的了解，为理解后续的章节内容打下基础。

1.1　正确认识软件架构

不同的人对架构有着不一样的理解。无论是在建筑行业、软件行业，还是在网络或书本上，“架构”一词有着各种各样的解释。在编程世界中，很多对架构的解释都是从技术层面来定义的，而且在不同的技术或平台上面，对架构的定义又不太一样，甚至有些定义和理解失去了其原本的核心意思，这就使得部分架构学习者感到迷茫。

如果认为“架构”是一个简单的实体，能够用一份文档或一张图纸来描述，那么你就错了。确实，架构师在为系统创建架构时经常要做出很多与设计相关的决定，而且会用文档的形式记录下来，但是架构和文档又不是等同的。之所以要以文档的形式保存，主要是为了便于以后对这些设计决定进行审核和修改，并且将其作为构建软件时的约束和指导。

1.1.1　什么是架构

首先，让我们来看看架构的一个定义：

软件架构是对系统的高层视角，或者是对系统的抽象。它关注的是某些对完成这个系统有最大帮助的方面，例如：可用性、稳定性，以及灵活性。同时，架构对如何达到这些目的给出了指导和约束。

如果要用最简单的方式来理解上面的定义，可以这样说：软件架构就是软件系统的一张蓝图。

下面我们用一个比较浅显的类比来说明这个“蓝图”的含义。

相信大家对冶金行业中的“浇筑”或多或少有一些了解。浇筑就是把液体材料倒在一个有着特定形状的沙模中，然后等液体冷却之后得到想要的产品或特定形状材料的过程。在这个过程中，盛放液体的沙模引导着液体慢慢成形，最后得到我们预想的结果。

其中，有一点要注意的是，沙模的特性（比如形状和大小）和倒入其中的液体是没有任何联系的，换句话说：沙模和液体是完全分离的，但是沙模又必须和液体在一起才能生产出所要的产品。

架构就好比上面例子中的沙模，软件项目就好比用于浇筑的液体。正如浇筑一样，架构引导着项目，最后得到了我们想要的结果。同时，我们也可以得出：软件的架构和实现这个系统的代码是没有很严格的关系的，这也就是我们常说的架构是平台无关的。架构可确保开发的过程在一定的限制或规则下进行。

对于沙模来说，如果没有浇筑液体的存在，它基本就没有任何作用；架构也是立足于项目的特定需求而设计的！

1.1.2　架构师的职责

既然已经了解架构的含义，那我们再来看看负责创建架构的责任人：架构师。架构师这个角色在任何软件开发项目中都是最有挑战性的。

1. 架构师的领导与决策能力

首先，架构师是一位技术领导，这意味着架构师除了拥有专门的技能外，还必须拥有领导能力，而且此领导能力要能体现在组织中的职位上。

从职位上来讲，架构师是项目中的技术领导，应该拥有进行技术决策的权威。不过，很多时候架构师和项目经理的职责很容易让人混淆，下面用电影行业的职位来打一个比方，帮助大家了解他们的不同：若项目经理是制片人（确保事情完成），那架构师就是导演（确保事情正确地完成）。架构师和项目经理代表了这个项目的公共角色，对于项目外的关注人员来说，他们是主要的联系点。架构师尤其应该是创建一个架构及给组织带来价值的投资倡导者。

在决策方面，架构师要综合考虑，果断下决定。例如，在某些情况不清楚或没有充足的时间探究所有的可能性及有交付压力的情况下，如果架构师不能进行决策，那是不行的。而且这样的环境会很常见，架构师要接受这个现实而不是设法改变它。

有些时候，架构师会在决策时咨询其他人并营造其他人共同参与决策的环境，但是进行适当的决策仍然是架构师的职责，即使有时候这些决策并不总是正确的（当然，是事后才发现这些决策不正确的）。因此，架构师必须是“厚脸皮的”，因为他们可能必须纠正他们的决策并原路返回。

没有决策能力的架构师会使项目慢慢被破坏。项目团队会对架构师失去信心，项目经理

将会担心，因为这些等待架构师决策的事项没有进展。更加危险的是：如果架构师没有制定关于架构的决策并将其编写成文档，团队成员会开始制定他们自己的（可能是不正确的）决策。

2. 架构师的角色可能由一个团队来履行

角色和人之间是存在差异的。一个人可能会履行很多个角色，一个角色也可能会由许多人来履行。由于架构师需要非常广泛的技能，所以，架构师这个角色也可能会由多个人来履行，这时，架构团队中的每个人都可以充分运用他自己的经验来履行此角色。特别是在理解业务领域和各方面技术所必需的技能时，往往需要多人合作才能达到相关要求。有一点很重要，就是最终的团队必须平衡。

如果架构师角色由一个团队来履行，拥有一个首席架构师就非常重要了。首席架构师是架构团队的协调人，经常应该有先见之明。如果没有这个协调人，要让架构团队的成员创造出内聚的架构或做出决策是很困难的。

对于一个不熟悉架构概念的团队来说，为了达成共同的目的，可创建并颁布一个团队规章，而且建议这么做。

优秀的架构师知道他们的优势和弱势。最优秀的架构通常是由一个团队而不是个人创建的，这都是因为“人多力量大”，人多则见识更广和更深。

3. 架构师要理解软件开发流程

大部分架构师都曾经做过开发人员（几乎绝大部分架构师都是从开发人员走过来的），同时，架构师应该了解软件开发流程，因为这个流程能确保团队的所有成员协调工作。

这种协调性可以通过定义涉及的角色、从事的任务、创建的工作产品、不同角色之间的移交点来获得。因为在日常工作中架构师会影响许多团队成员，所以理解团队成员的角色和职责，理解他们正在生产和使用的东西对于架构师来说很重要。实际上，团队成员也非常希望架构师能够指导他们的工作。

4. 架构师要掌握的技术与设计知识

架构设计会涉及技术知识，所以，一个架构师应该拥有一定程度的技术技能。不过，架构师不必是一个技术专家，他要关注的是技术相关的重要因素，而不是细节（其实很多时候，架构师也是技术专家，而且对细节理解得非常深入）。架构师需要理解像 Java EE 或 .NET 这样的平台上可用的关键框架，但是不必理解这些平台程序编程接口（API）的细节。

架构师必须与项目中的开发人员打交道，只有当架构师承认开发人员的工作价值时，在架构师和开发人员之间所进行的沟通才是有效的。这同时也进一步说明了架构师应该具有一定的编程技能，即使他们在项目中不必编写代码，也必须跟上技术更新的脚步。

架构师应该有组织地参与开发，并且尽可能地参与代码的编写。如果架构师参与实现，开发团队会从架构师那儿获得见识。架构师还可以通过查看他们决策和设计的第一手结果来进行学习，从而对开发流程给出反馈。

大部分成功的软件架构师都曾经是核心的编程人员。从某种程度上来说，他们就是通过这段经历了解到业务的某些情况的。如果没有这些知识，要实现架构上的重要元素（如源代码的