大师签名系列

MASTER SIGNATURE SERIES 大师签名系列

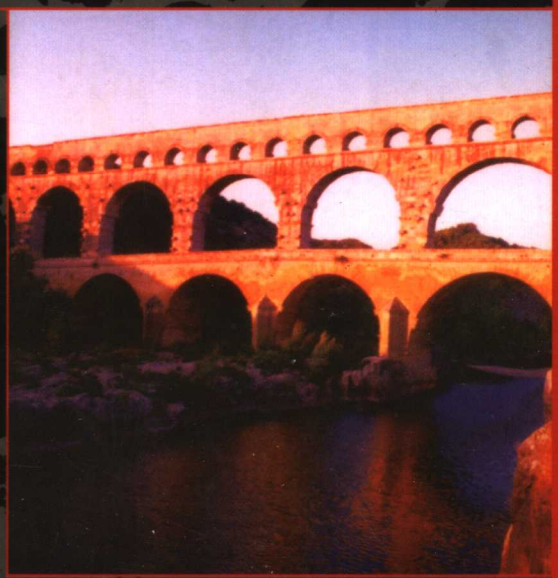
Beyond Software Architecture

Creating and Sustaining Winning Solutions

超越软件架构

创建和维护优秀解决方案

[美] Luke Hohmann 著
蓝莉 曾永和 译
龚波 王高翔 李志 审校



- 第一本完美结合软件技术和实际业务的图书
- 致力于创建成功软件和公司者必读

PEARSON Addison Wesley

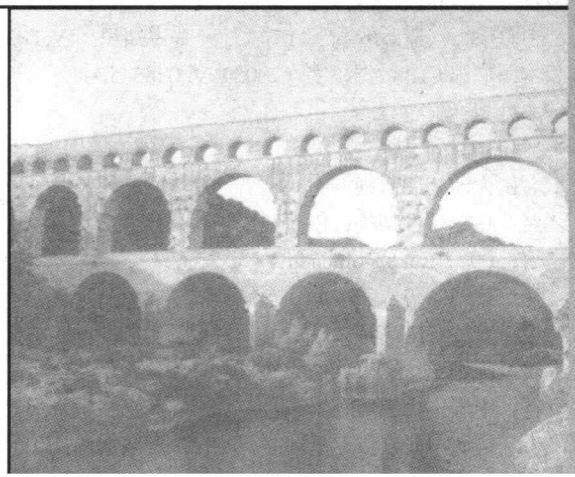
中国电力出版社
www.infopower.com.cn

大 师 签 名 系 列

Beyond Software Architecture
Creating and Sustaining Winning Solutions

超越软件架构
创建和维护优秀解决方案

[美] Luke Hohmann 著
蓝莉 曾永和 译
龚波 王高翔 李志 审校



中国电力出版社
www.infopower.com.cn

Beyond Software Architecture (ISBN 0-201-77594-8)
Luke Hohmann

北京市版权局著作权合同登记号 图字：01-2005-0827 号

图书在版编目（CIP）数据
超越软件架构：创建和维护优秀解决方案 /（美）霍曼著；蓝莉，曾永和译.
—北京：中国电力出版社，2005
（大师签名系列）
ISBN 7-5083-1550-2
I.超... II.①霍...②蓝...③曾... III.软件—基本知识 IV.TP31
中国版本图书馆 CIP 数据核字（2005）第 028345 号

丛 书 名：大师签名系列
书　　名：超越软件架构：创建和维护优秀解决方案
编　　著：（美）Luke Hohmann
翻　　译：蓝莉　曾永和
审　　校：龚波　王高翔　李志
责任编辑：陈维宁
出版发行：中国电力出版社
　　地址：北京市三里河路6号　　邮政编码：100044
　　电话：（010）88515918　　传　　真：（010）88518169
印　　刷：北京丰源印刷厂
开本尺寸：185×233　　印　　张：15.5　　字　　数：345 千字
书　　号：ISBN 7-5083-1550-2
版　　次：2005 年 5 月北京第 1 版　　2005 年 5 月第 1 次印刷
定　　价：32.00 元

对《超越软件架构》的赞扬

Luke Hohmann 是从终端用户的角度来审视软件开发的少数软件技术员之一。他坚信花一个小时与终端用户一起进行研究，其成果不次于用数小时来选择软件架构，或是花上好几天来琢磨假想用户的需求。大多数软件开发的文章主要讨论设计和开发健壮软件的方法。而 Luke 的新作——《超越软件架构》，则是讨论如何在软件生命周期内与用户充分交互以提供有价值的商业解决方案。Luke 关心软件的商业价值，正因为如此，他把软件应用和软件所承载的商业功能更加紧密地结合在一起。

—Bruce Bourbon
General 合作伙伴，Telos Venture 合作伙伴

可以把阅读 Dilbert 连环漫画的人分为两种：一种人会在看下一页前，花几分钟来感叹书中如何真实地反映了自己公司的生活，而另一种人则认为 Dilbert 是在提醒人们高科技公司能够并且应该比 Dilbert 描述得更好。前一种人应该继续看漫画，这本书是为后一种人撰写的。

—Tony Navarrete
Diamondhead Ventures 副总裁

Luke 对软件开发提出了一个已证实的方法论。在《超越软件架构》中，Luke 提供了实用和已验证的技术，开发主管可以应用这种技术来提高软件组织的生产力。

—G. Bradford Solso
Taviz Technology CEO

《超越软件架构》是我读到的第一本既包括权威人士的商业观点又包括软件架构的技术观点的书。它是一本能够同时打动市场人员和软件经理的好书。

—Damon Schechter
LOC Global CEO
《Delivering the Goods》的作者

有些书讨论技术架构，还有一些书讨论产品市场。但很少有书讲述这个理念：产品架构和市场信息必须整合才能开发出世界级产品。《超越软件架构》解释了如何向市场提供可获利的高质量产品，这在技术和市场之间提供了一个有价值的桥梁。

—Jim Highsmith
Cutter Consortium 董事
《Adaptive Software Development》的作者

产品开发经理、市场经理、架构师和技术带头人都应该阅读这本书，就如何定义和使用贯穿项目生命周期和产品生命期的软件架构方面，得到具有实践意义的观点。

—Johanna Rothman
Rothman Consulting Group, Inc.

在这本书中，Luke Hohmann 抓住了产品创建的本质。他巧妙地讨论了产品创建时市场和工程人员所扮演的角色，并通过两者的结合来建立理解和实现成功产品的良好基础。

—Lee Sigler
360 Market View, Inc. 负责人

终于有一本书讨论一些经常被忽略而又非常重要的运作性问题了，如许可证、部署、安装、配置和支持。对于软件开发者或软件用户来说，《超越软件架构》属于那种“在哈佛商学院学不到”的书。

—Mary Poppendieck
敏捷联盟的管理经理
Poppendieck LLC 总裁

Luke Hohmann 向软件架构师提出了一个充满热情的、清晰的忠告：软件架构不再仅仅是一种技术！技术架构已经有了一个富有深意的业务衍生物，忽略产品的便携性、可用性、可配置性、可升级性以及发布管理、安全和其他架构选择等业务衍生物，不仅会导致项目的失败，还可能会在客户不满意时被起诉。《超越软件架构》是成功软件产品经理的必读书。

—Ed Yourdon
众多软件开发相关图书和论文的作者

《超越软件架构》不仅仅是写给软件工程专业人员的！它同时也为那些要对公司的软件开发做出深层决策的经理主管人员和产品经理们提供了必要的知识背景。这本书也是一个使管理和工程保持一致的有效工具，因为它避免了在讨论业务和与客户相关的问题时，过深地陷入那些实现细节。

—David Chaiken
Systems Architecture
AgileTV Corporation 副总裁

产品市场影响产品架构。这点其实并不奇怪，然而大多数关于软件架构的书籍在这方面做得不够。这可能是因为我们缺少区分架构的技术方面和市场方面的语言。《超越软件架构》恰恰提供了这种语言来描绘这个重要的区别，并且为全面的架构成功提供策略。

—Dave W.Smith

《超越软件架构》，正如书名中所暗示的，成功地定位了开发整体解决方案中最容易忽略的因素。Hohmann 全面深入地讨论了这些因素。

—Craig Priess
Resonant Software 产品管理经理

浏览我的技术图书馆，显然大多数书籍都是垃圾，它们都是些技术创新或改变的牺牲品。只有一小部分的书籍是可以被保留或继续使用的。Luke Hohmann 的这本新书《超越软件架构》就是其中一本，它拓宽了可选择性，并且填补了重要的空缺。这是第一本可以展现软件创建全部步骤的书籍。通过融合和连接重要的业务和市场开发来探究技术因素，并提升和展现技术和市场过程是怎样结合来创建优秀的解决方案的。这个话题超越了纯粹编程人员、设计人员和其他技术人员的范畴。对于市场专业人员而言，这本书展现了他们的决定和策略是怎样影响到技术决策的。对于客户来讲，这本书为他们和软件生产商的合作搭建了最好的途径。对于软件公司而言，这本书为创造更好的投资提供了方案。同时，这本书运用了简单易懂的语言，把详细的内容转化成容易记住和应用的信息。这本书中阐述了永恒的话题，它会持续很长的时间。

—Clay Miller

我热诚推荐这本书。作为软件公司的执行总裁和创建者，我和很多软件工程师一起工作，并且工程部的高级经理向我汇报。Luke 是他们中最优秀的一名成员。他不仅是一名出色的工程师，而且能够迅速领悟怎样能够驱动那些优秀代码和架构决定的战略业务。我认为任何一名建造软件系统的人员都应该读一下《超越软件架构》。

—Kevin Rivette
BCG Consulting 执行顾问，《Rembrandts In The Attic》的作者

有这样的一些人，也许你曾经遇到他们、与之一起工作、在活动中听过他们的演讲，或者读到过关于某些人的文章：他们因掌握了最先进的软件模型、技术，或者开发过程和工具的理论知识，而期望得到尊重。那又能怎样？假设，你因为这样的知识而得到了荣誉；假设这种知识只是你所工作过的团队中最小的成就。然后又能怎样呢？什么才能把对项目和公司成功而言真正的贡献者从当时的中等水平的书本专家中区分开呢？Luke Hohmann 在书中给出了答案。安装和升级、软件的可配置性和定制性、同其他软件的集成性、可用性、日志功能、部门间过程和发布管理、商业模型、许可证和部署选择，正是熟悉如上业务的人使得团队可以开发更大的软件，而且促成了真正意义上的开发，这比泛泛而谈的 UML 图强多了。Luke 知道这些，是因为他曾经在这条战线上缔造业务上的成功。他把大部分时间用来干实事，而不只是单纯写作和谈论。现在他精练了他的思想，并且贡献出有益的经验同我们分享。就好像我一样，你可能正在频频点头并且在读这本书的时候不时地标注着下划线。Luke 的观察结果会引起你的共鸣。你的公司和全体软件开发的专家们，都会从这本书的字里行间和书中所描述的思想上获益。所以我相信你将会像我一样把这本书推荐给其他人。

—Randy Stafford
总架构师 IQNavigator Inc.

对于众多、细致的技术决策是如何塑造经济和战略前景的，Hohmann 提出了独一无二的观点。不同行业的公司决策者都会发现书中见解有着巨大的价值。

—Martha Amram
策略顾问和咨询师
《Value Sweep》的作者，《Real Options》的合著者

Luke 应用他的宝贵经验，消除了市场和工程开发各自为政的现象。这是一本高级工程师、软件架构师和产品经理必读的书籍。通过采纳这本书的实用建议，他们定能够集中精力击败竞争对手。

—Heinrich Gantenbein

我曾经担任 Luke Hohmann 的 QA（Quality Assurance，质量保证）经理。我可以告诉你这这个家伙熟悉先进的软件开发。我们曾经是同一战壕里的战友，为解决技术问题和实施复杂的业务决策苦干。如果要结合技术和业务两个方面，编写一本关于软件架构的书，那就非他莫属了。这本书很成功，它在这两个不同的世界（业务和技术）间搭建了桥梁。这是对于软件工程的产品经理和主管的“经典”手册。

—James Bach
Satisfice Inc.的创始人

有很多次，我的公司不得不解决一些本来可以避免的问题。如果涉及问题的诸公司在签订合同之前明确了他们的业务和许可模式，这类问题就不会发生。在《超越软件架构》这本书中，Luke Hohmann 很清楚地解释了怎样避免有关业务和许可模式的不必要的和需要付出昂贵代价的争端。这是高级产品经理和技术主管必读的读物。

—Rob Sterne
Sterne,Kessler,Goldstein and Fox International Authority in Intellectual Property 创始人

Luke Hohmann 的《超越软件架构》阐述了许多本质的、可贵的见解，作者从个人的角度出发（个身兼架构师和商业领队两职的个人）向读者阐明架构不仅在于描述各个层面；而且同时涉及通过理解并彻底融汇市场、业务和技术的各方动力，以创建优秀的解决方案。

—Craig Larman
《Applying UML and Patterns: An Introduction to OOA&D and the Rational Unified Process》作者

通过评估业务驱动和软件开发间的互动,《超越软件架构》提供了创建成功解决方案的新观点。这是一本能快速获得市场解决方案的使用指南：告诉读者如何发现缺陷并提供实用的应变之策来解决问题。本书真实、生动地介绍了生命周期的各个阶段，其中包括项目团队中明确的角色和责任。从何时应该放弃 ghost 软件、寻求其他人的帮助，或简单地权衡一个已验证模型,《超越软件架构》探究了可选择的方案并且定义了关键的决策点。它是那些正在寻

求对挑战和成功软件开发本质的理解，并期望第一时间获取成功解决方案的软件架构师和产品经理的必读书籍。

—Mark Welke
Hewlett Packard 高可用性市场经理

创建一个出色的应用软件需要考虑得面面俱到。杰出的软件不可能单单出自某个杰出的想法，或是某个出色的架构，又或是某次对客户需求的准确评估。事实上，它们受到多种因素的影响，其中包括：市场、技术、心理学、支持、经济、法律因素，等等。大多数书籍都定位在软件应用开发层面上，而 Luke Hohmann 的《超越软件架构》则涉及更广阔的话题，这些都是在开发应用程序过程中不可或缺的部分，也是许多书中没有涵盖的。我推荐每一位软件应用程序的开发者把这本书当成一个指引，以使得在创建软件的过程中每一个细节都可以被考虑到。没有一本书可以包含每一件事，但这本书却进行了一次值得称赞的尝试。

—Jim Gay
One Jump Consulting

一个成功的软件产品技术架构必须和市场现实情况相吻合。当目标很明确的时候，市场人员和技术专家总是发现他们置身于两个不同的战场，无法调和的态度和期待如同鸿沟一样将他们隔开。如果想要填平这条鸿沟而获得成功的产品，那么市场人员和技术人员都应该好好读读这本书。

—Dave Quick
Integrated Solutions Development Group,Microsoft,Inc.架构师

作为一名已经成为产品团队领导的技术专家，这本书提供了许多我以前学习、尝试和犯错误得来的经验教训。对于一些仍然致力于跨越软件开发和执行管理的人员来讲，这本书会令人耳目一新，并让读者有茅塞顿开之感。当我继续与团队一同工作以管理和开发新产品时，我肯定会参考这本书并把它作为一个教学资源，它能使我开拓视野，避免带来代价昂贵的缺陷。任何一个管理产品开发或市场营销、在管理上有抱负或对当前的管理存在不满的人都应当毫不迟疑地阅读这本书。

—Todd Girvin
Optiview Inc.总裁

Martin Fowler 写的序言

架构在软件行业里变得非常含糊，很难给出明确的定义。我把它作为一个非常主观的用语——是当人们描述他们的软件架构时，介绍系统的重要部分，这些部分如何互相磨合，以及在设计系统时如何做出重要决定的一个主观用语。架构也被视为一种技术问题，这意味着需要做出的重要决定都是技术性的决定。

过去几年，在我跟 Luke 的谈话中，得到这样一个有趣的事实：就是他所讨论的架构问题，往往是大部分关于架构的讨论中被忽略掉——而实际上是很重要的问题。这些问题包括：系统的市场问题、许可条款、商标、部署、记账。所有这些问题都具有技术和商业上的双重含义。高级技术人员需要思考这些问题，否则一个技术良好的系统会因为缺乏正确的商业决策而失败。

对于大多数把软件卖给其他人的销售商来说这些诸多问题构成了威胁。即便你是一个内部信息系统的架构师，这些问题的存在也会导致你的失败。与供应商签署的许可协议可能造成你布署软件的成本的巨大差异，如果你的业务部门决定引入 Charge-back 模式，那么记账就会尤为重要，同时商标也会影响公司业务的发展。

Luke 从软件开发的技术和商业两方面来编写此书。我发现这很吸引人，因为它引领 Luke 考虑人们经常不会谈及的问题。Luke 向读者展示的总是那些你认为不需要担心但却将你伤害得最深的问题，并给出了处理这些问题的建议。因此，这是一本在软件设计的技术方面非常值得称赞的书。

Martin Fowler
丛书编辑

Guy Kawasaki 写的序言

首先，你必须明白软件编写是门艺术，而不单单是一门科学。我为何要为这样一本集中在具体细节而不是创新技术的书籍写序呢？

不要高估艺术和软件的创造过程。静心地坐在一把舒适的椅子上，让软件的灵感穿越心灵，成千上万优雅的代码行毫不费力地从头脑中溢出。如果你创建过软件，那么你知道这是不现实的。

艺术创作很难，调试（debug）就更难。这是一本增加你和你的团队艺术修养的书籍。它是关于原则、团队合作、汗水和灵感的作品。我期待你读完本书之后可以创造出改变这个世界的艺术——那就是软件。

Guy Kawasaki
Garage Technology Ventures 的 CEO

前　言

很多优秀的书籍都谈到软件架构，这些书主要集中在对软件架构进行定义、分类和描述，同时也讲述如何选择架构，并提供指导。从长期来看，它们都很有参考价值。然而，这些书虽然可以帮助建立成功的架构，但它们却不能帮你建立优秀的解决方案。事实上，除了子系统和接口，诸如前端控制器和管道—过滤器等架构模式，以及建立第三范式关系型数据库之外，我们还应该关注软件架构以外的一些领域，并理解和执行那些必须解决的商业问题，这样才能建立优秀的解决方案。

商业问题的一个例子是技术支持。客户在使用你的软件时存在问题，这是不可避免的。你很久以前在某些方面的设计选择，如日志文件的设计、系统与其他系统的集成方式、系统如何配置，或者怎样对系统进行升级等等，这些都将决定解决问题的方式和程度。本书通过讨论广泛的商业问题以及这些问题和架构选择之间存在的关系，来帮助你了解软件架构之外的领域，从而创建优秀的解决方案。

本书讲述的观点都是我从使用各种系统的经验中总结出来的，这些系统包括成本低于50美元的单用户程序、用于学术研究的软件系统、诊断和解决内部开发系统问题的工具，以及耗资百万美元的分布式企业级平台。在这些系统开发过程中，我扮演了不同的角色。我曾经是一个独立的贡献者、一个直接管理者、一个公司管理部门的高级成员。大多数时间里，我参与或领导工程、产品营销和管理、质量保证、技术出版，以及一线和二线技术支持团队。我也曾经管理过跨城市、甚至跨洲的团队或项目。

向某类人提供某种价值是软件开发的宗旨。例如，研究型软件满足需要了解某种现象的研究人员的需求。对于那些需要处理从客户到供应链管理的企业级应用软件，它们应该设计为可以满足用户和业务的需要，并能带来持续性收益。其他像游戏软件、个人通讯录管理、为图形设计工具设计的库存管理系统等类型的软件也与此类似。

本书定义和讨论的各种话题适用于每个类型的软件。书中展示和讨论了那些经常发生于企业级应用软件（我职业生涯的大部分时间致力于这种软件）中的议题。虽然很难准确定义，但典型的企业级软件总是具有一个或多个以下特性：

- 被设计来支持一个部门或较大组织的业务需求；
- 自己创建或使用许可都相当昂贵（在5万美元到5百万美元之间，甚至更多）；
- 部署和操作需求复杂；
- 可独立操作，但与其他企业级应用程序集成能更好地满足业务需求。

本书同样适用于非企业级应用。创建可持续的软件解决方案（以多个版本的形式在很长

一段时间内满足客户需求）是一种具有挑战性的、愉快的、有益的努力，而这些并不仅仅局限于企业级应用！

虽然我总是提及软件架构和讨论技术性问题，但重点不在于设计架构的图表或文档，我也不会讨论有关建立稳固的、基于 Web 的分布式组件系统的深层设计原理。正如我前面所讲，许多书籍都涵盖了这些话题——事实上，这样的书籍太多了。它们所带来的副作用就是使得很多人将精力过于集中在技术细节上，而忽视了自己试图提供的商业价值。

本书没有集中讨论纯技术，而是把重点放在更广泛的领域，从而为开发团队提供实用、具体的选择，以使他们创建和维护优秀的解决方案。我发现，将重点集中在这些实用的方面，例如，为解决方案进行版本定义或商标元素的整合，可以有效地减少在开发人员和业务人员、市场人员之间存在的人为障碍。

这些障碍能够阻止技术团队和业务团队建立成功的解决方案。当工程师仅仅具有技术见解而不考虑业务问题，或者市场人员总是提出“给我这个特性”但没有考虑这些特性的底层技术时，都会给解决方案带来副面影响。实际上，当他们中的任何一方固守于自己的领域而没有考虑到它的影响时，创建和维护成功解决方案的可能性都会降低。

最大的麻烦在于有些人支持这样的观点，就是把技术问题从业务问题上分割开，或者把业务问题从技术问题上分割开。这样做最好的情况是证明了“这是错误的”，最坏的结果是导致一场灾难。开发人员通常被要求忍受极端的设计所带来的困难，例如为了降低系统的整体成本而要求占用尽可小的内存。投资者也被说服相信一个或更多技术性突破能够为公司提供成功所需的竞争优势。但是如果新的技术突破能带来商业模型的突破，投资者会更加乐意投资。

管理技术和业务之间的相互关系是本书的主线。仅仅运用技术或只关心你感兴趣的技术，即便它是一个优秀的系统，但最终还是会因为没有人使用它而消亡。反之，只关心业务，你能够有一个激动人心的解决方案，甚至可能得到资金，但却不可能提供可持续的价值。只有兼顾二者，才能获得成功的解决方案。尽管创建新的技术或优秀的系统是一件有意思的事情，设计灵活的、新的软件应用程序或者商业模型也是振奋人心的，但相对于建立和维护优秀的解决方案所带来的满足感，上面两方面就显得黯然失色了。

鸣谢

感谢所有帮助我撰写这本书的人。特别感谢 Don Olsen、Haim Kilov、Rebecca Wirfs-Brock、Myron Ahn、Rob Purser、Ron Lunde、Scott Ambler 和 Dave Smith 细致地审查本书内容。还有 Steve Sweeting、Craig Larman、Todd Girvin、Erik Petersen、Sandra Carrico、Adam Jackson、Tony Navarette、Chris Reavis、Elisabeth Hendrickson、James Bach 和 Alan Shalloway，他们都严格审查了部分章节。他们中的一些人要求特别苛刻，而本书正是因为他们的严格要求而变得更为出色。

特别感谢 Bob Glass，他和我共同缔造了本书的题目。他的一个电话比很多封电子邮件更

加有效。

感谢 Ron，和他一起撰写的那段时光特别有趣！

感谢 Steve Dodds、Lee Sigler 和许多在写书期间提供灵感和友谊的学生和同事。

特别感谢 Paul Becker——我的好朋友兼最初出版商，他很有耐心地等待我完成本书。Paul，在我完成第一本著作之后，已经有几年没有写书了。感谢你愿意一直等待着我，直到我觉得有新的内容可写。

我非常感谢 Addison-Wesley 生产和市场部门人员的专业精神和努力工作。感谢他们帮助把原始的手稿变成一本书。特别感谢 Mary O'Brien、Elizabeth Ryan、Marilyn Rash、Chris Guzikowski 和 Dianne Wood.。

我肯定忘了提及一个或多个帮助过我完成本书的人，请相信这并非出自我本意。我相信本书还有很多需要改进的地方。对于遗漏的地方和改进的意见，请联系我，我会尽我所能完善本书。

Luke Hohmann
luke@lukehohmann.com

目　录