The Pragmatic Programmers

架构师修炼之道

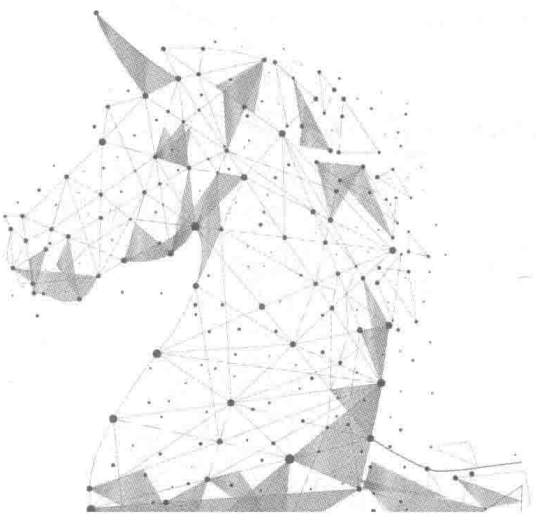
Design It!
From Programmer to Software Architect

[美] Michael Keeling 著
马永辉 顾 昕 译

華中科技大學出版社
http://www.hustp.com

架构师修炼之道

Design It! From Programmer to Software Architect



[美] Michael Keeling 著

马永辉 顾 昕 译

华中科技大学出版社
中国 · 武汉

图书在版编目(CIP)数据

架构师修炼之道 /(美) 迈克尔·基林(Michael Keeling)著 ;马永辉, 顾昕译. -- 武汉 : 华中科技大学出版社, 2019.8
ISBN 978-7-5680-5270-2

Ⅰ. ①架… Ⅱ. ①迈… ②马… ③顾… Ⅲ. ①软件设计 Ⅳ. ①TP311.5

中国版本图书馆CIP数据核字(2019)第154769号

湖北省版权局著作权合同登记 图字:17-2019-145号

书　　名 架构师修炼之道
　　Jiagoushi Xiulian zhi Dao
作　　者 [美] Michael Keeling
译　　者 马永辉 顾昕

策划编辑 徐定翔
责任编辑 徐定翔
责任监印 徐　露

出版发行 华中科技大学出版社（中国·武汉）
　　武汉市东湖新技术开发区华工科技园（邮编 430223 电话027-81321913）
录　　排 武汉东橙品牌策划设计有限公司
印　　刷 武汉华工鑫宏印务有限公司
开　　本 787mm × 960mm 1/16
印　　张 19.5
字　　数 370千字
版　　次 2019年8月第1版第1次印刷
定　　价 99.90元

本书若有印装质量问题, 请向出版社营销中心调换
全国免费服务热线400-6679-118竭诚为您服务

译 序

我与本书应该算是很有缘。我在浙江大学的学长苏杰是七印部落的发起人之一。我们都是产品经理，但我们走了不一样的道路。他毕业后加入阿里巴巴，然后写了《人人都是产品经理》，现在又开办了良仓孵化器。我则加入 Vobile 公司创业，从 0 到 1 做产品。后来我到了美国硅谷，从产品管理晋级到管理整个项目团队。这期间我放弃了去淘宝网工作的机会，与苏杰“失之交臂”。前几年我出了一本文学自选集，请苏杰写了寄语。十二年产品之路，造化各异，现在又有缘在七印部落汇合。

最早看到要翻译这本书时，我工作较忙，没想过参加。过了一阵子，看到苏杰发消息说还没找到合适的译者。我这才发现我和这本书挺有缘，我当时正在卡内基梅隆大学（CMU）读研究生，而本书的作者 Michael 和写序的 George 都是 CMU 的校友。这本书也跟我在学校学的课程（“为产品经理讲述架构原则”）有关联，于是我联系徐定翔编辑表达了想翻译的意愿。

我所在的公司很早就通过了能力成熟度模型集成（CMMI）认证，这是一套流程化的东西。CMMI 一般以瀑布模式开展，流程清晰、产出明确、变更可管理、风险可控。我们能够在早期成功推出面向大型企业的软件服务，就是因为采用了 CMMI。后来我们又引入了 Scrum，希望能做到小步快跑、快速迭代。刚采用 Scrum 时，我们确实遇到了许多问题，其中之一就是 George 在本书序里提到的：如何在快速迭代的敏捷开发中开展传统的架构设计。如果我们能早点看到这本书，也许可以少走一些弯路。

转眼间，我在软件行业工作已经多年。早几年，我也有将自己的工作经验写成书的想法。后来看了一些书，我觉得自己水平还不够，如果能翻译一些好作

品，比自己写书更有意义。我希望可以为这个行业、这个时代做一点贡献，哪怕很微小。

翻译也是一种修炼。我本以为翻译会占用很多时间，给工作和生活带来压力。不过情况实际并没那么糟糕。翻译过程中与作者 Michael 的沟通让我受益匪浅，对工作也有很大的帮助。理解了书上的“道”之后，我往往茅塞顿开。

想当年刚工作时，我作为产品经理常常跟架构师争辩设计优劣和职责划分。按照 Michael 的说法，产品经理更关注功能需求，而架构师还要把握非功能性需求。当然，Michael 是拓宽了架构师的工作范围。同样，优秀的产品经理也应当拓宽自己的工作范围。只有这样，我们才能一起开发出优秀的软件。

我与顾昕、徐编辑合作，像架构师那样对待这个翻译项目。我们在项目启动阶段就商定了翻译规范，选用了协作工具。我们还商量把整本书的翻译稿合并在一起，这样做虽然打开文档有点慢，但带来的好处是易于全文搜索和替换。

苏杰说人人都是产品经理，我理解他是说每个人都应该具备产品经理的思维。按照这个思路，我们也可以说人人都是架构师。掌握产品经理的思维，相对来说比较容易，毕竟普通人都是产品用户，都有对产品功能“指手画脚”的本事。而要掌握架构师思维，就稍微难一点了，所以从事 IT 行业的人都应该读一读这本书。这本书不仅是写给程序员和架构师看的，也适合从事产品管理、项目管理、数据分析、测试、运维工作的人读。

我读的第一本七印部落翻译的书是《启示录：打造用户喜爱的产品》，现在我还时不时拿出来读一读，常看常新。希望这本《架构师修炼之道》也一样，能够帮助大家尽快“修炼”成出色的架构师。

马永辉

2018 年 9 月于卡内基梅隆大学硅谷校区

序

Foreword

拿到这本书的终稿时，我惊奇地发现书中竟然没有提及敏捷开发。Michael 和我多年来一直在谈论这本书，我想我很清楚他的写作内容及意图：传统的软件架构理念在敏捷开发流程中一直很难应用，而 Michael 想到了解决办法。那么，“敏捷”为什么没有出现在书的每一页里呢？

Michael 是当代的普罗米修斯，他对技术着迷，决心将其驯服，造福众人。他是敏捷开发的真正信徒，也是软件架构方面的专家。白天，他是付诸行动的敏捷开发团队的领导者；晚上，他又是卡内基梅隆大学里学习软件架构的学生的指导者。他是我所知道的唯一拥有这样多重身份的人。通过参与卡内基梅隆大学组织的 SATURN 软件架构会议，我对他了解颇多，他将敏捷社区的理念和思想领袖推荐到架构师社区。他一直在寻找两全其美的方法，将敏捷开发与架构设计完美融合，而不是像油和水那样泾渭分明。

很多人曾尝试调和这种矛盾，但收效甚微。曾经有人想把敏捷开发塞进瀑布模型的实施阶段。还有人坚持认为应该由“一把手”架构师做重要的决定。这些人几乎都是囿于理论泛泛而谈，却说不出到底有哪些成功的案例。他们写的东西通常是站在某一阵营的角度，试图照搬另一个阵营的方法。

本书的与众不同之处在于它更好地融合了敏捷开发和架构设计，这也是“敏捷”这个词没有出现在每一页上的原因。Michael 凭着对敏捷开发及其价值的深入理解来谈架构设计。Michael 亲自提出和改进了许多方法，但尤其令我感到兴奋的是，他也从过去几年的会议中汲取了最好的想法和技术，这些还从未有人写过。比如在本书第三部分，你看不到架构师单枪匹马为团队指明方向——刻诫于碑、铸法于鼎。相反，你将看到如何在以周为单位的迭代中完成各种任务，如何鼓励整个团队参与设计，将设计提升为团队关注的头等大事。而且这些都有实际的应用案例。

思想领袖们推翻了缓慢的、低效的、“官僚主义”的软件开发流程，但他们也告诫我们，敏捷绝不是不规范的、草台班子式编程的“挡箭牌”。前敏捷时代的那些“官僚主义”的开发流程大部分是符合规范的，至少说明了应该在何时开展哪些设计活动。虽然有些团队自称他们遵循敏捷开发流程，但据我看，当下有太多不规范的、草台班子式的开发行为。

现在有了这本书，问题是接下来会发生什么？虽然做预测很难，但我还是相信，我们正在向软件开发的下一平稳状态过渡——融合敏捷性与纪律性。我们的流程将融会贯通，一方面掌握敏捷开发所倡导的快速反馈循环，另一方面掌握架构设计技巧，开发出更出色的软件。

虽然我们还没有达到那种状态，但本书将指引我们朝着那个方向前进。让我们一起携手共创未来。

George Fairbanks

《恰如其分的软件架构》作者

前 言

Welcome!

软件架构是开发优秀软件的基础。虽然出色的架构本身并不足以确保软件成功，但错误的架构几乎注定导致失败。架构非常重要，所有软件开发人员都应该知道如何进行设计。

本书讲解如何设计出色的软件架构。首先声明，本书不是象牙塔里抽象的软件设计教材，也不要指望在书里找到变魔术般解决所有问题的框架或技术。你能学到的是应用基本的设计原则和经验，成长为优秀的程序员、架构师、技术领导者。

此外，设计出色的软件不仅靠设计原则和经验，如何开展设计也同样重要。本书将讲解如何应用设计思维和以人为本的思想与团队进行协作。这种架构设计办法将极大提高设计决策与团队成员的联系。以人为本能让你做出更好的设计决策，从而开发出更出色的软件。

目标读者
Who Should Read This Book?

本书是写给所有曾站在白板前画方框和线条，试图解决棘手问题的人看的。

如果你是软件架构设计新手，本书很适合入门学习。我们将从介绍基础知识开始，由浅入深逐步讲解优秀软件架构师必须掌握的核心技能。

如果你是对架构设计略知一二的程序员，本书将有助于你整理思路。你会读到那些你既陌生又熟悉的概念，填补你自己都未曾意识到的知识空白。读完本书，你将更加深入地理解架构师的工作，以便日后更好地领导他人。

如果你是久经沙场的软件架构师，本书将教你从一个全新的视角来审视如何

领导团队。今天，越来越多的初级程序员希望在软件开发中发挥更大的作用。书中讲解的基础知识将帮助你引导他们全面地参与到设计过程中来。本书阐述的协作设计方法可以让你安全高效地与经验不足的团队成员进行合作。

如何阅读本书
How to Read This Book

本书分为三部分。第一部分与第二部分建议从头至尾通读，第三部分则便于参考和检索。

第一部分介绍软件架构的基础知识和架构师必备的设计思维。

第二部分讲解架构师需要掌握的核心技能和知识。

第三部分讨论一系列实用的架构设计方法。世上没有万能钥匙，每位软件工程师都有自己的一套经验、方法、技术。第三部分将介绍我自己的经验、方法、技术。

第二部分和第三部分的每一章都会讨论一种设计思维模式（以下简称思维模式）。思维模式是我们看待世界的方式，帮助我们在正确的时间关注恰当的细节。总的来说，思维模式可以分为四类：理解、探索、展示、评估。

同行的技巧和建议
Community Tips and Advice

打开这本书，你就加入了一个软件架构师的社区，大家在这里互相帮助，分享建议、技巧、方法。我邀请了几位架构师分享他们认为你应该知道的技巧和建议，这些建议穿插在书中相应的章节里。

这些杰出的贡献者是 Len Bass、BettBollhoefer、Simon Brown、George Fairbanks、Thijmen de Gooijer、Patrick Kua、Ipek Ozkaya。详细信息请参阅本书附录。

案例研究
Case Study

谈论抽象的东西往往容易流于抽象。为了防止这种情况的发生，我引入了一个实际案例：Lionheart 项目。它来自我以往做的真实系统。第 1 章将介绍这个案例。随着本书内容的展开，你将看到这个案例的更多细节。

动手练习
Get Your Hands Dirty Exercises

实践出真知。要成为优秀的软件架构师，不能纸上谈兵，还要参与实践。就像现实世界中的架构设计一样，练习的答案不止一种。练习过程与最终解决方案同样重要。

在线资源
Online Resources

本书主页上有图书的详细信息[1]。你可以在论坛上发帖讨论，或者指出印刷和内容上的错误。讨论区是进行内容交流和分享练习答案的绝佳场所。

欢迎与我同行，让我们开始吧！

[1] https://pragprog.com/book/mkdsa/design-it

目 录

Contents

第一部分 软件架构导论

第二部分 架构设计原理

第三部分 架构师的工具箱

第一部分

软件架构导论

Part I Introducing Software Architecture

首先介绍一些基本概念，包括核心架构原则和基本设计知识。