ThoughtWorks® 中国公司总经理**张松**　**敏捷**教练**王宇**
InfoQ 中国创始人兼 CEO **霍泰稳**

作序推荐

软件开发践行录

ThoughtWorks®中国区文集

ThoughtWorks 中国 著

- ThoughtWorks® 中国的软件精英们多年宝贵实践经验的凝结
- 涵盖过程改进、工程实践、团队建设和体验设计 4 大话题

人民邮电出版社
POSTS & TELECOM PRESS

软件开发践行录

ThoughtWorks®中国区文集

ThoughtWorks 中国 著

人 民 邮 电 出 版 社

北 京

图书在版编目（CIP）数据

软件开发践行录 : ThoughtWorks中国区文集 / ThoughtWorks中国著. -- 北京 : 人民邮电出版社, 2014.8
ISBN 978-7-115-35881-3

Ⅰ. ①软… Ⅱ. ①T… Ⅲ. ①软件开发－文集 Ⅳ. ①TP311.52-53

中国版本图书馆CIP数据核字(2014)第150559号

◆ 著　　ThoughtWorks 中国
责任编辑　陈冀康
责任印制　彭志环　焦志炜

◆ 人民邮电出版社出版发行　　北京市丰台区成寿寺路 11 号
邮编　100164　　电子邮件　315@ptpress.com.cn
网址　http://www.ptpress.com.cn
北京艺辉印刷有限公司印刷

◆ 开本：787×1092　1/16
印张：18.5
字数：351 千字　　2014 年 8 月第 1 版
印数：1-3 000 册　　2014 年 8 月北京第 1 次印刷

定价：49.00 元

读者服务热线：(010)81055410　印装质量热线：(010)81055316
反盗版热线：(010)81055315

内容提要

ThoughtWorks 是一家全球软件设计与定制领袖企业。ThoughtWorks 首席科学家 Martin Fowler 先生是当今世界软件开发领域最具影响力的几位大师之一。在中国，ThoughtWorks 已经成立了北京、西安、成都、上海、武汉五个分公司，聚集了大量高水准的软件专业人才，为各类企业、政府部门和非营利组织提供全球品质的专业服务。

本书是 ThoughtWorks 中国区软件技术人员的文章合集，挑选和收录了包括胡凯、熊节、徐昊、郑晔、张逸等人的 33 篇精彩文章，涵盖了过程改进、工程实践、团队建设和体验设计 4 大领域。这些曾经以各种形式在网络、报刊、社区发表或分享，有些文章还曾引起积极的反馈和热烈的反响。本书可以说是一群极有天分的软件精英的思想和观点的汇聚，是他们多年的宝贵实践经验的凝结。

本书适合软件行业的各类从业人员阅读和参考。对于想要对软件构建过程进行持续思考和改进的人，本书具有启迪思想和激发灵感的作用。软件开发人员可以通过阅读本书，吸收作者们在工程实践中的宝贵经验；项目经理则可以窥得团队建设和管理的奥秘。

序 一

在 ThoughtWorks 中国公司成立十年之际，人民邮电出版社汇编了公司中国区员工最近几年发表的文章，推出了这本文集，算是 ThoughtWorks 中国区厚积薄发之作。

在中国的软件社区，ThoughtWorks 一直顶着敏捷开发方法的领先者形象，并由于给不少知名企业做过咨询而为业界所了解。在跟社区朋友聊天的时候，我发现不少人都不知道其实这家公司其实只有不到 10%的人在做咨询，而其他大多数人都工作在各类软件开发项目上。

ThoughtWorks 服务的客户中有规模巨大的金融、电信、物流企业，也有快节奏的互联网公司，还有资源严重受限的 NGO 公益组织。ThoughtWorks 的团队在产品开发生命周期各个环节，从概念、思路的产生，到用户体验设计，开发实现，再到生产环境的上线，甚至是直到产品、服务的推向市场，都常常承担着关键的职责。客户和产品的多样化，以及全生命周期的参与，使得团队有机会接触不同类型的系统，面对不同类型的约束和挑战，从而获得难得的阅历和视野。

这本书的作者都是我身边的同事，他们撰写文章，活跃在社区行业会议的讲坛上，通过分享各自在实战经历当中积累，致力于为社区和行业产生正面的影响力，并赢得了尊敬。我非常荣幸能与这样一群人一起工作。

本书涉及内容非常广泛，包括从用户体验到开发、测试和运维、从技术到管理等诸多方面的点点滴滴，虽然不见得非常系统和全面，但都是来源于一线。读者既可以选读感兴趣的章节，也可通读全书，一览 ThoughtWorks 团队的实际开发全貌。

《精益软件度量》作者
ThoughtWorks 中国公司总经理
张 松

序　二

很难形容当我看到这一篇篇实践经验时的感受。不光因为这些文章中所描述的经验也有我参与讨论的共鸣，我更是被这不停探索技术卓越的状态所感动。

回想 2001 年我在《程序员》杂志看到那一页描述极限编程的文章，震撼之后我深深相信这就是软件的未来。你可以想象一下连讲 J2EE 的书籍都充满错误的年代，找到关于敏捷软件开发的资料是多么困难的一件事情。在实际工作中，这些书籍更是缺乏最后一公里的力量。为了更加拉近信念与现实的距离，我加入了 ThoughtWorks 中国公司。在交付与咨询之余我们不停思考、不停质疑、不停总结。所以在你面前的这本书充满了最后一公里的力量，虽然有些内容在文字和理论方面略显单薄，但我相信你在阅读本书之后会更有信心地面对日常软件开发过程，也会再次注入更多的软件热情到你的身体之中。

过程改进部分，你可以理解以过程为形式的改进，所以重音应该在“改进”之上。

看板任务管理：看板方法已经成为企业渐进式变革的一套方法论，其核心观点是把管理职责交还给知识工作者本身通过可视化的方法实现渐进式的管理。对于看板理论我非常留意我们说的是 Kanban 还是 kanban（大写 K 哦），可以说文章更多地触及 kanban 落地的一些具体实践，并详细说明背后意图。

实战：持续交付中的业务分析：虽然业务分析的套路现在已经被“精益创业”，“商业模式画布”，“设计思维”等高大上的词语或概念所占据，但如果仅仅回到价值交付而言，看完这篇文章你会有一些基本思路来面对眼前的问题并引发更多的思考。

建设 DevOps 能力，实现业务敏捷：我认为熊节这篇文章是一篇非常好的宣传性文章，整体思路清晰，我看完之后有想动手实践的冲动。如果读者你也和我一样，建议你先拿文中的几个工具开始试验一下，我相信你的收获会更多。

自我成长时考虑精益的方式：熊子川这篇文章是精益画布的一个精简版本，看完之后我在思考我们是否需要商业画布上的这么多的部分，还是用这个精简的版本确定思路之后就着手制作第一个短裤版本的软件扔向市场。我个人感觉很多情况之下第二个选择就已经足够。

运用系统思考，走上改善之路：这篇文章非常切“改进”这个题目，强烈推荐。

分布式开发：敏捷的实践对于本地团队基本上是有一些最佳实践的，但是如何在分布团队也能实现敏捷呢？看看 ThoughtWorks 中国的经验吧。

精益创业和敏捷：两个都是大词，虽然当今的敏捷已经包罗万象，但如果给敏捷一个小一些的帽子并思考与精益之间的关系还是非常有必要的。

我和敏捷团队的五个约定：短短文章包含了很多内容，我个人觉得覃其慧可以用更多的笔墨来描述背后的故事。但这种留白也可以让读者思考自己环境下测试人员的状态与挑战，体会这五个内容背后的故事。

工程实践部分，此部分我认为是本书含金量最高的部分。如果你对其中的某些知识领域不感兴趣，建议你推荐给喜欢相关领域的朋友，相信他们会非常感谢你的推荐的。

团队建设，人始终是敏捷的关键挑战，读者可以在这里看看 ThoughtWorks 如何面对这个挑战的。

建设全功能团队——实践篇：看看胡凯是如何带好新人的。

建设全功能团队：看看胡凯是如何进行人员配置的。

高杠杆敏捷团队中的团队建设实践：看看 ThoughtWorks 的新人是如何快速提高技能的。

全功能团队——数据篇：看看跨职能团队的数据吧，如果你在乎“产能”的话。（虽然我个人非常反对使用这个词）

体验设计，好的开始是成功的一半，这也就不奇怪现在这篇文章放在本书最后一篇的重量。我非常认可纸质原型 Paper Prototyping 而非 Visio 或 Axure 此类原型工具这种观点。我相信这篇文章会给那些 UX 或 UI 人员以更多的弹药来面对敏捷转型的挑战。

羡慕你拥有这么多支持与资源之余，我开始有些担心你是否能承受这一软件经验的饕餮盛宴了。

敏捷教练　王宇

序　三

当 ThoughtWorks 市场部的同学来询问，是否可以将以往其同事发布在 InfoQ 中文站上的内容结集出版时，我的第一反应是 Why not？大好事一桩啊，这不也是在践行 InfoQ 的“促进软件开发领域知识与创新的传播”的理念吗？虽然在沟通的过程中，由于版权使用的原因多了一些小插曲，“好事多磨”，看到排版好的电子文件，还是很欣慰的。

也许不像现在的 BAT 那么光鲜，但是 ThoughtWorks 自进入中国以来，就一直保持着自己的影响力。其中很重要的一点就是 ThoughtWorks 的同学们好像都特别能写和能说，不过也难怪，要不然怎么能做咨询师呢。这也是我一直很敬佩和尊敬 ThoughtWorks 的原因之一，要知道分享知识是很费时间的，写东西和演讲都需要花费大量的时间准备。如果抠门一点的老板会想，将这些时间放在给客户提供咨询上，不是更能产生收入吗？从目前这本书的出现来看，ThoughtWorks 显然没有这样做，而且很可能在背后努力推动大家去分享。有这样的推动力，对技术社区的价值自然是善莫大焉。

看到这本书里很多熟悉的篇章，有些还是我亲手编辑过的，不敢想象，原来一件事情长期坚持下来会有如此美好的结果。而且这次通过过程改进、工程实践、团队建设、体验设计等几个维度将这些文章分门别类，读者会更容易按图索骥找到自己的所需。借着这个机会，我还特地去 InfoQ 中文站上又找到几个热门的话题，比如云计算、运维、大数据等看了看，每个话题下面都有许多国内外技术专家的辛勤之作。想来也给国内的技术人员带去许多创意和思考，陪伴着大家一起度过漫漫架构设计和编码之路。

分享总是那么让人幸福，也总是会促进社区的进步。站在 InfoQ 的角度，我感谢我们的这些作者；站在读者的角度，我感谢这些可爱的“知识传播者”。祝各位读者阅读愉快。

InfoQ 中国创始人兼 CEO

霍泰稳

前　言

关于本文集

三年前因缘际会，我加入了 ThoughtWorks 中国公司。在这之前，参与线上和线下社区的我，已经认识了不少早期的 TWer，而加入之后更是近水楼台，我跟他们在一个项目工作，听他们的分享，得以从更近距离学习和体会他们身上似乎与生俱来的价值观和对软件构建本身的独到认识。也是一直以“程序员中的编辑，编辑中的程序员”自诩的我，从一开头就认定，传播这群人的意识和经验对于我来说责无旁贷，而这样的贡献也权当是允许自己滥竽充数混迹其中的些许安慰。

这个群体最初聚集了一拨价值观鲜明、爱分享、在社区极具号召力、对软件构建有独特理解和洞察力的人。因为他们的热情、鼓励，甚至偏执和煽动性，在社区掀起了一阵阵潮流，而社区的反应则褒贬不一。但是我想，不该拒绝和否认的是，的确是他们在引领国内的技术和开发社区群体，积极地吸收国外先进的软件思想，对国内软件研发状况和质量多年停滞甚至倒退的状况进行反思。我甚至一度认为这群人这个组织，是国内社区和开发者关于软件构建方面的启蒙者。

需要有人记录这个时代和这个时代的人。我们很难现在就评价这拨人做的事情，这拨人的思想和所做的努力，对于国内软件开发境况的改善有怎样程度的影响，甚至在多年后依然会发现这是件很难的事情。但这些人这些事情的确存在过，他们的努力也一直存在着。

读者对象

本书适合的读者：

- 对软件构建过程进行持续思考和改进的人。
- 对技术新趋势持续拥有热情的人。
- 热衷于敏捷项目管理锻造优秀团队的人。

如何阅读本书

从目录上你很容易看出这本文集分了四个部分：过程改进、工程实践、团队建设和体验设计。这也是遵循了 ThoughtWorks 全球文集的一贯体例。事实上，当你阅读其中某些文章的时候，你会对文章为什么会出现在这个分类下感到疑惑。

但请你相信，编者已经尽了最大的努力，希望把文章放在最适合的分类下，供读者查找和参考。这毕竟只是三十三篇文章，每一篇的标题都足以彰示内容和方向，所以请尽情享受文章的内容盛宴吧。

致谢

感谢我的同事们，这一篇篇文章的作者，可以忍受我不停的威逼利诱和催稿，最后能交付高质量、足以流传的文章。虽然你们中相当一部分人已经离职，但见文如见人。

感谢我的市场部同事小菊和婷婷，你们为这本文集的出版设定流程和计划，帮助我从更多内容中遴选出这三十三篇入选文章，并和出版社反复斟酌合同，她们是让这本文集三年后得以面世的重要贡献者。

感谢人民邮电出版社的编辑陈冀康，继张松的《精益软件度量》之后，他继续为 ThoughtWorks 的软件思想在国内的传播贡献力量。

感谢 InfoQ 中文站的负责人霍泰稳，因为这本文集内的绝大多数文章，都曾经发表在 InfoQ 的中文网站上。谢谢泰稳予以优惠的回购价格，让这些文章具备了可以纸版发行的版权。

目　录

第一篇　过程改进

第二篇　工程实践

第三篇　团队建设

第四篇　体验设计

第一篇　过程改进

看板任务管理

黄亮

作为一个开发团队的管理者，例如当你是一个团队的项目经理的时候，任务的完成情况通常是你最关心的内容之一，比如说分配的任务是否能够按时间完成，整个项目的进度是否尚在计划之中，团队内的人是不是都在高效地工作，大家有没有什么困难，等等。在软件开发团队中，任务的分配、跟踪和管理通常是这个团队管理者的一个重要的工作内容。

1. 从问题谈起

我曾经碰到过一个项目经理，她管理一个团队开发一个 Web 应用，团队里开发人员大概 10 个，测试人员 3 个，业务分析师 1 个人。对于任务的管理她是这样做的。通常，她会将需求分析人员分析得到的需求给每个人分一些。然后每个人在领到任务之后会给她承诺一个大致的时间点。整个项目大致的交付计划用一个 Excel 表管理，根据客户要求的交付时间点，同时考虑到一些需求之间的集成测试关系，她定出了每个需求的大致交付时间点。只要每个开发人员承诺的时间点和期望的相差不大，她都可以接受，这样每个开发人员就知道自己应该在什么时间点交付什么东西。

一切本该很完美，但是不和谐的问题不断出现。最经常发生的事情就是大家在承诺的时间点到来的时候不能按时交付，每次她询问进度的时候，都会被告知还差一点就完成了。通常的说法是“底层部分已经做完了”或“还差页面部分就可以搞定了”，然而实际情况是又过了相当长的时间才真正完成。当然也有按时交付的需求，但是她发现也许是大家经常加班，已经开始疲倦了，有时候明明很简单的、可以提前完成的需求，大家还是到最后一刻

才交给测试。

也有的开发人员拿到自己的那一批需求之后，会批量工作，把若干个类似需求的底层逻辑全部实现，然后再实现上层内容。她默认了这种做法，就像这位开发人员说的“这几个需求都差不多，只要底层做好了，基本上就都完成了”。虽然这部分工作早点和其他人一起集成测试会比较好，但是他这样做也只能推后集成测试的时间点了。还好她承诺给测试团队的交付时间点还在 1 个月之后，只要 1 个月之内能够完成这些需求就可以了。

项目执行过程中还有一些其他的问题，比如有的新人经常碰到问题，但是出了问题并不主动问其他人，而是在胡乱尝试中浪费了时间；还有个别开发人员非常激进，经常花时间去重构代码，追求完美的架构设计，进度很让人担忧。组内的开发人员还经常被其他项目的事情打扰，因为有几个人刚刚从之前一个项目中调过来，前一个项目的有些问题只有他们熟悉并有能力解决。她就不止一次发现有一个开发人员在修复其他项目的 bug。

她会不定时地去询问每个开发人员的开发进度，当需求的计划交付时间点逼近的时候，这种检查会越来越频繁。开发人员感受到压力，有时候甚至需要加班来完成开发工作。尽管她花了很多精力去跟踪和检查每个需求的完成情况，还是有很多出乎意料的事情在不断发生。尽管她一直相信，只要开发人员能够完成任务，采用什么方式她都不干预，而具体的时间也是由他们自己分配的。但是她渐渐感觉到任务越来越不可控，计划通常无法按时完成，每天对大家的检查花费了大部分时间，然而却不能揭示出真正的问题。

运转良好的项目都差不多，而问题项目的问题各有各的不同。虽然每个团队的问题可能不完全相同，但是当我们审视这些项目的运作和管理方式的时候，不难发现一些诸如多任务并行等共性的问题，这些问题给软件项目带来了各种各样的浪费。当一个团队采用瀑布开发模式的时候，开发阶段全部结束之后测试人员才会介入，开展测试活动，在一个通常很漫长的开发阶段内，各种开发活动中的浪费、估计不准确以及成员自己的拖沓、被打扰、问题阻塞等，都被掩盖了。只要在最终时间点前能够全部开发完成，不管是前松后紧还是加班熬夜，都已经成了项目开发的常态。项目经理只能看到交付的最终时间点，问题不能及时地解决，而等到问题暴露的时候，可以使用的调整手段也非常有限。

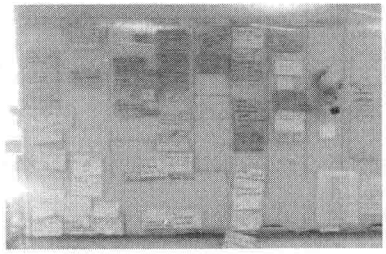
这样的一种团队生存状态在外部环境要求短交付周期、需求允许经常变化的情况下显示出了极度的不适应。市场环境的变化驱动了软件需求的变化，这种变化催生了缩短交付周期的诉求，较短的交付周期使得人们不必去预期过于长远的需求，具备根据市场的变化快速地制定和调整软件需求的能力。而当交付模型由几个月的瀑布模型转变为数周甚至更短的迭代

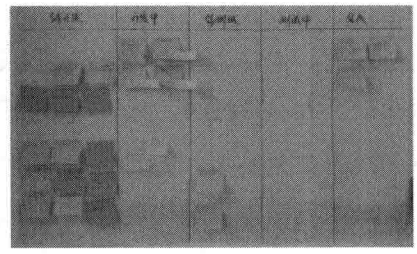
模型的时候，我们在前面谈到的团队中的各种浪费、低效、半成品堆积等问题，就会急剧地爆发出来。

熟悉敏捷方法的读者可能知道，敏捷方法包含一系列实践，来帮助团队实现短周期快速交付，更好地响应需求变化。比如说用户故事（User Story）方法将需求从用户价值的角度进行组织，避免将需求从功能模块角度划分。小粒度的用户故事可以在一两周的迭代内完成开发和测试（并行开发），从而可以缩短交付周期。问题是，在敏捷团队内，我们如何有效管理大量小粒度用户故事，同时避免上述项目管理中的问题呢？下面我们结合敏捷开发中的看板工具来看看敏捷团队是如何管理任务的。

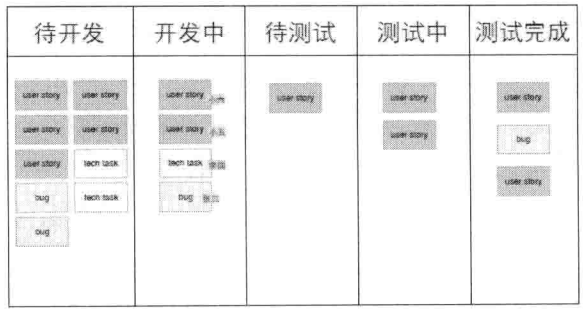
2. 可视化看板任务管理

看板源于精益生产实践，敏捷将其背后的可视化管理理念借鉴过来，经过一番改造，形成了有自己独特风格的可视化管理工具。曾有人总结过 scrum 和 kanban 的使用[1]，而很多时候，我们也将它叫作迭代状态墙。





我们先来看看怎样用这个状态墙来管理迭代任务。说起来其实是一个很简单的东西。



通常一个迭代的状态墙反映了某一个迭代的计划和任务进展情况。状态墙按照一个迭代内团队的典型开发活动分成几栏，例如“待开发”、“开发中”、“待测试”、“测试中”、“测试

完成”等。在一个迭代之初，我们会将计划在本迭代完成的故事卡放到“待开发”这一栏中。可视化状态墙的一个好处就是所有团队成员都可以实时地了解本迭代的计划和进展情况。开发人员领取任务时，就将他领取的故事卡片从“待开发”移到“开发中”，同时贴上带有自己名字的小纸条。当他开发完成之后，就将故事卡片移到“待测试”一栏。测试人员看到“待测试”栏里有待测的故事卡，就取下一张，移动到“测试中”，然后开始这个用户故事的测试；测试完成后，就将故事卡移动到“测试完成”一栏。如果测试人员发现了一个 bug，那么他可以用红颜色的卡片记下这个 bug，然后放到“待开发”这一栏中。状态墙上除了用户故事、bug 之外，还会有一些诸如重构、搭建测试环境这样的不直接产生业务价值的任务，这三类任务用不同颜色的卡片放到状态墙上统一管理。

这样一个简单的工具，是如何帮助我们消除浪费、解决项目管理中的问题的呢？让我们逐条分析一下看看。

2.1 如何减少返工带来的浪费

返工是软件开发过程中的一大严重浪费。比如说开发人员完成的任务交给测试人员测试的时候，关键流程不能走通，阻碍了测试进程；交付给客户的东西被客户说“这不是我想要的东西”；分析人员将还没分析透彻的任务交给开发人员，在最后验收的时候发现开发人员加入了自己的一些“发挥”。这些都会造成返工。返工意味着没有一次性将事情做对，意味着流程中的上游没有交付高质量的产品，也可能意味着团队成员间的沟通出了问题。

在传统的瀑布流程中，我们往往是期望通过前期细致入微的工作来确保一个阶段的工作被高质量完成之后才移交到下一阶段。后来我们慢慢从失败的经验中学习到，这种方法在变化的需求环境下实在是太脆弱，不仅不能如愿保证质量，而且会造成更大的浪费，交付周期也不能满足要求。于是我们引入了迭代式开发方法[2]，一个需求的分析、开发、测试、验收成了一个小粒度的更连续的过程，在这个小的交付循环中，看板帮助我们以更细节的粒度来管理一个任务每个阶段的工作质量。

通常我们是这么做的。当我们把一张故事卡从“待开发”移动到“开发中”时，这张卡片必须是已经分析完成的。也就是说，当开发人员准备真正开始开发这张故事卡之前，我们的需求分析师们必须保证这张卡片所包含的所有内容和细节都已经分析完成，不再有模棱两可的细节，不会留给开发人员过多的自我发挥和想象空间，而且这些细节必须和客户确认过，而不只是团队自己“设计”的结果。

这一道关看似很寻常，实际上很多项目会在这里出问题。很多时候开发人员开始开发的