敏捷测试的最佳实践，第 1 部分：敏捷的实质

IBM 软件开发中心的敏捷测试实践分享

谢 明志 (xiemz@cn.ibm.com), 高级软件工程师, IBM

简介： 本文讲述了作者在两年的敏捷测试和开发工作中的经验和体会。从敏捷的实质，敏捷测试的方法和过程，到如何帮助传统团队转变为敏捷团队做了详细阐述。本文是系列的第一篇文章，着重讲述敏捷实质。

查看本系列更多内容

标记本文！

发布日期： 2008 年 4 月 21 日
级别： 初级
访问情况 1873 次浏览
建议： 0 (添加评论)

从游戏开始……

敏捷开发空间

敏捷开发空间是 IBM developerWorks 为敏捷方法相关资源准备的资源中心。在这一敏捷开发空间里，我们将讨论与敏捷开发、敏捷测试、敏捷配置管理、敏捷项目管理等等，与当下流行的敏捷浪潮相关 的敏捷技术领域。同时，我们还会介绍 IBM 在敏捷方面所进行的最佳实践，以及 IBM 为敏捷开发所提供的解决方案和产品等技术信息。

欢迎您随时访问 敏捷开发空间，获取更多信息。

有个非常有意思的游戏能够帮助大家理解敏捷和传统开发的差异。游戏有两个角色，一个是“老板”，另一个是“员工”，在 2 分钟内，“员工”需要在“老板”的完全指挥下，即“向前一步，向后一步，停，向左一步，向右一步”，完成 60 步移动的任务。“员工”需要执行“老板”的每一个指令，不允许做出相违背的动作。“老板”则不参与行动，只发出指令指挥“员工”的活动。我们体验这个游戏 时，当场 60% 的参与者成功完成了任务，大致估计出我们的工作效率是 50%*60%=30%。游戏后，参与者被问及对这种行为方式的感受时，无论是“员工”还是“老板”都表示非常不满。

接着，大家又做了另一组游戏。2 分钟内参与者被要求独立的、自主的完成 60 步移动任务，在这次游戏里，所有参与者任务相同，大家可以自行决定、并依据自己的判断随时调整其步伐方向，快慢。最后，我们发现所有参与者不但毫无折扣的 按时完成了任务，因而工作效率也达到 100%*100%=100%，而且所有人对于这种新的工作方式更是产生了极大的兴趣。

以上两个游戏方式的对比就折射出传统开发（前者）与敏捷开发、测试活动方式的对比，其中优劣不言而喻。

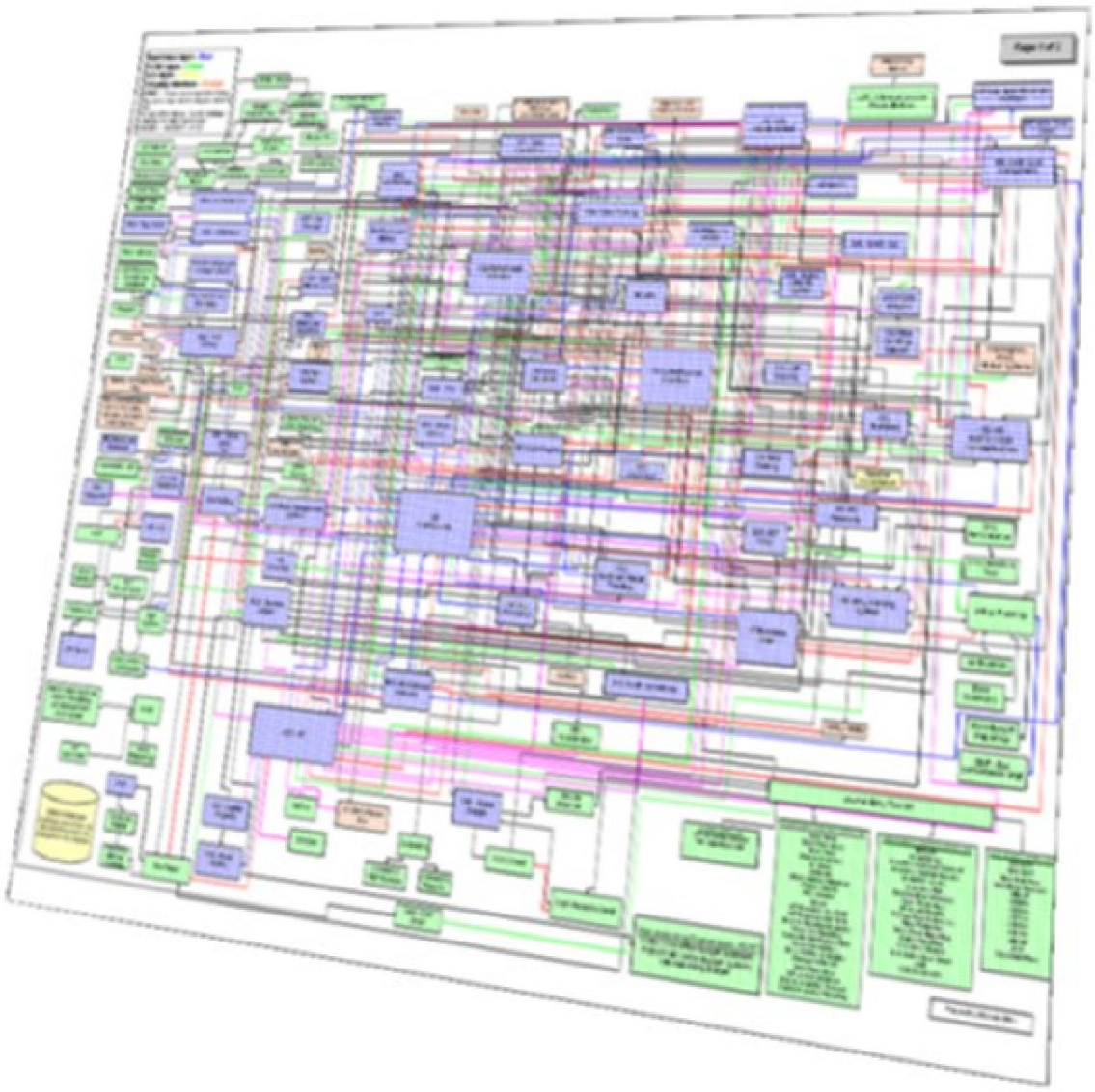
而敏捷开发、敏捷测试又是怎样一个概念呢？他们是否能够帮助我们的团队突破束缚，在日益激烈的竞争环境里表现得更为出色呢 ？请参考我的这个系列文章——“敏捷测试的最佳实践”。

回页首

敏捷的价值

首先我们解释一下什么是敏捷，在字典中我们得到解释，敏捷，即反应迅速、可以快速变化。如今敏捷开发已成为众所周知的时髦 IT 词汇，在这个领域里敏捷又被诠释为迭代的，快速应对需求变化，轻量级，并且简洁。

图 1. 面对客户业务复杂度问题提出敏捷解决方案



IBM 重视敏捷开发，敏捷的软件开发策略之也被广泛推广开来。中国软件开发中心是 IBM 软件部部署敏捷开发方法的重点实验室之一。我们也是 IBM 中国软件开发中心最早使用敏捷方法的开发、测试的团队之一。这篇文章主旨为帮助那些愿意采用敏捷，和正在采用敏捷开发、测试的团队正确了解敏捷的实质。

笔者做敏捷项目已经近两年时间，对于敏捷的理解，认为最为关键的是需要注意两个方面，它们是“高度迭代”和“持续不断的客户反馈”。

- **高度迭代**：迭代就是指产品的开发过程中，一个完整的开发活动周而复始的进 行，产品的功能、性能、可用性在周期活动的叠加中不断得到更新和加强。甚至指在一个迭代周期内产品活动也具显著的周期性。同时，团队间、团队内部成员的高 度协作及时帮助解决了各成员的依赖性问题，因此，也促进了各个成员工作的顺利开展，保障了产品活动稳定的持续性、周期性。以测试为例，传统开发模式下，测 试人员可以因进入测试阶段的条件不完全满足而继续的等待。而在高度迭代的敏捷项目里，不同的是，我们希望测试人员能够尽可能的做能够做的工作，尽可能的早 工作。“等待”在敏捷开发、敏捷测试范畴里已是一种错误概念了。

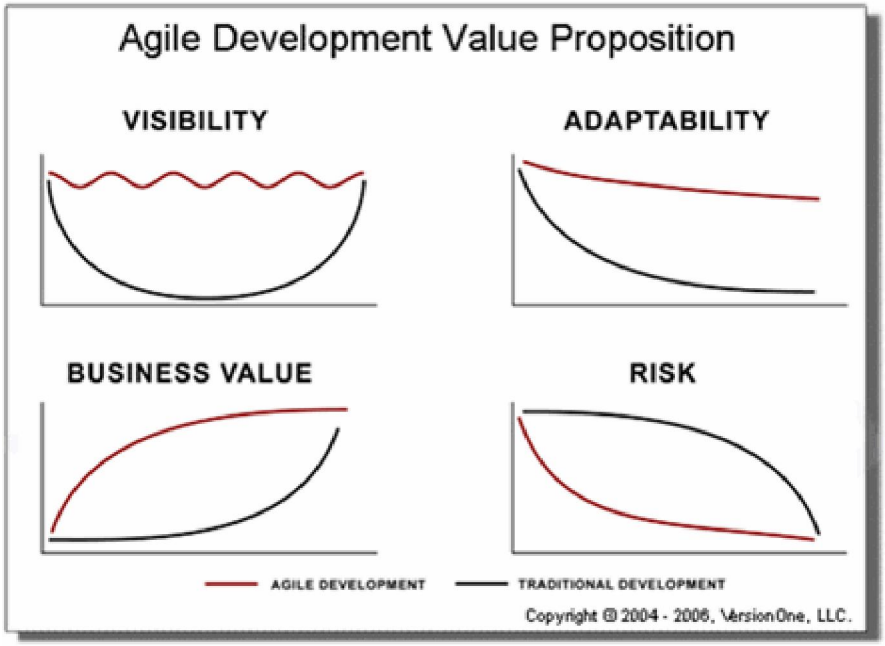
- **持续不断的客户反馈**：指在产品开发任何时期，代表项目业务（Business）的利益干系人（Stakeholder）都要参与到产品的需求分析，设计，以及其他活动的决策制定中来。致力于在短时间内帮助团队实现将客户的需求转化为高质量的可消费产品，并转化成利润。

回页首

敏捷开发的商业价值

敏捷开发自 2001 年《敏捷宣言》（“*AGILE MANIFESTO*”）[1] 的创生，经过多年的打磨和退火已经成为今天非常流行和有过许多成功案例的开发模式。正如前人所说，传统的东西就 是用来打破的，传统的瀑布式开发模式必然逐渐退出历史舞台，敏捷开发、敏捷测试是在新环境里产生出来的打破传统的新开发模式。而敏捷也将会在将来，甚至现 在转化成更适合现代化软件开发、测试团队的方法和实践。在本文的第一部分，我们以两个游戏类比了敏捷和传统开发的差异，这里为了进一步帮助大家对敏捷的价 值有更清晰的理解，我们借鉴前人的研究结果：

图 2. 敏捷与传统开发的比较



首先敏捷开发过程比传统开发要为项目和产品带来更低的风险（RISK）。为什么呢？传统开发缺乏持续的客户反馈 ， 产品一旦从需求阶段退出，整个开发团队近似封闭工作，团队虽努力去实现曾经认定的目标，但因月有阴晴圆缺，市场需求也瞬息万变（例如提出需求的客户已经退 休）。这使得产品在数月后，数年后发布时已经失去了占领甚至进入市场的最佳契机。

而如果你还在考虑使用传统开发模式用现在乃至将来一、两年的时间来开发一个结构复杂，精益求精而又功能庞大的产品，那么你得好好做好失败的准备了。而正是因为出于对这种风险的考虑，越来越多的人认识到敏捷开发要比传统开发能够为企业带来更大的利润空间和更低的投资风险。

其次，利益干系人（Stakeholder）的频繁参，与使得团队在产品开发的各个环节遇到的种种问题得到了及时的解决。因此，我们认为敏捷开发拥着比传统开发更大的透明度（VISIBILITY）。作为老板，项目的负责人一定对这样的开发模式带来的可控性充满了向往。

团队或者产品的适应能力（ADAPTABILITY）也决定着其生命力，因为敏捷开发模式鼓励团队采纳新技术，欢迎变化， 并能够快速应对之，使得产品和团队自身都有很强的适应力和生命力。而在传统模式里，当需求变更时，复杂的变更管理流程要求通过正式讨论、审批，也备以足够 的文档和说明来支持每次“决策”。这不但带来了人力，物力的浪费，更重要的是它严重延长了企业投资到利益回报的周期。而只有拥有反应迅速的敏捷开发才能够 帮助企业赢取市场，降低风险。

以上我们谈及了敏捷开发拥有的比传统开发能为企业带来更大的商业价值和提供企业更大的发展空间。而对于个人而言，笔者认为敏捷开发也提供给了个人更多的发展机会和提出了更高的要求。以下是笔者从个人角度做出的分析。

回页首

敏捷开发有益于个人发展

敏捷项目首先拥有一支支小规模但职能全面的团队，在这样一个普通的敏捷团队里，拥有具备不同职能的 7 名成员，1 名 UCD(User Centered Designer)，1 名 Visual Designer，1 名测试人员（Tester）， 1 名 Information Developer 和 3 名开发人员（Developer）。因此，每一支敏捷团队中的设计、开发、测试、美工、文档等等工作分属给了这个团队里不同的，唯一的人。每个人在团队里 因而必须具有对其所涉及领域强的责任心和领导力。就测试而言，测试人员就是好比一辆跑车里的唯一的驾驶员，项目就好比这辆跑车，测试人员需要及时修正行驶 方向的偏差，确保这辆车在正确的道路上稳步前行。如果，测试人员没有具备足够的责任心和领导力，只是人云亦云，恐怕这种测试要与不要没什么分别，敏捷项目 的质量也更让人担忧，而敏捷也就失去了原有的意义。因此，作为唯一的测试人员，他（她）将拥有对测试的所有权，计划、设计并且执行所有的测试工作。而也因 为拥有了更多的主人翁精神，积极的工作热情，测试人员勇敢的面对工作中的各种挑战，学习新的知识和努力培养自己的工作技能。敏捷无疑对个人发展产生了很大 的推动作用。

敏捷团队中的每个人，需定时和团队的其他成员坐下来看看团队的整体进度，计划下一步工作，并一起探讨所遭遇问题的解决方 案。也需适时的寻求团队中其他成员的帮助解决时下紧要的问题。通过频繁的合作与沟通，个人的协作能力、

沟通能力也得到了较大的提高。下面我们具体就这两个 方面进一步谈谈敏捷开发是如何帮助提高个人的协作能力和沟通能力的。

敏捷开发培养了个人的协作能力

与传统开发不同，敏捷开发更加侧重以人为本，发挥人的积极性，以此提高团队的工作效率。真正实现以结果为导向的职场守则。 作为团队的一份子，无论是测试、开发人员，还是设计人员，他们都将为团队成败担当不可或缺的责任。团队是高度协作的团队，个人除了做好本职工作外，敏捷开 发模式要求个人能够了解和争取更多的扩展性的工作，也能帮助团队其他成员解决他们的遇到的各种独特问题，以加快实现团队的统一目标，即在更少的时间里生产 出能够推向市场的可靠产品。

这里再次提及高度协作，笔者认为高度的团队协作主要表现为以下特点：

1. 团队成员积极分享经验，知识，自我培养所需技能和自我成长；
2. 团队成员相互帮助，愿意成为他人后备队员，随时做好准备投入新的战斗，因而保障了团队的高昂士气和强大的生命力。
3. 任何决策是团队共同的决策，工作是团队共同的工作，团队工作的最后成果因而也是来自团队中每个人的辛勤培养和贡 献。而团队的失败更是整个团队的失败，绝不会是某个人的单方面的过失。这种荣辱与共，共生共息的信念将让团队的力量超过团队各个力量之和。同时，项目团队 的工作效率在团队成员技能的提升和信念的巩固中不断提高

敏捷开发锻炼了个人沟通能力

我们把团队看做一个高度协作整体的同时，不可忽视的是团队内部仍存在的各种矛盾，对这些问题的听之任之将破坏团队的凝聚 力、生产力。这中间反映出来的很多问题不是敏捷方式独有，但团队成员因为敏捷，学会了自己解决各种问题。而相应的沟通能力也随着问题的解决得到很大幅度的 提高。

在过去的项目经验中，我们不难发现种种人与人之间矛盾的产生。而经典的矛盾论也让我们不得不的接受，矛盾是永远存在的，但 这并没有什么可怕。而是一旦我们发现了矛盾的存在，就要迅速找到解决办法，这也是团队的相当重要的工作。尤其是在团队组建初期，团队开始采纳新开发模式 时，锻炼团队解决如下矛盾的工作非常重要：

- 测试人员是质量的工程师，开发人员是产品的缔造者，在对质量标准的认同上常有不一致（当然，传统开发也会产生）；
- UCD 的设计实时反应用户需要，但有时不能顾及代码的可实现性；
- 开发人员而却更喜欢用想当然的理解，和习惯的方式填写代码，甚至主观的抵制接受新设计思想和拒绝新类型缺陷的修复，因此与团队的整体目标、出发点产生分歧；
- 从整个项目组织结构看来，敏捷团队之间（一个项目通常有多个敏捷团队组成，各个团队有自己的侧重点）存在设计，开发的不一致性，这使得产品在整合阶段产生额外的成本。

正如前面所论述，矛盾总是有能够解决的途径，敏捷项目的组织中用管理方式去干预是一种直接、快速的方式，而今天敏捷方法论 者们更加推崇的是鼓励团队内部进行很好的交流和沟通后自行解决。也正是通过不断加强团队间和团队内部的相互理解，不但矛盾得到很好的解决（解铃还须系铃人 嘛），个人的交流和沟通技能也得到了进一步提高。

回页首

敏捷开发培养了个人的创新意识

创新能够为企业带来新发展契机，创造新价值，因此，创新对于企业还是个人而言都非常之重要。不断培养员工的创新能力、鼓励 创新活动也是几乎所有企业的人才培养战略之一。而敏捷开发恰恰就是要打破传统的模式，形成全新的敏捷开发、敏捷测试方法、实践和过程，并鼓励团队采用新技 术和技术创新。因此，团队的每个人需要能够创造性的工作，并乐于接受新事物，通过不断的改进、突破过时的做法，努力提高团队的工作效率，来适应产品的增量 发展需要。

而也因为敏捷开发模式对于很多团队仍很陌生，在运用敏捷开发的过程中我们会遭遇许多新挑战、新困难。如何处理这些问题，解决方案本身就是无以借鉴的，自主创新才是唯一出路。

举个例子，敏捷项目初期，产品停留在原型论证和底层架构初步设计中，产品功能不多，复杂度较小，一般的手动测试就可以将质 量保障做好。产品的中后期，因不断有新需求、新功能的加入，产品复杂度增大。团队倘若仍仅凭固有的手动测试方式来覆盖产品的各个功能、非功能点需求只恐怕 是力不从心了。因此，考虑用自动化测试来提高团队工作效率是可取的方法之一。但是，当产品发展到中后期，也没有富余的资源可以被抽取出来做自动化测试了。即使现招聘新人，也因为新人对产品不了解，只怕自动化测试的效果也未如人愿。那我们是否应该在开发活动的初期就尝试自动化测试的设计、并自动化一部分功能 测试呢？也未然，因为在产品开发初期，产品的功能和界面的变动会比较大，自动化测试收入产出比异常低。因此，何时、何地、如何设计、运用自动化测试来帮助 降低人力成本，提高测试效率是需要我们大胆创新的。

回页首

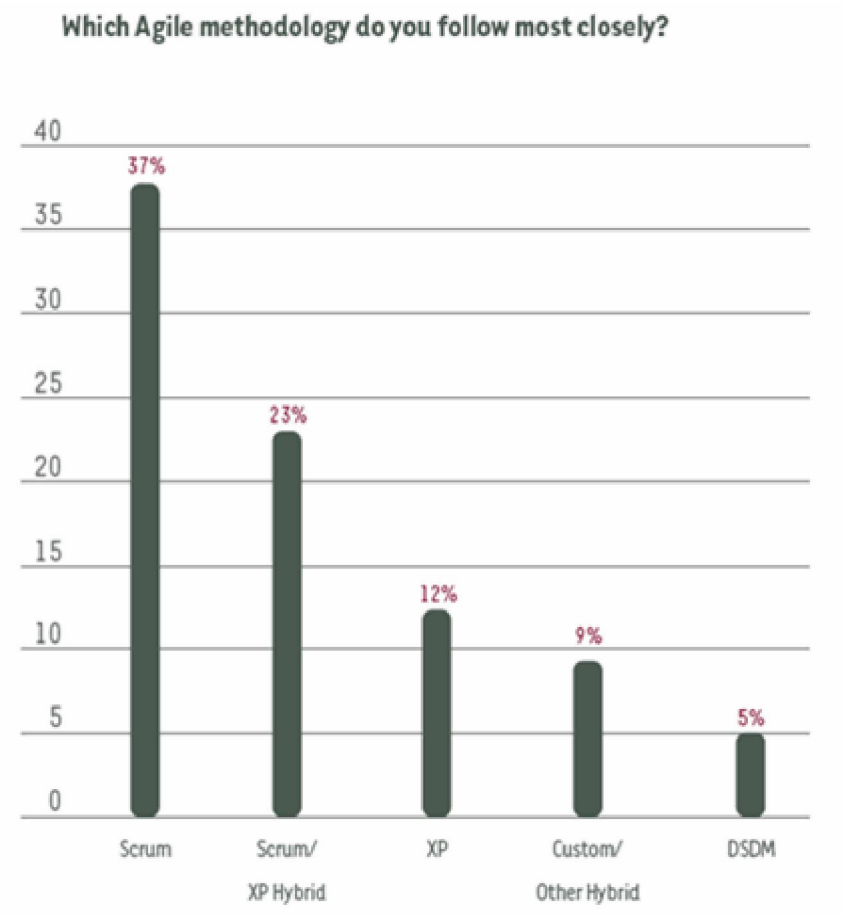
敏捷方法的共同点

以上我们通过商业、个人发展两个方面讲述了敏捷开发的价值和意义。那什么才是敏捷开发呢？在业界又有那些方法是敏捷开发的具体实现呢？各种方法有什么共同点吗？通过下文对各类敏捷方法的共性分析，我们也将进一步了解敏捷的实质。

敏捷方法

业界流行的敏捷开发的方法有许多(要了解各类敏捷方法和分类请参阅本文参考资料中文献），我们需要根据项目大小，性质来选 择合适自己的敏捷方法。比如说 XP（极限编程，eXtreme Programming）更加适合小型项目 3-5 人的团队。Scrum，DSDM (Dynamic Systems Development Methodology) 等更加适合大型团队的项目开发。而敏捷开发也不是一成不变放之四海皆准的准则，而是一个方法的最佳实践。各个团体和企业也不断定制着自己的最佳游戏规则。 VersionOne 的敏捷开发的调研报告中很好的对比了各个方法被业界采纳的比例(见下 图 3)。这个图表就是 2007 年对来自 71 个国家 1700 多人的调查结果的说明。

图 3. 流行的敏捷方法



除了图例中的方法外还有 Crystal, Lean Software Development, Feature Driven Development, Xbreed, RUP 等等。

敏捷方法的共性

虽然各种敏捷方法的名称、所需环境、适合的团队有很大差异，但是他们拥有相似、相同的以下几大特点：

- 拥抱变化（Embrace the change）

无论是多么明智，多么正确的决定，也有可能在日后发生改变。因此，团队要能够充分理解我们的利益干系人 （Stakeholder）和客户代表为什么经常提出新的需求和设计要求，一句话，就是心中有数“唯一不变的是变化”。团队更要信任利益干系人（Stakeholder）做出的每次决定和需求的调整都是将产品开发推向更正确的发展方向，新变化将进一步降低风险，实现团队最大化利益，理 解这是适应市场变化的必然行为。

而在接受变化的同时，我们应该积极的向 利益干系人（Stakeholder）和客户代表反映实现活动中暴露出来的可能的设计缺陷和错误。在实际工作中，团队成员应该用优先级制度来划分事情和目 标先后顺序，在迭代周期内对于还没有最终决定的设计方案可以予以后来实现、测试，不用急于投入资源展开全面的开发、测试活动。这样一来，开发测试团队也会 人员也将更加适应，真正拥抱变化。

- 客户的参与（With Customer Representative on site）

首先谁是客户（Customer），客户代表（Customer Representative） 呢？利益干系人（Stakeholder），或者我们可以理解为我们的客户（Customer），产品的最终使用者(End user),内部使用者(Insider),商业伙伴(Business Partner)。利益干系人（Stakeholder）作为团队中最了解业务（Business）的人物将帮助开发团队的快速达到目标和做出适时决策。 开发团队拥有很好的技术但在业务（Business）方面他们需要 利益干系人（Stakeholder）的帮助。而通常在敏捷的开发项目中，团队中的任何一个人若需要帮助时，只要简单的邀请大家参加一个 15 分钟会议，或一封邮件、一个电话便可以解决。

但是，如果利益干系人（Stakeholder）各执一词怎么办呢？为解决这个问题，将 Product Owner 引入到讨论中来，作为 Product Owner 他可以作为是 利益干系人（Stakeholder）的代表，能够在分歧中做最后抉择。因此，通过这样的客户代表的参与，团队更好的了解了所做事情的价值和意义，其工作 效率也因而得到很大提高。

利益干系人（Stakeholder）能够帮助团队中的每一个人更好，更快的完成了工作，他们的直接参与成为了敏捷开发、敏捷测试的重要前提。

- 较少的文档（With less documents）

敏捷开发更重视生产出可用的产品而不是详细文档。而时常有发觉文档又是无论敏捷还是传统开发、测试不可或缺的一部分。笔者认为，传统开发的文档在敏捷开发里仍有大用，只是原来十来页的内容精炼到现在的一页半页。

敏捷主义者相信文档不是最佳的沟通方式，他们鼓励通畅的交流和沟通，要求避免和减少陈词滥调和空头支票。尤其是复杂的文档 说明只是增加了沟通成本，因而敏捷开发、测试的文档不需要长篇累读，需要的是简洁，清晰。任何一段清楚的文字，甚至一张图片，照片，一封记录着会议记录的 邮件都是我们认可的

敏捷文档。因为是无论是通过文字板书的文件还是其他的沟通方式和载体都是为了帮助团队进行更高效的交流和沟通。只有团队保持着沟通上、 理解上的一致后才能够充分发挥出团队最佳战斗力。但凡这是帮助团队有效沟通的方式，敏捷开发是不会放弃的。

- 最大化的生产力（Maximize Productivity）

敏捷开发模式要最大化的提高团队的工作效率。无论是依靠剪除冗余的文档工作，还是提供民主的、通畅的沟通平台都是为了帮助 团队能够集中有限的精力处理有意义的问题。据调查，通常人会在两个、多个任务并行的情况下产生出出最高工作效率。而敏捷也恰恰使用了各种方法得到团队的最 大生产力。

敏捷开发的 Scrum 模式，要求在计划阶段，团队成员主动定制迭代周期的所有工作任务，因此，本身从团队开始迭代活动的那时起，已经在在多重工作的压力下紧张工作了。而在日常 的迭代生产活动里，各个成员需要当众简单汇报当天的工作进度和承诺下一个 24 小时的工作计划。因此，通过增加敏捷人员的工作的透明度，无形之中，团队成员的生产力进一步得到提高。

- 测试驱动开发（Test Driven Development）

测试驱动开发，是让开发人员在编写功能代码之前，根据对需求的理解先设计和编写单元测试代码。先思考如何对将要实现的功能进行验证，再考虑功能的实现。然后迭代的增加新功能的单元测试和功能代码编写，直到完成全部功能的开发。

- 自动化冗余工作（Automate the redundant work）

将团队成员从冗余的劳动中解放出来，无论是自动化的测试还是自动化工具的开发只要能够节约成本都是敏捷开发、敏捷测试的目标。

- 民主的团队（Democracy in team）

敏捷团队是一支民主的团队，团队关系是平行的，每个团队成员能够平等的参与讨论，决策。传统开发的垂直的官僚机构在敏捷开发中已是过时的。

- 尊重团队（Respect to team）

敏捷团队的决定权交有团队自己，决定是团队统一制定。无论是产品设计方案还是产品的功能实现都是的最佳结果。团队脱离了任 何一个成员的工作都是不完整的，所以我们应当足够尊重其他成员的劳动果实和表达对其他成员的充分信任。尊重团队，尊重团队中的每一个成员都是敏捷开发的原 则之一。

回页首

你敏捷了吗？

你敏捷了吗？经过上面的学习，我们应该已经了解了敏捷的实质，并且笔者认为如果您的团队已经表现出上述的特点，那么您的团 队已经敏捷了。但是，往往很难做到如此理想的敏捷。而同时，我们需指出敏捷与否也并非我们的最终目标，我们的目标是能够通过学习敏捷的方法和最佳实践来开 发可以适用于自身特点的方法和过程，帮助项目灵敏的适应市场变化，让我们变得敏捷起来。

因此，我们依然希望进一步帮助大家了解如何变得敏捷，而首先，还是让我们学习大师留给我们的一套基本准则帮助我们判断项目开发敏捷与否吧。通过按照此标准的衡量，我们将容易得出项目是否敏捷的结论，也能够因地制宜的找到问题所在，最终实现敏捷。

Scott W. Ambler 在其文章 *How Agile Are You?* 中指出了以下七条原则帮助大家来判断什么项目是敏捷的项目。

1. 项目中有利益干系人（Stakeholder）的参与
2. 团队拥有并且可随时执行的回归测试
3. 关注产品自身而不是冗余的文档
4. 项目开发拥有严格的源码管理、版本控制
5. 开发能够积极面对和响应项目需求变化
6. 团队作为整体直接担负项目责任
7. 能够自动化重复性的活动

回页首

结束语

在敏捷测试最佳实践系列的第一篇文章中，您了解了敏捷实质，敏捷为我们带来的更多的业务价值的提升和一个更好的个人发展平台。对敏捷方法的共性分析中，我们了解了敏捷开发的关键特性，也了解了判别是否敏捷的方法和标准。

敏捷测试最佳实践系列的第二篇文章将基于笔者亲身经历的敏捷测试成功案例介绍敏捷测试的实践经验，使您对敏捷测试的结构和 部署细节有更加清晰的认识。最后，本系列的第三篇文章将帮助您了解从传统测试转向敏捷测试所需具备的条件和方法，并分享笔者在敏捷测试的实践过程中遇到过 的几类问题和解决这些问题的主要方法。

接下来，请继续关注敏捷测试最佳实践系列之二，“方法与实践”。

回页首

附加说明

免责说明

本文不带有任何明示或暗含的保证。文章提供的建议或最佳实践只作为一般的经验分享，只在作者的特定环境下验证过。作者不保 证这些建议或最佳实践在任何情况下都有效。本文的内容有可能不太准确或包含错误，作者对此深表歉意。本文中任何带有主观性的陈述都只代表本文作者个人的观 点，不代表 IBM 公司的官方立场。相关细节，请直接咨询作者。

敏捷测试的最佳实践，第 2 部分：方法与实践

BM 软件开发中心的敏捷测试实践分享

谢 明志 (xiemz@cn.ibm.com)， 高级软件工程师， IBM

简介：

本文讲述了作者在两年的敏捷测试和开发工作中的经验和体会。从敏捷的实质，敏捷测试的方法和过程，到如何帮助传统团队转变为敏捷团队做了详细阐述。本文是系列的第二篇文章，着重讲述敏捷测试的方法和过程。

查看本系列的第一篇：敏捷测试的最佳实践，第 1 部分：敏捷的实质。

查看本系列更多内容

标记本文！

发布日期： 2008 年 5 月 23 日
级别： 初级
访问情况 1188 次浏览
建议： 0 (添加评论)

平均分 （共 3 个评分 ）

前言

如果您已经阅读过敏捷测试系列文章的第一篇，敏捷的实质， 您应该已经了解敏捷的定义，了解什么样的团队是敏捷的团队了。而您也可能早已开始思考，什么是敏捷测试的实质？敏捷的测试团队又是如何形成自我管理、自我 发展的组织呢？测试团队又是如何安排日常工作呢？敏捷测试活动与传统测试活动有很大差异吗？为了进一步让您了解如何将敏捷原则运用到活生生的日常测试活动中，我们为您推荐敏捷测试系列文章的第二篇——敏捷测试的实践。

在敏捷活动如火如荼的推广运动中，我们显然无法预知如何在您的特定的复杂环境中您能否最后达成所愿，也无法为您预测出前进 道路的分岔口可能唯一的正确的线路，我们却可以为您点起一盏明亮“街灯”，在这迷雾中驱除黑暗。我们将为您提供一个可以借鉴和可供参考的成功的敏捷测试实 践案例。我们将逐一向您介绍、分析这个案例中的敏捷团队的组织结构，主要的敏捷测试行为，迭代的测试模型和一套以四周为周期的敏捷测试活动时间表。

请您运用您已具备的敏捷实质、敏捷原则的知识，并结合您的独特项目环境、带着您的问题，与笔者一起再度分析这个案例，希望您最终也能得到满意的答案，并随后开始实施部署敏捷测试。

回页首

敏捷测试的实质

测试不仅仅是测试软件本身，还包括软件测试的过程和模式。产品发布后才发现很多问题，很可能是软件开发过程出了问题。因此测试除了需要确保软件的质量，即软件做了正确的事情，以及软件做了应该做的事情以外，敏捷的测试团队还要保证整个软件开发过程是正确的。

敏捷开发的最大特点是高度迭代，有周期性，并且能够及时、持续的响应客户的频繁反馈。敏捷测试即是不断修正质量指标，正确 建立测试策略，确认客户的有效需求得以圆满实现和确保整个生产的过程安全的、及时的发布最终产品。敏捷测试人员因而需要在活动中关注产品需求，产品设计， 解读源代码；在独立完成各项测试计划、测试执行工作的同时，敏捷测试人员需要参与几乎所有的团队讨论，团队决策。作为一名优秀的敏捷测试人员，他（她）需 要在有限的时间内完成更多的测试的准备和执行，并富有极强的责任心和领导力。更重要的是，优秀的测试人员需要能够扩展开来做更多的与测试或许无关，但与团 队共同目标直接相关的工作。他（她）将帮助团队其他成员解决困难、帮助实现其预期目标，发扬高度协作精神以帮助团队的最终获取成功。需要指出的是，团队的 高度协作既需要团队成员的勇敢，更需要团队成员的主动配合和帮助。对于测试人员如此，对于开发、设计人员，其他成员也是如此。

回页首

敏捷测试的方法与实践

是的，敏捷测试也需要高度迭代工作、频繁得到 STAKEHOLDER、客户的反馈，需要动态调整测试计划、测试的执行。并且，敏捷测试人员参与到了更多的敏捷生产活动中，积极的影响了团队做出的决定和计划。

是的，“人”才是敏捷的实体，敏捷测试也是以人为本的。不难理解，“敏捷”的一切都围绕着人展开，如敏捷鼓励直接，平行的 沟通；敏捷需要持续的客户反馈以及敏捷活动的设计，方案和决策需要团队协同制定等等，敏捷测试需要一支非同寻常的团队，不同于以往传统开发模式下的团队结 构。关于敏捷团队、敏捷测试团队的组成和介绍，将是我们讲述敏捷测试实践的第一步。

“人”是重心，方法、策略是辅。为了适应不同的团队结构，不同的项目环境，敏捷项目和敏捷活动的实践也应该因“人”而异，但是，并不是说可以天马行空，我行我素。一旦脱离了正确的敏捷方法、和敏捷原则的指导，我们的敏捷活动就好比摸黑前行了。

这正是我们需要学习前辈和敏捷主义大师们的经验意义所在了，笔者在过去的实践中受益颇多的也正是前人的实践经验和方法。因 此，学习前人的经验和方法，并运用这些最佳实践来帮助敏捷开发团队，甚至是传统团队来解决时下重要的问题是十分有意义的事情。笔者不敢妄自尊大将自己的一 般实践纳入经典方法范畴，但经历了两年的研究和改进，笔者提出的敏捷测试的原则也得到了业内同僚和“大师”的普遍认可。经过多次和其他项目团队的经验交 流，我们也不断的改进着我们的原则、方法。因此，笔者要非常感谢参与讨论的同僚们，没有你们的热情参与，也不会有今天的笔者信心百倍的执笔了。正如笔者在 借鉴了前人的成功案例中的经验和方法之后定制了符合项目需求的测试原则一样，相信，读者们在阅读了笔者的敏捷测试原则和方法后，同样也会有所收获。而对笔 者经历的敏捷实践活动中的方法和测试模型的讲解将成为我们讲述敏捷实践的第二步，也是本文的重点。

综上所述，笔者将运用本文的主要篇幅为大家讲解这个敏捷实践。它们是：

1. 敏捷团队组织构成，敏捷测试团队的任务和使命；
2. 敏捷开发团队以测试为驱动的开发方式——测试驱动开发，这是种独特的测试？还是开发？
3. 递增型的迭代测试，它首先是对敏捷测试过程活动和生命周期模型的介绍，通过学习经典的敏捷增量测试模型，我们将敏捷测试的各类活动有机的组合到了一起。在此之上，对定制后的独特敏捷增量测试模型的分析和理解，帮助我们理解测试活动的规划和管理；
4. 以及需要特别关注的递增型迭代测试的关键活动之一——“静态测试”；这也是笔者认为的最高难度、最具影响力的敏捷测试活动。它将测试团队最早的引入产品开 发环节，测试人员以第一用户的角度判断设计的有效性，此活动较早的暴露了设计缺陷、避免了团队对目标的不一致理解等，是测试活动中最有创造性价值的部分；
5. 最后，笔者将谈谈测试活动中的测试计划和管理，即关于测试任务估计，测试活动计划，各个重要测试活动时间分配与安排的介绍。

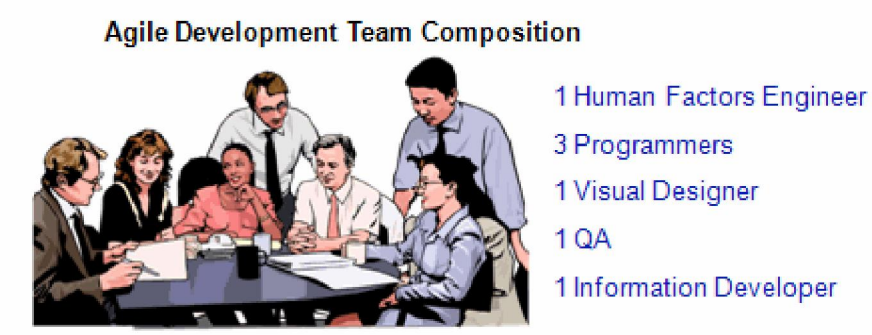
然而，敏捷测试不是一蹴而就的，做到真正的敏捷，无论是从传统测试模式向敏捷测试的过渡，还是组建全新的团队都是需要循序渐进的，同时也需要团队成员的通力合作和不断的实践来完善敏捷测试的实践原则和方法。

回页首

敏捷团队

考虑到敏捷团队的组织结构，让我们以笔者亲身经历的项目为例来说明。笔者曾共事的整支产品开发团队被划分成 4 个相对独立的敏捷开发队伍，而每支队伍拥有相同配置的 7 名成员，他们分别具有不同的职能属性。如图 1 所示，每支敏捷队伍组成成员角色包括 1 名 UCD（User Centered Designer），主要负责产品的主要设计，其工作主要包括界面设计、用户的用例设计等等；1 名 Visual Designer，主要负责产品界面的色彩搭配、控件的外观设计和 UCD 界面设计方案的初步实现和美化；1 名 Information Developer，主要负责产品中信息的编辑和重要文档的撰写工作； 3 名开发人员，主要负责产品的实现。和 1 名 QA，主要负责产品质量的保障。（更多的我们将 QA 定义为具有相比于测试人员拥有更多责任的一个职能，在本文中，为了简便起见，我们仍称之测试人员）。4 支敏捷的队伍拥有相同的 SCRUM MASTER STAKEHOLDER。通常会在同一时间进入一个迭代周期，制定各自的敏捷计划，并在同一时间退出，发布各自功能实现。而 4 支队伍的劳动果实被集成到一起就形成了可发布的产品了。

图 1. 敏捷开发团队的组织结构



因为敏捷团队中只有 1 名测试人员，因此需要一臂承担测试策略的制定，测试计划，测试脚本，测试用例设计以及测试的执行，帮助团队发现潜在问题，并协助解决问题的工作。敏捷团队 的敏捷原则也是测试人员敏捷活动的规范，测试也需要拥有和团队的良好沟通，高度迭代的活动和不断的获得 STAKEHOLDER 的反馈。那团队的结构与敏捷本身有什么直接关系呢？与敏捷测试又有多少关联呢？

谈到这里，想起曾经有朋友向我咨询有关敏捷团队的某些职能的人力配备的问题。其实，笔者也无意论证 7 个人为什么是最佳组合，为什么不是 17 个，20 个人的组合。但是，敏捷原则告诉我们敏捷团队是高度协同、民主和平等的团队，为了让团队中每个人充分高效的工作。相同职能下的组员至多不好超过 3 名，最佳配置也是不同职能下配置 1 个人头。因此、在这样一个小型、平行的组织结构里，沟通更加易于建立，沟通复杂度也相对较低，相比 17、20 人的团队组织，

沟通的代价也小很多。相反，很难想象在一个敏捷团队中会拥有诸多不同风格的执行者，决策者将是个怎样的混乱情况。

此外，经历过敏捷测试的体验，我们发现一个单一的敏捷团队最好保持较小的“尺寸”。这是因为拥有很多测试人员的敏捷团队通 常不但需要更大的实际工作量来匹配庞大的机体而导致团队任务量更巨大，更复杂，失去自我管理的信心，而每个测试人员也将要花费大量精力和时间投入到内部沟 通，和可能因为内部缺乏一致而导致的更加频繁的反复沟通中。

每名敏捷的测试人员需要和其他敏捷团队成员保持频繁而必要的沟通以保证对目标，需求、设计的充分正确的理解，对需求变化能 够迅速的做出反应。另外，还需要与职能队伍中的其他测试成员保持一致性。如此一来，沟通代价激增了，它将占到测试人员的工作中的较大比例。而这种内部沟 通、协调，却不能定义为敏捷的 Backlog 项目来计入团队整体的工作输出。因此，整体的测试效率并不一定随着人力资源的投入而递增。非但没有实现敏捷原则，而恐怕因为团队的组织结构变得更加庞杂。 所谓的“自我管理、自我发展”的团队只能因而依靠传统的管理和规划了。笔者认为，除了因为特殊阶段，特殊时期，敏捷团队需要“聚合”更多资源来一并解决存 在的高优先级的体力型问题外，敏捷的团队应该尽量保持着较小的“尺寸”。

果真项目投入了很多的人力资源，无论设计还是实现、测试团队拥有较大的人数，那么我们应该考虑将这样的团队可以分得更小 些，工作量也相应分得更精细些，直至接近我们建议的最佳组合。至少我们认为要做好敏捷测试，就要确保敏捷团队中的每一个职能拥有足够清晰的职责范围和一定 程度下自由空间和在这个空间内的充分授权。因此，但从人数和职能构成上，敏捷团队的构成一定是不可忽略的重点。

正像我们前面提到的，确认软件开发过程的正确性也尤为重要，因此作为敏捷的测试人员，更要了解敏捷的过程，比如说，测试人 员需要帮助 UCD 和开发人员确定需求的可行性，测试人员要督促开发人员及时发布 build，以保障迭代结果最终能够在通过足够的测试后成功发布等。在 build 发布后，测试人员要首先验证当前迭代的任务是否已经完成，其次才是验证产品功能的正确性。也为了能够在日后重复性和体力型劳动中解放出宝贵的时间去覆盖测 试更加紧要的内容，如可用性，全球化等，测试人员需要自动化一部分测试，创新的、灵活的工作。

敏捷团队是自我管理的团队，高度协作的团队，质量目标即是测试团队也是整个开发团队追求的目标，因此开发团队应将做好单元测试，设计团队将帮助测试团队设计好测试用例作为计划内的一项工作。这里我们推广、建议开发人员采用、普及测试驱动开发模式。

回页首

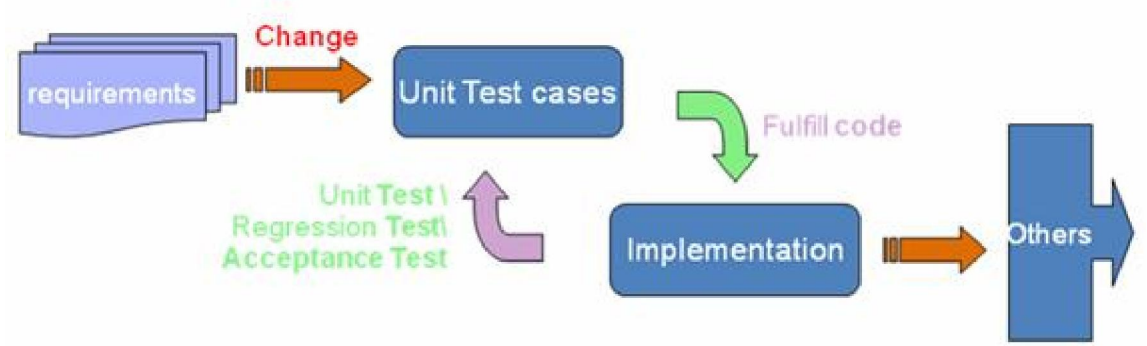
测试驱动开发

测试驱动开发表现出迭代开发的最核心的就是开发人员自己能够第一时间确认其需求得到了正确实现，而当单元测试覆盖了更多的 内容，代码质量也将得到提高。测试驱动开发的指导思想就是让开发人员在编写功能代码之前，先根据需求编写测试代码。先思考如何对将要实现的功能进行验证， 并完成单元测试脚本的编写，然后编写足够，仅仅足够的功能代码满足这些测试用例，直至通过测试。按照这个方法，递增的在迭代中增加新功能的单元测试和功能 代码编写，直到完成全部功能的开发。

在单元测试活动中，测试人员也被鼓励参与到单元测试的设计中来，不但可以帮助开发人员构思出更多的单元测试用例来扩大单元 测试的覆盖率，还可通过学习如何使用单元测试，如何复用单元测试用例到回归测试和功能测试，以达到最大化利用有效的资源（如下图）和节约成本的作用。同 时，在回归测试和用户接收测试（User Acceptance Test）中如能将单元测试脚本有机的关联起来，并自动化其执行，更能够进一步提高测试的成效并降低测试成本。

单元测试脚本因随需求、设计的变化随时更新。需要开发人员站在全局的立场，开发始终坚持先修改测试脚本，再修改产品原代码。此时，也建议测试人员参与单元测试脚本的改进，帮助开发人员合理的变更单元测试脚本，同时着手修改测试计划和测试策略。

图 2. 测试驱动开发



回页首

递增的迭代测试

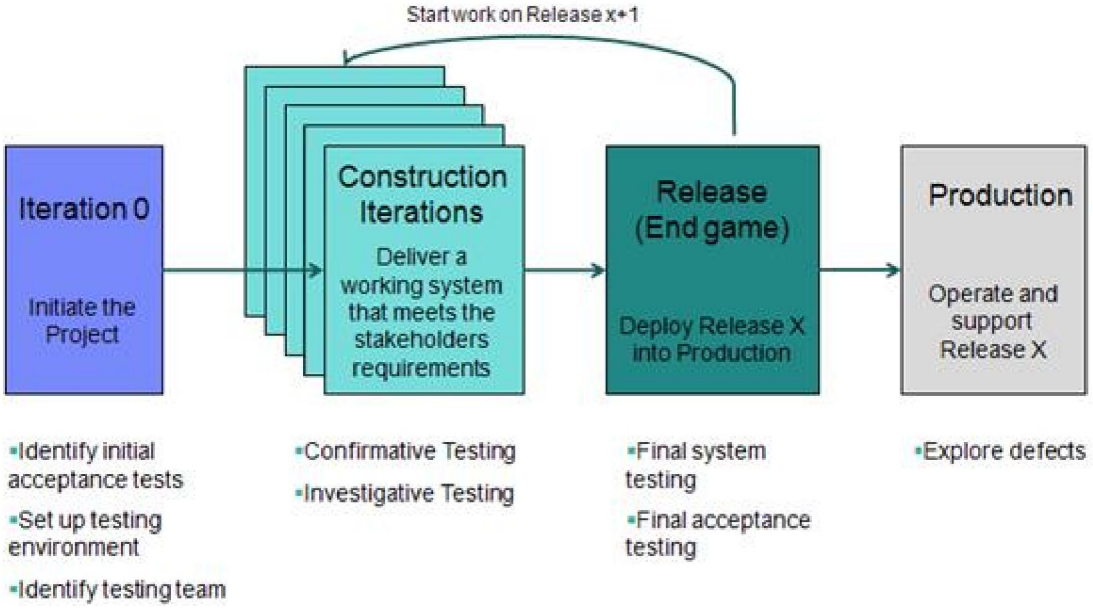
测试驱动开发的原则应该运用于每一迭代周期的开发中。但是，测试驱动开发的单元测试仍然是以开发为目的的活动，虽然自动化 测试，回归测试和用户的接收测试（User Acceptance Test）可以通过复用单元测试脚本提高以后的测试工作的效率，但单元测试不是我们敏捷测试讨论的重点。

敏捷测试活动的主要承载者还是敏捷测试人员。测试人员如何运用敏捷原则指导测试活动还是笔者在敏捷测试实践一文中要讲述的重点。以下，笔者将通过两个迭代测试模型来帮助了解测试人员如何结合敏捷原则实践敏捷的测试活动。

经典敏捷增量测试模型

测试是敏捷开发过程重要的环节，自始自终测试贯穿于每个迭代。Scott W. Ambler 认为就整个产品的敏捷开发生命周期可以分为 4 个阶段，即初始阶段，项目的建设阶段，产品发布阶段和产品的维护阶段，在关键的项目建设阶段中，测试被分成两个部分，Confirmatory 测试和 Investigative 测试。[1] 下面，我们来讲讲迭代的测试的这两个方面。

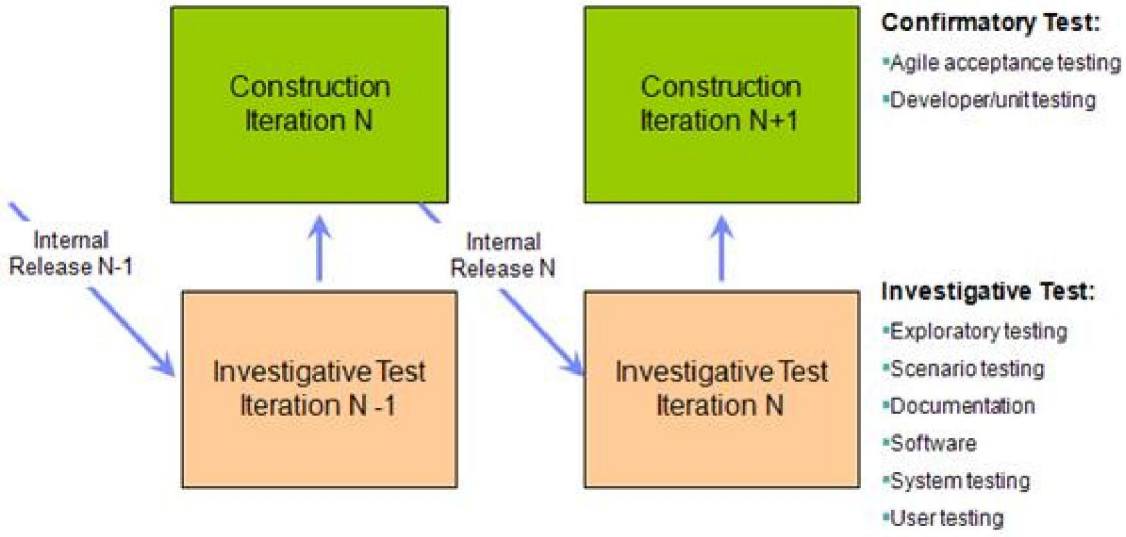
图 3. 敏捷测试生命周期



Confirmative 测试就是对 build 的有效性和关键的功能是否正确进行验证，测试人员依据测试用例和测试脚本的完整测试是工作的重心。原文中的 Confirmative 测试包含了开发人员的单元测试（必不可少的重要开发活动）和被称之为 Agile Acceptance Testing 的测试部分，这段时间的测试任务用来保障迭代的必须输出的质量。基本的功能、非功能的任务，但凡是在迭代开始时制定的计划中承诺的高优先级需求，哪怕是很 繁琐的细节工作都要被充分的测试和验证。

Investigative 测试是对 Confirmative 测试的补充和是更广泛的测试活动，测试团队在完成 Confirmative 测试后的时间用来做这部分测试，它包含功能测试，文档测试和系统测试以及和其他产品、环境之间产生的必然的 Integration 测试，还有个非常有趣的测试活动叫做 Exploratory 测试，笔者认为这部分测试是测试人员创造性的采用多种不同途径去尝试测试产品。就好比“猴子敲键盘”一样，测试员使用各种手段来考验 build 和产品的稳定和正确性等。

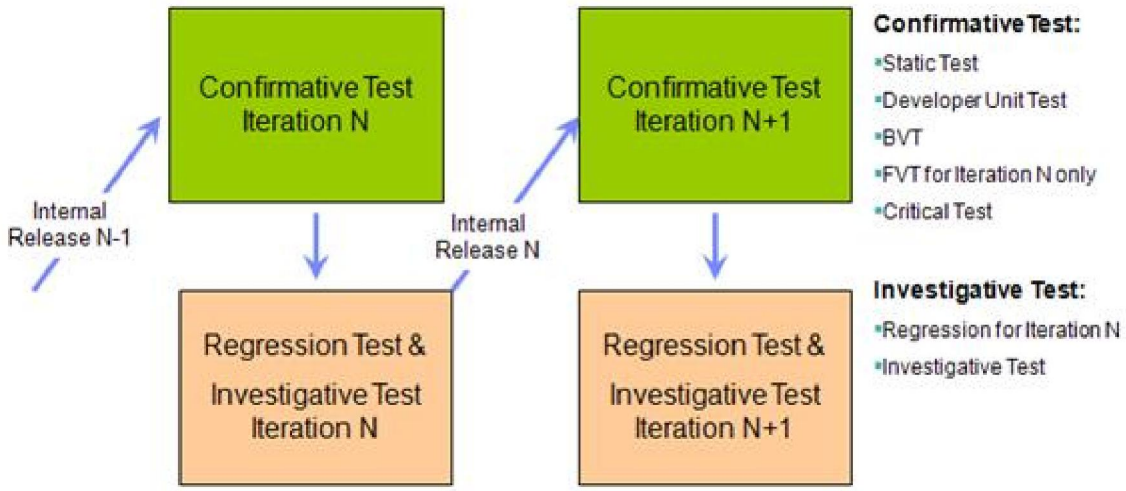
图 4. 敏捷测试的 Incremental Testing



自定制的敏捷增量测试模型

我们在敏捷项目开发的过程中使用了定制的测试流程，我们同样有相同的两部分测试，即 Confirmative 和 Investigative 两部分。不同的是，我们原则的将这两部分测试都放在当前迭代的计划内完成。原因是，敏捷测试团队针对当前迭代的任务计划本应服务于当前的产品，过去的迭代 产物，或者因为需求变更不再适用，又或者因为未解决的质量缺陷使得实际测试效果不佳。而同时，因为 Product Owner 和 STAKEHOLDER 的期望是团队能够高效的完成当前迭代的任务，完成更高优先级的工作，每个迭代的考核亦非常清晰。为了完成服从当前的高优先级任务，计划，也为了 STAKEHOLDER 更好的关注和帮助当前问题的及时解决，测试人员对以往 Build 的测试应应合理的计入先前迭代的任务而不是当前迭代计划。倘若真要测试以往的产品而不是最新的，敏捷测试的管理也将变得有些困难，同时测试团队所关注的问 题也只能是过时的，只能成为团队的低优先级的问题。这不是与团队整体的目标背离吗？因此，我们建议测试团队使用我们改进后的敏捷增量测试模型，即在当前迭 代仅仅完成当前迭代的计划，而所有测试都需要围绕最新的产品 Build 进行。

图 5. 定制的 Incremental Testing



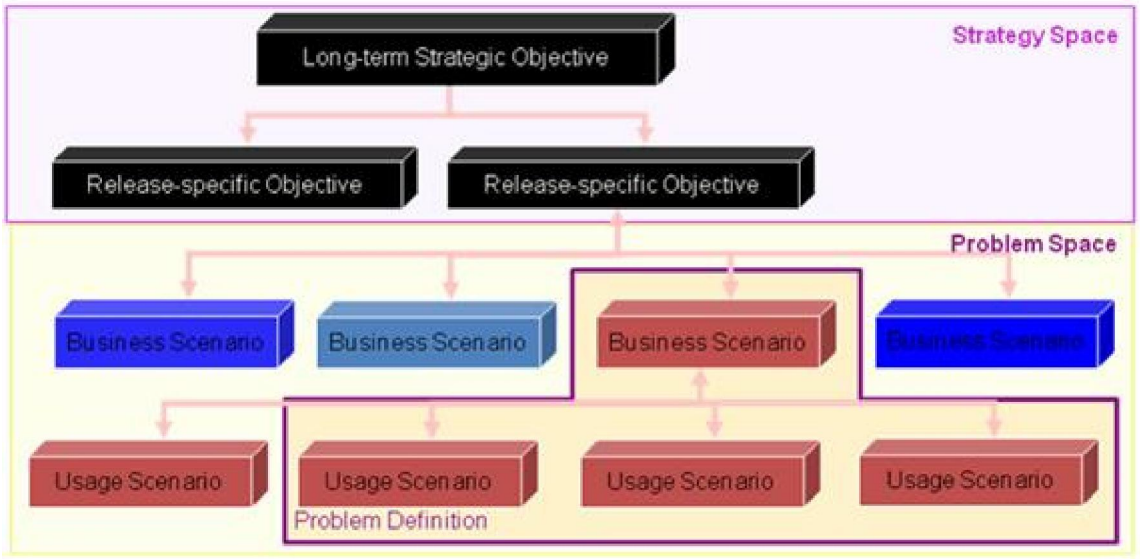
在我们新的增量测试模型中 Confirmative 测试包含对需求的静态测试，开发人员做的单元测试，以及 Build 验证测试，功能测试（仅限于对当前迭代功能）和重要的其他类型测试。通过了 Confirmative 测试的产品 Build 就可以作为在迭代结束时发布了。而为了产品拥有更好的质量，也为了完成对那些较低优先级的质量验证的需求得以确保成功的实现，我们需要针对性的做系统测 试，压力，并发，安全甚至全球化的测试，这部分我们把它叫做 Investigative 测试。还需要强调的是，为了保障产品的最终稳定，为了产品不会出现在代码重构后的质量回归现象，我们将回归测试（Regression Test）放在这个阶段，用以涵盖对以往关键功能的再度验证。

静态测试

在笔者的测试模型里，confirmative 测试增加了“静态测试”，本人认为这部分测试是对测试人员最具挑战的部分。一个好的测试人员能够第一时间找到需求分析、设计中的模棱两可，遗漏，错误的地 方，能够促进团队前期工作的高效完成，将很大程度降低将来产品的质量缺陷的数量，积极影响了敏捷开发的最终输出。这部分工作是测试团队，开发、设计团队最 默契合作的阶段，交流非常频繁，正是通过积极的沟通和及时的修正与团队目标“误差”使得团队更加明确其方向，更有凝聚力和也得以发挥了团队的最佳战斗力。在笔者的项目经历中，往往这个阶段会需要一个迭代周期 1/4 左右的时间。这同时也说明了静态测试在敏捷测试类型中的重要性。

在敏捷开发过程的静态测试即项目迭代开发前期测试人员的最主要工作。值得再次强调的是，在这段时期测试人员的工作重心是认真了解需求和用例设计，并针对设计的可行性，可用性进行验证，确认设计是对需求的准确实现，最佳实现。

图 6. 静态测试需要的 Strategy Thinking



对于静态测试的方法，我们在这里需要推广 RUP 的 Use Case，这是个很好的分析需求的方法，也是测试人员作为静态测试的使用的方法之一。对产品长期战略和业务的熟悉可以帮助测试人员更好的理解团队的用例设 计，视图、模块等，也能够通过对比分析，直接找出某些设计中的缺陷，以提高设计的质量。项目的开发前期阶段，设计是占有非常重要的比例，而设计是直接影响 产品质量的环节，因而确保这一阶段的质量对产品的品质提高起到决定性作用。针对开发出来的用例，设计者对用例的描述通常故事情节比较简单，缺乏完备和逻辑 的缜密。而开发和测试需要的是详细的设计，这个用例设计和各种辅助的逻辑应该将用例定义的清晰和明确，例如边界条件，例外条件，数据的格式和其他性能指标 的界定等等都需要涵盖。因此，测试人员应该领导团队帮助明确出用例更多的细节。比如，在一次设计用例讨论中，设计者提出“我需要一个 Overlayer”。那么测试人员应该要问：“如何打开这样一个 Overlayer，如何关闭？”“这个 Overlayer 需要多大平面尺寸，是否支持调整，是否支持对屏幕大小的自动适应”，“Overlayer 的打开和关闭是否需要有动态渐变的效果？”，“Overlayer 的是否需要滚动条？”，等等。

这个过程能够帮助团队发现早期的设计缺陷，通过发现问题，并定制新的设计方案，团队也通过这个过程，更好的了解了测试的重 要性，也摒除了可能存在的对需求的种种“假设”，因而更加明确了团队的目标和方向。这是个非常关键的过程。尤其是对测试人员而言，任何“假设“都是有害 的，所以一定需要把不清楚和模棱两可的问题从一开始就理清楚。

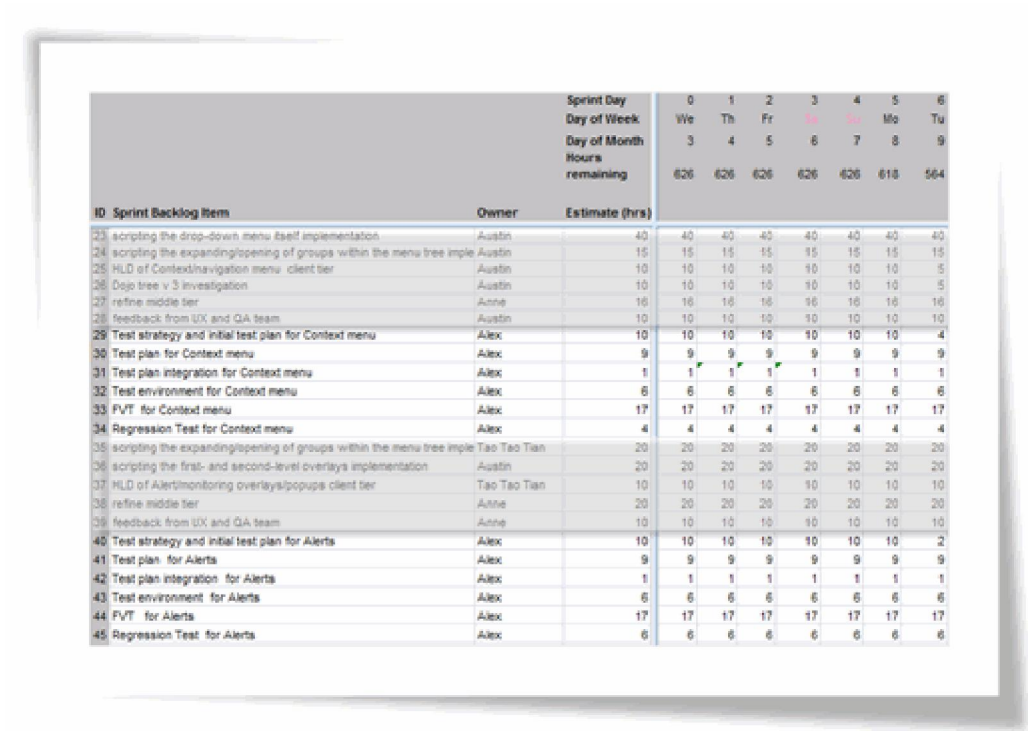
回页首

敏捷测试的计划与管理

做好了测试的思想准备，并了解如何从需求开发出测试用例，敏捷测试人员接着需要做的就是参考产品需求和团队的设计制定和计 划测试任务和各种测试活动，即测试策略和测试计划的制定。每个迭代敏捷开发都将关系产品的功能点和非功能点的需求作为产品的 Backlog 编入待开发的序列，敏捷团队从中会选择适量的 Backlog 项目来作为当前迭代的 Backlog 去实现。测试人员同样根据需要制定出相应的测试工作，并罗列于团队的 Backlog 项目中，承诺了在迭代结束时可以做好的足够的测试工作。

对于测试计划中的 confirmative 测试，这部分必须做到高质量的按时完成。而对于 Investigative 部分，我们应该适量的计划到当前迭代中，切忌不要做 over commit。

图 7. 计划测试 Backlog 项目

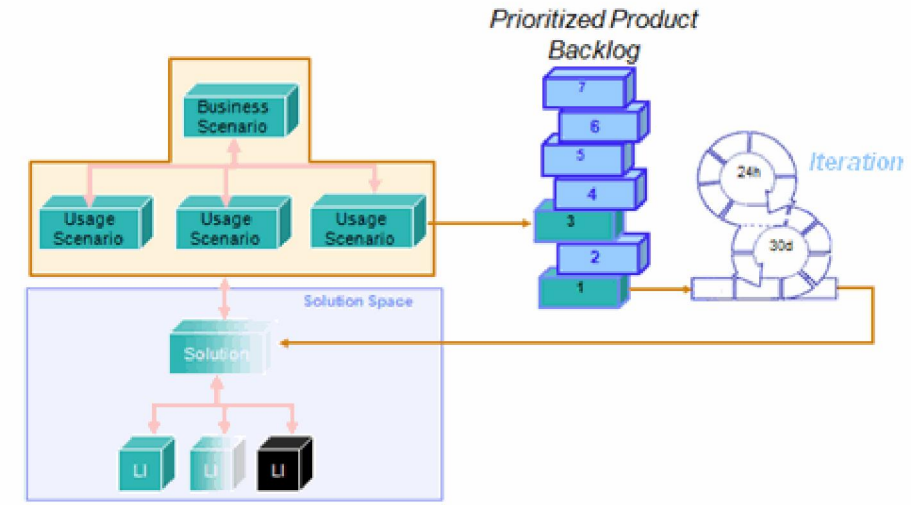


ID	Sprint Backlog Item	Owner	Estimate (hrs)	Sprint Day	0	1	2	3	4	5	6
				Day of Week	We	Th	Fr	Sa	Su	Mo	Tu
				Day of Month	3	4	5	6	7	8	9
				Hours remaining	626	626	626	626	626	618	564
23	scripting the drop-down menu itself implementation	Austin	40		40	40	40	40	40	40	40
24	scripting the expanding/opening of groups within the menu tree imple	Austin	15		15	15	15	15	15	15	15
25	HLD of Context/navigation menu client tier	Austin	10		10	10	10	10	10	10	5
26	Dojo tree v 3 investigation	Austin	10		10	10	10	10	10	10	5
27	refine middle tier	Anne	16		16	16	16	16	16	16	16
28	feedback from UX and QA team	Austin	10		10	10	10	10	10	10	10
29	Test strategy and initial test plan for Context menu	Alex	10		10	10	10	10	10	10	4
30	Test plan for Context menu	Alex	9		9	9	9	9	9	9	9
31	Test plan integration for Context menu	Alex	1		1	1	1	1	1	1	1
32	Test environment for Context menu	Alex	6		6	6	6	6	6	6	6
33	FVT for Context menu	Alex	17		17	17	17	17	17	17	17
34	Regression Test for Context menu	Alex	4		4	4	4	4	4	4	4
35	scripting the expanding/opening of groups within the menu tree imple	Tao Tao Tian	20		20	20	20	20	20	20	20
36	scripting the first- and second-level overlays implementation	Austin	20		20	20	20	20	20	20	20
37	HLD of Alert/monitoring overlays/popups client tier	Tao Tao Tian	10		10	10	10	10	10	10	10
38	refine middle tier	Anne	20		20	20	20	20	20	20	20
39	feedback from UX and QA team	Anne	10		10	10	10	10	10	10	10
40	Test strategy and initial test plan for Alerts	Alex	10		10	10	10	10	10	10	2
41	Test plan for Alerts	Alex	9		9	9	9	9	9	9	9
42	Test plan integration for Alerts	Alex	1		1	1	1	1	1	1	1
43	Test environment for Alerts	Alex	6		6	6	6	6	6	6	6
44	FVT for Alerts	Alex	17		17	17	17	17	17	17	17
45	Regression Test for Alerts	Alex	6		6	6	6	6	6	6	6

接着，测试人员需要撰写测试用例和测试脚本了。测试用例的撰写和传统测试基本没什么差异，按照已有的模式撰写就好了。笔者 的测试团队，使用了 XML 文件格式保存所有测试用例，原因主要是沿用了测试团队原有的习惯，而同时也因为 XML 测试用例能够更好的匹配自动化测试的需要，并且便于更新。测试脚本是自动化测试的产物，敏捷开发周期短，变化多，很难拿到一个稳定的产品再开始做自动化。 而自动化脚本的设计自然要避免去测试不稳定部分，开始的迭代周期几乎百废待兴，自动化作用不大，到了中期，笔者的团队自动化测试才稍成气候。

测试人员需要的是在根据测试策略开发出这相应脚本和用例，需要把握测试范围，与计划和测试策略一致，测试也要量力而行，做到足够的测试，保障迭代的正常退出就很好了。

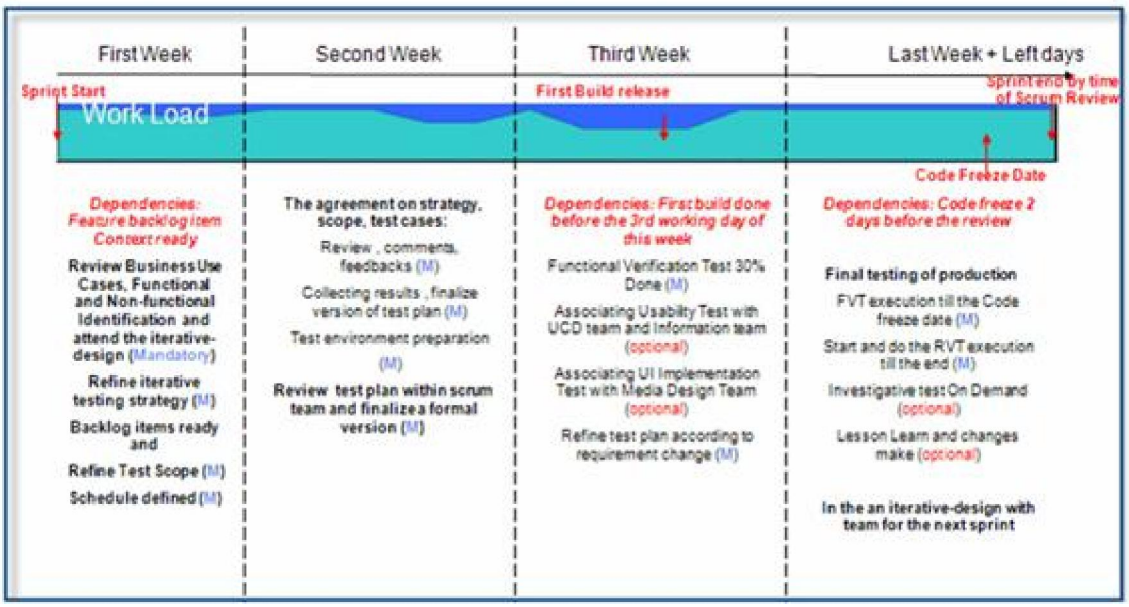
图 8. 依据 Business Scenario 制定测试策略



敏捷测试的活动时间表

通过实施上述的敏捷测试原则，测试团队可以逐渐形成符合自身特点的敏捷测试流程、敏捷测试最佳实践，久而久之形成独立的敏 捷团队文化。以笔者所在团队为例，历时 1 年，经历 12 个迭代后，我们逐渐形成了一套可以遵循测试活动时间表。我将他放到文章的最后，这张表包含了敏捷测试团队的各项活动安排和必要的前提与进入条件。在这张表 中，测试团队较早的涉入，较早的开展测试，以及各项相关工作，并与设计和开发人员紧密的合作，它确保了测试团队乃至整个敏捷团队的工作的按期完成，是值得 向大家推荐一种最佳实践。因为篇幅关系，这里仅对其做简单的描述。

图 9. 敏捷测试 Work Flow 最佳实践



第一周的工作如先前所讲，做静态测试，确定测试策略和制定可行的测试计划，划定测试范围。这个阶段的前提是敏捷团队确定了当下迭代周期内需要完成的 Backlog 项目。

第二周的工作是准备开始测试的执行，因此要准备好测试用例和测试环境。特别的是，测试人员是根据需求和团队讨论、设计出的 用例来开发测试用例的。虽然测试用例的细节程度不能与传统开发中针对已经开发完的产品、产品开发文档开发的测试用例相比，相反，许多细节，尤其是还未由团 队最终确定的内容仍是待定状态；但是，这些细节并非影响测试的用例设计，相反它不但简洁、易懂，也拥有很好的灵活度，它能够实时响应各种变化。而且，测试 用例记录了大量的待定部分，它时刻在测试人员的脑中，测试人员用这种方式简单的告知团队，我们还有未完成的设计和未定的方案，测试用例帮助团队对产品的理 解同步于此。

第三周的第三天，第一个可以执行的 Build 已经能够发布了（这个前提需要开发人员的密切配合）我们开始功能测试了。到第三周周末，一部分功能测试已经完成，大部分关键功能得到验证。

第四周我们要结束测试，包括 Confirmative 的测试部分和 Investigative 的测试。在迭代验收之前团队要通力合作解决各种能够解决的问题，保障 Backlog 的如期完成。这里有一个问题值得再次提出来和大家讨论，或许曾在敏捷项目中的许多人也都遇到了，那就是出现了一些质量缺陷没有来得及在迭代退出前完全解决 的现象。那是不是说明测试不能够退出呢？笔者的回答是“不”。迭代的验收时间即是迭代退出时间，也是测试团队必须退出的时间。在实施中，我们将这些来不及 解决的质量缺陷分成三类，一类是“希望能够解决”的质量缺陷，这部分未解决质量缺陷将要成为新一轮迭代的“将做事宜”，甚至可能作为新一轮迭代的需求做到 Backlog 里。第二部分是“必须解决”，这部分因为直接关系到最基本，最关键的功能，这样的质量缺陷必须被立刻解决，否则就必须承认本次迭代的失败了。第三部分是 “不再重要的” 质量缺陷，这部分质量缺陷可以因为技术的不可实现，对客户产生较小的影响或者考虑到不久因为代码重构，这样的问题不在存在的理由，经过团队和 STAKEHOLDER 的批准可以置成“推迟解决”，待日后解决或者定义到产品的版本说明文档中。

回页首

结束语

在本文中，我们承接了敏捷测试最佳实践系列文章的第一篇，将前文的敏捷原则融入到敏捷测试的实践活动中来。本文所讲的实践围绕“人”和“过程”两个重点展 开，指出敏捷活动的主体仍然是“人”，是“敏捷团队”。笔者给出了一种基于 Scrum 敏捷开发模式下的敏捷测试团队组织结构。接着，讲述了这支团队的敏捷测试“过程”。因实践本身是对原则和方法的具体定制，不同的实施背

景，敏捷活动的具体 表现形式各异。所以，本文始终倾向于提供给您一个可以借鉴的并已经实施成功的敏捷测试“过程和方法”，并与您分享实践经验。

本文为敏捷测试最佳实践系列文章之二，帮助您更详细的了解了敏捷测试的具体实施。而关于如何将一支传统测试团队发展成为一支敏捷的测试团队，在敏捷实践中又有可能遭遇哪些困惑和问题？笔者将在本系列的最后一篇文章“向敏捷测试转变”继续为您讲解。

回页首

附加说明

免责说明

本文不带有任何明示或暗含的保证。文章提供的建议或最佳实践只作为一般的经验分享，只在作者的特定环境下验证过。作者不保 证这些建议或最佳实践在任何情况下都有效。本文的内容有可能不太准确或包含错误，作者对此深表歉意。本文中任何带有主观性的陈述都只代表本文作者个人的观 点，不代表 IBM 公司的官方立场。相关细节，请直接咨询作者。

敏捷测试的最佳实践，第 3 部分：向敏捷测试转变

IBM 软件开发中心的敏捷测试实践分享

谢 明志 (xiemz@cn.ibm.com)， 高级软件工程师， IBM

简介：

本文讲述了作者在两年的敏捷测试和开发工作中的经验和体会。从敏捷的实质，敏捷测试的方法和过程，到如何帮助传统团队转变为敏捷团队做了详细阐述。本文是系列的第三篇文章，着重讲述如何帮助传统团队转变为敏捷团队。

查看本系列的第一篇及第二篇：

- 敏捷测试的最佳实践，第 1 部分：敏捷的实质
- 敏捷测试的最佳实践，第 2 部分：方法与实践

查看本系列更多内容

标记本文！

发布日期： 2008 年 6 月 30 日
级别： 初级
访问情况 905 次浏览
建议：0（添加评论）

平均分 （共 2 个评分 ）

引言

简洁，轻量的敏捷开发模型是为了提供给软件开发团队一种迅速应对客户需求变化，能够高效完成项目工作，降低整体风险的开发模式。敏捷的测试也是服务于这个目标的测试团队对测试工作的敏捷定义。

传统测试模式下成长起来的测试团队要如何转向敏捷，从个人和团队的两个层面又要做出那些转变呢？有什么方法和标准衡量敏捷测试团队的绩效？如何帮助团队的每个人规划正确的发展路线？团队在部署敏捷的过程中又会遇到哪些问题呢？本文主体就这些问题展开论述。

回页首

有关敏捷测试团队和个人的绩效

在我们过去一年多开发敏捷项目的令人难忘的经历中，测试团队携手开辟了一条新的道路，并发展至今。测试团队也曾多次因其突 出的能力，积极的态度以及卓有成效的工作成果屡受嘉奖。今天我们仍然在思考怎样做好敏捷测试，因为我们仍然遇到更新的问题，我们在积累成熟经验的同时也在 不断尝试改进原有方式和突破对新问题的困扰。在这里，我们的实践或许因基于幸运的历史背景和人文环境，能够基本成功的部署敏捷，但我们对敏捷测试在整个软 件开发过程中的角色定位，和职责的理解，仍然对将要采用敏捷测试，以及感兴趣于敏捷的测试团队，和对那些需要从传统测试转变到敏捷测试模式的团队起到参考 作用。

“如何看待敏捷测试和对测试人员做绩效考评呢？”——敏捷团队中无论开发还是测试都不是个人的开发和测试，这是团队的工 作。一名好的测试人员除了能够做好本职测试工作外，表现为愿意并能够做超出原有范围的工作，能够并愿意帮助团队其他成员解决其他复杂问题，实现团队的共同 目标。测试人员能够主动发现并弥补团队中的重要缺失的环节，帮助团队其他成员完成工作任务。

测试团队的职责也从仅仅发现问题的工作中向着眼于整个项目质量保障转变。因此不难得到结论，在敏捷团队中，优秀的测试人员 身上有其他成员的影子。在时间紧迫的情况下，他能转变成其他角色，做出更多的创造性的成绩。充分地发挥了个人战斗力，在帮助他人的同时，自信的态度和各项 工作中的技能得到增强，自身也可以获得更大发展。

在一次关于敏捷开发、测试经验交流中，我们认为敏捷测试团队更需要其他团队的协助，在设计测试用例时，团队的设计人员应该 帮助测试团队设计测试用例，并帮助测试团队做更多的面向客户环境的真实测试。开发团队也要确保开发任务的按时推进，和测试团队保持紧密的合作，在测试任务 紧要时，也能够转变其职能帮助测试团队完成测试任务。

回页首

测试人员向敏捷转变所需要的技能储备

这引出我们今天要探讨的一个问题，测试人员如何在技能上做好准备以面临敏捷开发的全新挑战。我们认为，一名优秀的敏捷测试 人员，需要有较强的学习能力，至少有主动学习的意愿。除了需要了解各种类型测试以及各种测试工具，测试技术外，也需要了解项目中软件设计模式，软件语言， 以及项目的程序组织架构以便建立和团队的共同语言空间。

因为敏捷团队是一个高度协作的团队，在这样的团队中工作需要很好的沟通技巧，语言能力和协作能力。

除此之外，团队的每个成员，都应该认真学习并努力寻找适合自己团队的敏捷模式，并沟通以达到团队成员对敏捷开发模式的统一认识，使得团队其他成员对自己工作的充分理解和建立起成员之间的相互信任。

回页首

测试人员向敏捷转变所需要的方法

培养好的敏捷测试人员，需要培养其技术能力，也需要用正确的培养成员的敏捷思想。敏捷的方法指导敏捷团队行动，是敏捷测试 原则的实践。从一开始，就深刻影响着团队中每个人。当然，方法不是放之四海皆准的，需要团队对敏捷原则深入理解，执行敏捷测试实践后逐渐形成的规律。而一 个传统的测试团队，在固有的行为规律下，在成熟的产品线里，或者层次分明的复杂组织结构里如何做好向敏捷的转变呢？似乎这种改变给许多人带来希望的同时伴 随着一点恐慌？我们有没有可行的策略、方法可以遵循呢？可否让团队又能够发挥在传统开发模式下的力量集中的优势，又能够做到敏捷的随需应变呢？回答是肯定 的。

在做转变的实施前，我们需要有心里准备，任何从传统开发到敏捷开发转变不可能一蹴而就，自然也没有人能够将一个传统开发模 式下的测试团队一夜之间变成彻底的敏捷。对这些还没有敏捷起来，但仍然以此作为目标的项目团队我们建议循序渐进，基于笔者的亲身体验，提供以下实施的方法 请大家参考。

首先我们建议采用迭代的开发模式作为向敏捷的模式转变的起点。很多传统开发模式或者基本上还是瀑布式的开发，或者是周期性 的瀑布式开发，这些都不是敏捷的迭代。敏捷的迭代是高度的迭代，不是瀑布开发的不断累加。换句话说，传统开发是传递性的工作，一方完成，另一方接手。而敏 捷活动的迭代行为更强调尽早开展各项活动，从迭代的一开始就协同工作，共同实现团队迭代的目标。而一旦抵达迭代的周期中最后一个工作日，此迭代宣布退出。

当完成了向迭代活动的转变完成后，接着，我们开始寻找项目过程、管理、执行中最紧要的问题，并使用敏捷开发中的最佳实践来 一一解决这些实际问题。也许，一开始这个过程是很缓慢，而且很难做到一步成功，但是必须通过不屑的努力和足够的耐心，慢慢转变团队的固有思维方式，并最终 努力获得团队对改进后结果的统一认可。而一个问题被解决，或者不再是项目中最严峻的问题时，我们应该开始寻找下一个待解决的困难了。重复这个过程直至成功 的将团队中有悖于敏捷原则和实践的过程和方法调整过来，同时将正确的思路和方法带给团队。

在最近的几次与其他敏捷测试团队的讨论中，我们同时了解到许多软件开发项目中的测试团队遇到过类似的一些问题，如开发团队 没有做单元测试或做得太少，继而在开发过程中的遗留了大量质量缺陷和频繁的回归现象。这使得测试压力急剧加大，测试过程严重受阻，甚至影响到整个迭代的退 出和项目的输出结果等等。又或者传统的开发中的测试团队因为很少有条件去认识客户，了解和实际用户相关业务需求。测试脚本和用例的设计只是基于开发人员撰 写的功能说明。因此，难以做到对需求变化做出快速反应。经过讨论，我们推荐给对这样的团队如下参考方案。

- 首先开始采用测试驱动开发（Test Driven）。

开发人员首先要善于使用测试驱动开发方法写每一行代码，先写测试脚本后写代码，并反复使用单元测试脚本验证所写代码的正确性。

1. 列出需求
2. 为需求撰写一个单元测试脚本
3. 执行测试确信测试结果是失败的
4. 然后，写上仅仅足够的代码以使得先前的测试可以通过
5. 当所有测试通过了，便可以开始写下一个测试脚本
6. 针对需求有效的实现所有测试脚本

另外，当需要代码重构时候，也应该先重构单元测试脚本，在改动代买之前同样先改写测试脚本。

- 尽早的开始测试，开始系统测试，不要等待到功能完全做好才开始。

了解计划中的待实现的功能，了解其权重分配，设计系统测试和功能测试用例。测试执行的一开始可以是针对部分功能的，之后可 以逐步扩展。接着开始采用迭代的过程完成测试任务，即将测试任务划分为多个周期，一开始可以做些关键的功能性测试，可以对代码中的可复用部分（组件，构 件）做完整的安全测试，

性能测试，压力测试，并发测试，全球化测试等。接着的迭代周期可以做边缘化的功能测试和其他测试，最后的几个迭代应该用于回归测 试，和关键的性能和稳定性测试。

然后策略性的进行自动化测试，设计并开发可以用于日后回归测试(Regression)和用户接收测试 (Acceptance Test)的自动化脚本，持续维护与开发这些脚本。自动化测试为团队带来的是长期效益，自动化测试的开发也应该首先选择部分测试对象，例如，API，框架 等比较稳定和关键的功能做功能测试的自动化；对产品的性能指标，压力测试也要较早的制定自动化测试的计划。

最后，要学会做静态测试，做好需求分析，做好对设计逻辑的分析。测试人员要更多的思考需求的可实现性，将自身作为第一用户积极参与项目和系统的需求分析，设计和开发。积极地参与前期工作，并迅速反馈给设计和开发其静态测试结果。

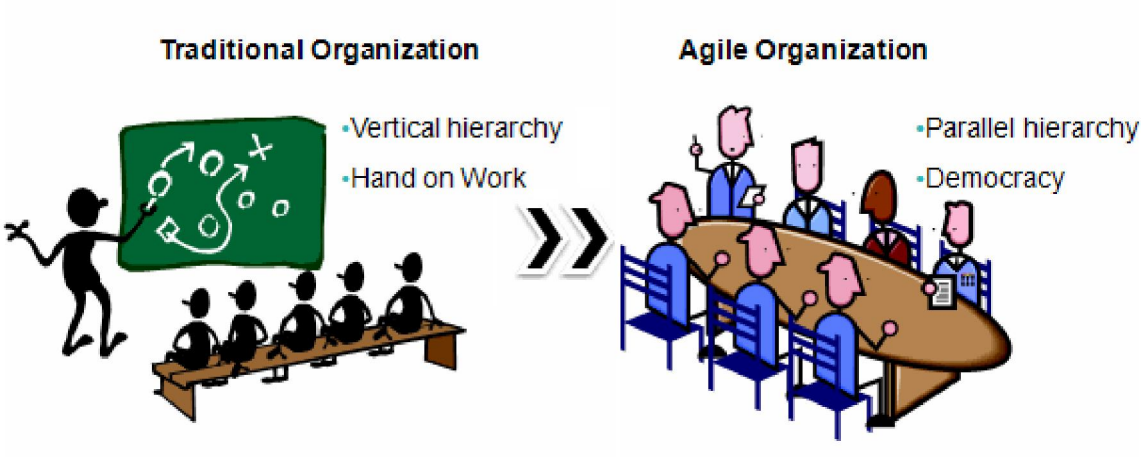
而且，要做好敏捷测试，我们需要转变测试等待开发的思想，测试人员需要了解开发，需要读懂代码，才能够更好的帮助开发人员分析和分离复杂问题。有甚者，测试人员可以成为开发人员的后备力量。当团队中需要更多的人撰写代码时，测试人员应该勇当其职。

回页首

团队组织的变化

敏捷的测试团队在实战中常常不需要其他人帮助做计划和分配任务，成员在各自的敏捷团队中自我管理模式下制定可行的计划与自 行分配任务，并直接报告给项目的管理层。只有在特定的集中测试模式下才需要通过测试团队的领导者形成整体的测试计划和报告。因此，敏捷测试团队是一支即能 独当一面又能够默契合作的适应力，灵活性很强的团队，这正是团队自我管理的核心思想。因此，转变到敏捷开发模式，传统组织结构也需要经历一次调整，调整围 绕两个主题，第一，团队充分授权各成员，对于如何交付结果，成员可以保持较大的灵活性，但成员自身需要对这些结果负责。第二，团队领导者需要协调团队成员 对资源的共享和竞争，当敏捷团队各自计划和实施其工作时，团队领导者应该帮助团队成员建立比较清晰的责任界限，团队角色划分，协调团队中各种资源的使用， 鼓励团队之间的相互交流和协作， 帮助培养团队成员对其所属的自主决策能力。

图 1. 传统组织结构转变为敏捷组织



团队需要建立良好的鼓励机制，这里我没有用激励一词，因为我们认为团队成员已经饱受着来自自身、外部的各种压力和刺激。团 队要在这种高压的环境里保持高昂的斗志，一定需要更多的鼓励和关心团队中的每一个成员。并且，通过鼓励团队的创新，尊重团队的多样性，培育各种想法，使得 团队自身更加善于思考，高效的达到预期目标。相反，处处压制和迫使团队成员按自己的方式做事情，将使得团队的活力降低，生产率下降。

当然，在这里我们更要需要关注团队的核心人物——团队的领导者，在敏捷团队中，团队的领导者更多是一个实战经验丰富，能够 独立完成自身所承担的敏捷项目组工作外，也能够辅导和帮助其他团队成员做好各项工作的重要角色。优秀的团队领导者往往会成为主导项目成败的关键，然而一个 不能够很好转变固有思路，自身都不能做好敏捷测试所涉及的工作的所谓的领导者是自然没有能力去完成培养和发展自我组织，自我管理的团队的艰巨任务的了。很 显然，越来越多的传统团队的领导者具备了领导者和管理者的双重身份，然而，敏捷的团队中，并不需要这样的双重身份。而团队民主，平等的建立才是团队领导者 和团队所有人需要尽力保护和推崇的原则。而敏捷团队中的领导者更多是团队的领军人物，他（ 她 ）明确下一步要做的事情，勇于挑战新事物，不怕承担责任，具备正值，公正，协调能力强的特点，更重要的是他（她）是团队中的模范，他（ 她 ）的影响力，正是领导力的源泉。因此，如果传统测试团队需要发展成为敏捷测试团队，那么团队领导者的职责和形态的转变也必然十分重要。

最后，团队用人和人才配置仍然需要慎重考虑，换句话说，就是因人而异的安排工作。举个例子，在一个分布式的敏捷团队中，集 中在中国的测试团队需要划分出几只小团队分别服务于不同的敏捷开发团队，其中有美国、德国、日本三个实验室参与到这个项目中来，面对时区差异，项目所需技 能差异。我们建议在没有更好的办法的情况下，选择了体力很好的同事在有时差 12 个小时左右的敏捷团队中，将学习能力强的同事放到新技术含量最高的敏捷团队中，代码经验丰富的同事放到做底层架构的敏捷团队中，这样来以达到测试团队的最 佳组合与配置。

回页首

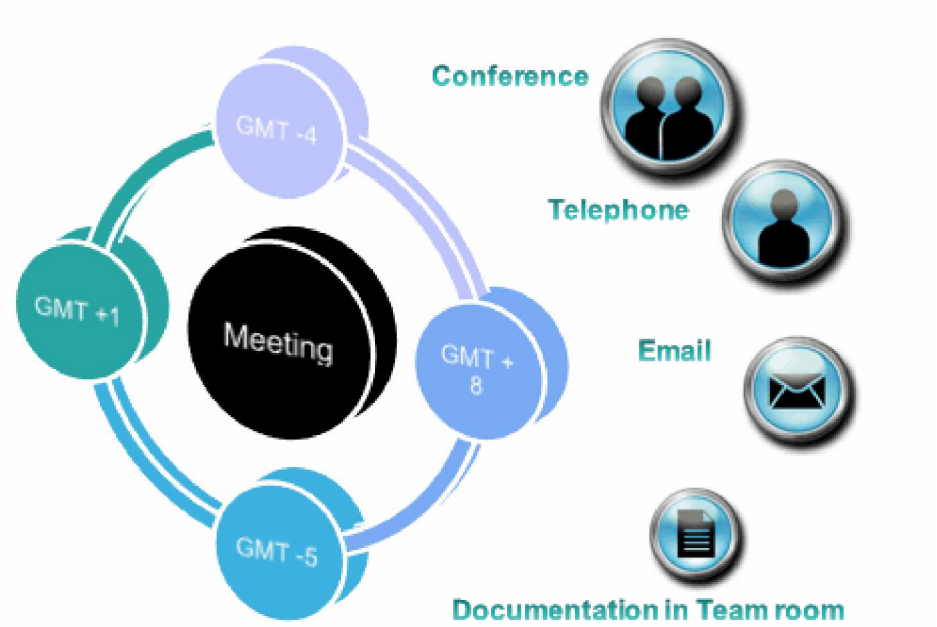
敏捷测试遇到的哪些问题？

任何方法都不是百分之百正确，部署敏捷原则的测试团队仍然会遇到了许多困难。需要我们勇敢的面对和并痛痛快快的解决他们。这里笔者给出了一些被大家讨论得最多的热点问题，结合实际经验分享一些解决方案，希望可以有所帮助。

时区地域的差异增加了沟通成本

某些著名企业在部署敏捷开发项目时曾不允许将同一项目的开发团队跨地域分离，原因是为了方便团队成员间的，团队间的沟通和 交流，降低成本。而在面临外包所带来的巨大利润诱惑下仍然存在许多跨地域，跨时区的敏捷项目团队。部署敏捷开发时的常有人质疑这种方式下的敏捷是否能够真 正成功。也不乏听到源于时区地区差异造成的沟通不便而引起的声声抱怨。

图 2. 时区地区差异增加沟通成本



不得不说地域和时区的差异带来了团队沟通的额外的开销，但是恐怕这也是短期内无法改变的事实，在不能改变环境的前提下，我 们选择了改变自身。因此，敏捷团队更需要通过各种方法保障沟通的通畅，做到沟通即有效。因为不能实施面对面的讨论，所以为了更好的交流，我们建议采用会议 以外的方法，如电话，即时通讯，邮件来弥补时区地区差异的障碍。在我们的深刻体验中，建立起通畅的即时通讯，和信息共享的数据库空间是项目成功的不可或缺 的因素。而团队成员之间经常的感情交流，更能够促进团队间的相互信任和润滑之间的摩擦从而加快沟通。

除了鼓励团队的自我管理，自我建设外，在初期帮助分布在不同地域的团队间建立一些官方交流通道是非常有帮助的。例如，我们 建议对这些跨区域、分布式的敏捷项目在项目初期就找出能够被双方甚至是多方接受的稳定的定期的会议时间和其他有效沟通方式，也建议建立起活动经费经常用于 组织团队之间的各项增进感情交流的活动。另外，帮助团队的成员依据团队整体工作时间的最佳的安排和改变个人的正常工作时间的也或许成为必要的选择之一。

不会休息的团队

敏捷开发中因为是高度迭代，周期短，任务紧，而自身的积极工作态度更使得团队成员不愿意休假并长时间的高度紧张的工作，甚 至频繁的加班加点。在我们的项目中，中国人更加表现出积极的一面。在过去的一段时间里，中国软件开发实验室的很多人中，当然也包括我们，倾向于牺牲个人时 间来满足分配更多工作的要求。从感情上我佩服这种孜孜不倦，无私的工作态度，更惊讶这样的旺盛的精力。然而，很快我们发现我们错了。

这样长时间缺乏休息的团队成员工作效率，工作状态的在四五个月后发生了变化，也突然发现他们一个接一个的开始变得萎靡不振，身体健康状态也随之变坏，以致后来很多人甚至卧病在床了。而这时恰恰又是项目中后期产品开发的重要时期，项目因此承受很大风险。

不会休息的团队是不健康的团队。其实，项目往往不是短期行为，通常一个产品的发布需要经历半年以上的不懈努力和投入，而长 时间的超负荷运转会使得工作效率和员工身体透支甚至可能导致难以控制的严重后果。所以，如果你的项目还要长期发展，应该帮助团队认识到轻松的团队氛围，张 弛有度的工作安排是保障项目最终按时交付的重要因素。

而造成团队不能休息和压力过大的原因，恐怕也是团队自己，为什么这样说呢？原因是敏捷开发团队在计划每一个迭代任务时，许 多团队成员对任务量和计划外突发情况估计不足，导致一开始就承诺了过多的项目任务。因此，团队成员除了加强对这场耐力赛跑的认知，和建立正确的计划外，敏 捷团队的管理者，有义务和责任帮助各个团队建立起适量的工作计划，动态调整团队的工作任务，以保障整体的稳步前进。关注团队压力承受能力的提高和合理分配 团队的工作计划，时间，从长远看来，对团队成员的工作效率和团队的工作绩效的提高具有非常重要的作用。

团队间的协作

越大型的项目，敏捷团队的体积可能越大，迭代的次数越多，产品的架构也更容易产生变化，设计的复杂度也更大。因而，我们需要更加重视产品架构建设，代码重构策略和加强团队间的协作。

最好的设计来自团队而不是某个人，无论是代码还是架构设计还是测试方案都很大程度的依赖于团队的共同承担。而实际项目经验 告诉我们，敏捷模式下，往往因为各个团队具有较为独立的活动能力和决策力，团队成员通常更关注所属团

队的责任范围内工作，无论是开发人员还是测试人员都潜 意识的将依赖于其他团队开发和测试的工作放到靠后的开发周期，寄希望于所有需要的前提和依赖条件最终一定得到解决，而那时后再开始这部分工作也不迟。因 此，这种弱势的团队协作成为项目进度不能保障的罪魁祸首。

在我们项目中，也曾经因为某些组件间接口定义时没有做好统一规划，以致产品整合阶段测试和开发进展非常缓慢，回归现象频繁 出现，团队的士气受到了极大的伤害。因此作为敏捷团队，特别是敏捷测试团队应该更早的暴露接口缺陷，来设计适量的测试用例覆盖这些耦合部分的活动将为产品 的整合带来更小的风险。帮助开发人员尽早认识到其后果的严重性，项目将最终受益。而敏捷原则中提到的不断的整合的思想正是我们克服这个困难的最佳实践。

除了项目管理层通过干预手段解决这部分问题外，鼓励团队成员主动承担额外的工作也是做好协调团队间工作，降低总体风险的最佳途径。

回页首

结束语

在这份共分三部分的系列文章中，笔者介绍了如何理解敏捷，敏捷测试，以及分析了笔者亲身经历的一个成功的敏捷测试案例；并基于敏捷开发原则，拟出了一套可供大家借鉴的敏捷测试原则和方法。

本系列的最后一篇文章介绍了传统测试团队向敏捷测试转变的条件和方法。并分享笔者在敏捷测试的实践过程中遇到过的几类问题和解决这些问题的主要途径。

而这里，我们应该指出并不是只有敏捷了，你才能将项目做成功。敏捷项目要有诸多条件和良好的环境去培养，例如“优秀的敏捷 主义者”，“民主的组织管理制度”，“灵活的产品体系”和“创新的思维”。因而，不是所有的项目都能够，或者有必要转向敏捷测试，敏捷开发。敏捷测试也不 是万能解药。

最后，笔者祝愿勤于敏捷测试实践，并有意转入敏捷测试的团队早日成功。

回页首

附加说明

免责说明

本文不带有任何明示或暗含的保证。文章提供的建议或最佳实践只作为一般的经验分享，只在作者的特定环境下验证过。作者不保 证这些建议或最佳实践在任何情况下都有效。本文的内容有可能不太准确或包含错误，作者对此深表歉意。

本文中任何带有主观性的陈述都只代表本文作者个人的观 点，不代表 IBM 公司的官方立场。相关细节，请直接咨询作者。

参考资料

- 查看本系列的第一篇：敏捷测试的最佳实践，第 1 部分：敏捷的实质。
- 查看本系列的第二篇：敏捷测试的最佳实践，第 2 部分：方法与实践。
- 查看文章 Agile Testing Strategies 了解 Scott W. Ambler 的敏捷测试方法论
- 查看 《敏捷宣言》： *Agile Manifesto* 。
- 参考 *Benefits of Agile Development* ，了解传统开发和敏捷开发的对比
- 参考 Agile Survey Results Download，了解更多业界敏捷开发
- 查看文章 *Agile Methodologies* ，了解各种敏捷方法。
- 访问 developerWorks 中国网站 Rational 专区，了解 developerWorks 上更多的敏捷最佳实践。

关于作者

谢明志，2004 年加入 IBM 中国软件开发中心，拥有 4 年的软件测试经验。2005 年加入 IM 部门敏捷项目开发测试团队，2007 年加入中国软件开发中心 Test Community 开始推广敏捷开发，敏捷测试。