敏捷开发技巧指南

敏捷开发技巧指南

敏捷开发是一种以人为核心、迭代、循序渐进的开发方法。关于瀑布方法和敏捷方法的分析已经探讨过多次，但瀑布方法在某些项目和开发团队中还存在价值。《敏捷宣言》声明指出，个人和交互高于流程和工具。由于开发项目的利益攸关者已经变得越来越分散，遍布在全球各地，甚至经常横跨了几个时区，基于云的开发环境已成为必备之选而非锦上添花。TT SOA 在这本技术手册中将介绍敏捷开发的一些技巧以及瀑布方法和敏捷方法的对比，同时还涵盖了云对于敏捷开发所起到的作用。

瀑布方法和敏捷方法

“敏捷”一词早已在软件项目管理的世界中成了流行语。流行语最显著的作用是提醒人们注意新概念，就改进现有流程和方法的许诺。敏捷被捧为项目管理方法和过时的瀑布方法后的一个大事件。然而，瀑布方法在某些项目和开发团队中还存在价值。

- 瀑布 VS. 敏捷方法的利弊分析
- 瀑布方法如何利用敏捷价值观？

敏捷开发技巧

什么是最佳的敏捷方法？回答这个问题有点像解释在不知道什么工作的时候为这个工作选择什么样的工具。所有的敏捷方法都共享了一些相同的原则和实践。那么如何选择最佳的敏捷方法？生命周期管理工具是敏捷的必需品吗？为什么敏捷流程能减少分布式软件开发问题呢？

- 为你的开发选择最佳敏捷方法
- 为何混合瀑布式/敏捷流程能减少分布式软件开发问题呢？
- 生命周期管理工具是敏捷开发的必需品吗？

敏捷和云

软件应用和现代业务模型如此紧密地交错在一起，现代业务模型开发需求更是空前高涨。但是挑战也随之而来，尤其是对于地理位置分散的团队更是如此。如何使用云为地理分散的团队提供更多的好处呢？

- 敏捷需求管理：利用云的六大优势
- 如何用云改善敏捷测试？

瀑布 VS. 敏捷方法的利弊分析

“敏捷”一词早已在软件项目管理的世界中成了流行语。流行语最显著的作用是提醒人们注意新概念，就改进现有流程和方法的许诺。敏捷被捧为项目管理方法和过时的瀑布方法后的一个大事件。然而，瀑布方法在某些项目和开发团队中还存在价值。

要了解敏捷和瀑布之间的不同，首先要了解（或者重新认识）项目的管理周期。假设一个项目，要建设操场上的秋千。项目管理周期通常描述为，初步搜集客户的需要，来描述秋千的摆动。项目经理注释，在秋千上还要有一个舒适的扶手椅。最终的需要很可能会因秋千缺少一些关键特性而结束。

虽然上面谈论了一个项目从开始到结束的生命周期——项目从点 A 到点 Z 的实际议案通常遵循瀑布方法。典型的瀑布方法通过计划、构建、测试和部署——对于需求而言，任何晚期破坏性的改变将会是围绕业务的一项工作或者被迫对进行这个改变要求的工作支付一大笔钱。

- 典型的方法在简单的项目开中已经使用并且行之有效
- 涉及到一系列路径或者阶段，必须要按顺序完成
- 在完成前要收集所有的需求
- 项目进行到了后期，改变需求变得更加的困难、昂贵

推荐：持续时间短的项目（6 个月内）应具有明确的目标和保证金的存在

随着项目变得越来越复杂，为了支持越来越多由金融市场和政府规章制度引起的业务环境变化，客户要求 IT 快速响应他们变化的需求。

敏捷的出现。敏捷是一个较新的项目管理方法，以解决瀑布涉及的问题为目标。

敏捷的特性：

- 低开销的方法，强烈价值和原则，而不是过程
- 项目的重点是在一个连续生命周期内重新评估，如一周、一个月、或者比较久的时间
- 尤其是有利于需求经常变化的小团队
- 允许在组件中进行产品测试

它不是：

- 一种避开文档的方法
- 一种鼓励更多变化的方法
- 推迟计划的方法

尽管敏捷有好处，可以更快的交付关健的业务，但是大多数据 IT 机构涉及到敏捷项目执行时，会遇到无数的问题。常见的问题是,开发者有时会采取放任的态度，没有成功交付 IT 项目的严谨性。

敏捷不会适合于每一个人，瀑布也是。这引出了一个问题 - 正确的项目管理方法是什么？

就大多数事情而言，多数人都会选择一个中间立场。最近出现了一个敏捷、瀑布混合版 - 项目经理要严格要求自己，最后使业务和 IT 机构都有一个满意的结果。

（ ：

原文链接：http://www.searchsoa.com.cn/showcontent_50282.htm

瀑布方法如何利用敏捷价值观？

您觉得一个团队是否可能遵循敏捷价值观的同时，使用传统方法，像瀑布方法，这样这个团队还是敏捷的吗？

我个人更喜欢 Elisabeth Hendrickson 对于敏捷定义的解释，我觉得这是最精确的。敏捷开发意味着频繁交付业务价值，以一种可接受的步伐，至少每季度一次。

我有点不爽的是“瀑布”已经成为“完全不正常的软件开发”的同义词。瀑布是软件开发中一套定义清晰的阶段，每个阶段之间都有一个“gate”来确定项目是否能够进入下一个阶段。

九十年代初期，我在一家企业工作，用的就是瀑布流程。然而，我们的文化强调质量。所有的角色，包括程序员、测试员、分析员、产品经理、培训专家和技术作家都参与到整个产品生命周期中。程序员进行代码复查，用接近 100%的覆盖范围进行自动化单元测试。测试人员自动化 GUI 级别的回归测试。我们有足够的时间进行探测性测试。我们在测试中用到了很多今天敏捷团队所用的实践，像持续集成和自动化部署。偶尔，我们会在周六工作（上至副总裁的所有管理者都参与），但是我们从来都不是一只“死亡军队”。

最终，我们发布的产品代码只有少量瑕疵，我们的客户很满意。然而，我们一年中只发布一次或者两次产品。尽管人们在协作中角色不同，我们还是会关注自己的工作。我不是说我们要采取一种真正的全团队方法来保证质量。然而，在产业环境中，我们所做的确实奏效了，我们不需要做敏捷，但是我们也很敏捷。

当我为一家软件开发组织工作的时候，却有了不同的体验，他们想要敏捷，每两个周就发布一次产品。没有自动化的单元测试，没有可持续步调；程序员每周例行公事地工作 60 到 80 个小时。

我的测试团队有效地利用敏捷价值观和原则，在探测性测试和自动化 GUI 回归测试上取得了不错的成果，此外，在编写面向客户的测试时，程序员对于驱动开发很有帮助。然而，我们不能在代码内测试质量。软件照例失败了，“个人英雄主义”成为主流，哪个幸运的程序员发现了 BUG，推动了产品进展，就会得到奖励。最好的程序员精疲力竭，安装时的技术债务拖累了开发团队，企业最终彻底失败。没有质量和可持续步调的频繁发布不等于敏捷。

如果你在一个传统的 phased-and-gated 的开发环境中工作，别担心什么是否符合敏捷定义，但是要让敏捷价值观和原则来指导，在合适的地方使用敏捷实践。用不同的方式实验来改善你们的工作，减少技术负债。持续改进是所有软件团队都应该追求的目标。

（作者：Lisa Crispin　译者：张培颖　来源：TechTarget 中国）

原文链接：http://www.searchsoa.com.cn/showcontent_54801.htm

为你的开发选择最佳敏捷方法

什么是最佳的敏捷方法？

回答这个问题有点像解释在不知道什么工作的时候为这个工作选择什么样的工具。所有的敏捷方法都共享了一些相同的原则和实践。正确的问题应该是“哪一种最适合眼前的项目？”

无论你追随的是哪一种方法，敏捷方法的使用不只为软件开发者提供了一种承诺。业务专家和开发团队之间的紧密协作，以及频繁的面对面沟通都是任何敏捷项目成功的关键必不可少的部分。要想敏捷，就要积极主动并参与其中。但是我们如何实现这个目标呢？让我们来看一些流行的敏捷工具。

敏捷宣言

2011 年二月，17 为软件编程实践者聚在一起，就如何更有效地构建软件开发了一个宣言、他们并没有在大多数意见上达成一致，但是有四件事情是清楚的，这就是敏捷宣言，所有的敏捷方法都基于此：

- 个体与交互重于过程和工具
- 可用的软件重于完备的文档
- 客户协作重于合同谈判
- 响应变化重于遵循计划

可能由于思想派别的不同，这些原则对于你来说完全陌生。但是实际上它们是敏捷方法所需要的一套工具。其余的就是应用正确的敏捷方法进行这项工作。

极限编程（XP）

极限编程是一种敏捷方法，需要客户（谁来验收软件）和开发者之间高水平的参与。这里的客户通过优先用例中最重要的功能，从而驱动开发产出。开发团队通过持续编程、测试、计划和紧密协作迭代地交付用例。正常运转的软件频繁地被交付，典型的是一到三周。

优点：快速、积极的交付模式；高度协作；最低限度的预先文档化。

- **缺点：**高度严格的纪律方法；要求 IT 部门之外的人员高度专注；要求团队中的每一个人精通 XP 知识才能成功；可移交文档少。
- **用途：**适合由高级开发者组成的小型开发团队，团队成员具有出色的沟通技能，可以同非 IT 人员共事并对其进行管理。

SCRUM

SCRUM 与 XP 相比，采取一种更加粗略的方法来构建软件。它是一种项目层管理和控制迭代工作的框架。在 SCRUM 中，“产品所有者”，我们称之为业务应用所有者，同时和业务和 IT 团队共事，以“产品订单”的形式确定和优先系统端的功能。各种团队成员以短跑的形式，签名交付可装运的正常运作软件的增量，通常持续 30 天左右。

范围蠕变由开发团队控制，所以一旦交付没有任何功能可以再添加到这次短跑中来。一旦可装运增量被交付，产品订单就被分析，如果需要，重新优先化并再一次循环开始。

- **优点：**范围蠕变的出色控制能力；鼓励团队协作和透明度；管理开发缺陷的良好可视性，因此适应性强；可以根据对团队和地理位置进行扩展。
- **缺点：**关注粗粒度性能；有时导致“快餐式”编程；少量文档可交付。
- **用途：**适用于关注于产品的 IT 部门，主要聚焦在产品和项目管理者的协作中交付宽泛的性能。

特征驱动开发（FDD）

FDD 是以开发者为中心的流程，以实现功能为目标。把系统分解成一个一个的功能集，每个功能集又习细分为具体的功能。不像 SCRUM 和 XP，FDD 关注具体的工作单元，这些工作单元通过严格流程审查，FDD 的起点是起源于创建一个全局的模型轮廓（不要求很精确，大概模样就可以），然后是周期低于两周的一系列的“design by feature, build by feature”的迭代，逐渐丰富模型功能内容。

FDD 及时且反复地促进了有形的，正常运转的软件的开发流程结果的严格控制。

- **优点：**出色的文档化；高质量的设计和代码检验；可控制的创建流程；模型是面向类的，适用于面向对象编程语言，像 Java；早期开发流程中，质量度量的良好洞察力；适合扩展到大型团队。
- **缺点：**要求高度的设计和开发技能；如果最初的模型不正确很耗时；项目管理来跟踪项目很耗时；计划耗时。
- **用途：**适用于法规驱动的环境下的团队开发者，像保险、金融、银行和政府，这些地方流程成熟，质量是最主要的需求。

结论

因此哪一种敏捷方法才是最佳的？谁都不是。每一个都对应具体的项目，没有一把万能钥匙。选取哪一种取决于 IT 环境和你所工作的内部流程。改变 IT 文化很难。但是敏捷方法的好处应该是让 IT 团队的生产率进入一个新水平的催化剂。

（作者：Andrew Townsend 译者：张培颖　来源：TechTarget 中国）

原文链接：http://www.searchsoa.com.cn/showcontent_49864.htm

为何混合瀑布式/敏捷流程能减少分布式软件开发问题呢？

Nari Kannan 目前任 V-Soft 咨询公司首席交付官，Nari 在信息技术领域有 20 年的经验，最初在 Digital Equipment 公司做高级软件工程师。他曾服务于业务流程改进，IT 咨询、人力资源和物流应用。

横跨许多大洲和时区，做分布式软件开发是现实的。根据我的经验，在分布式开发环境中，瀑布式和敏捷软件开发方案都有缺点。本文中，我将要演示，如何做一个混合的瀑布式/敏捷模型，并结合敏捷开发模型最好的特性和最好的传统的瀑布式软件中开发生命周期模型。

瀑布式模型软件开发，仍然有很多组织机构在使用，它有好处也有弊端。它做的很好，因为在代码开始写之前，它强调建立许多正规的文档——像适当的需求、功能说明书、技术说明书和技术架构文档。然而它做的不好，因为它的项目交付时间一般都非常长。所以，你经常等几个星期或者数月，在软件交付前，从客户或者是远程分布式开发团队的软件开发经理，发现你一直偏离方向，这已经太晚了。

敏捷开发发生在短迭代中，并在短时间内发布。如果你有一个工作版本，也就是说，每 10 天，或者更早一些，结束交流的分歧。客户得到一个发行版本和一个演示版本一起执行，提供流程修改的反馈，这个反馈应该在这个周期的前期提供。你不需要等几周或者数月，就可以发现软件交付一直偏离方向。用户尽早的看到软件，提供一个反馈，告诉开发者是否正确。

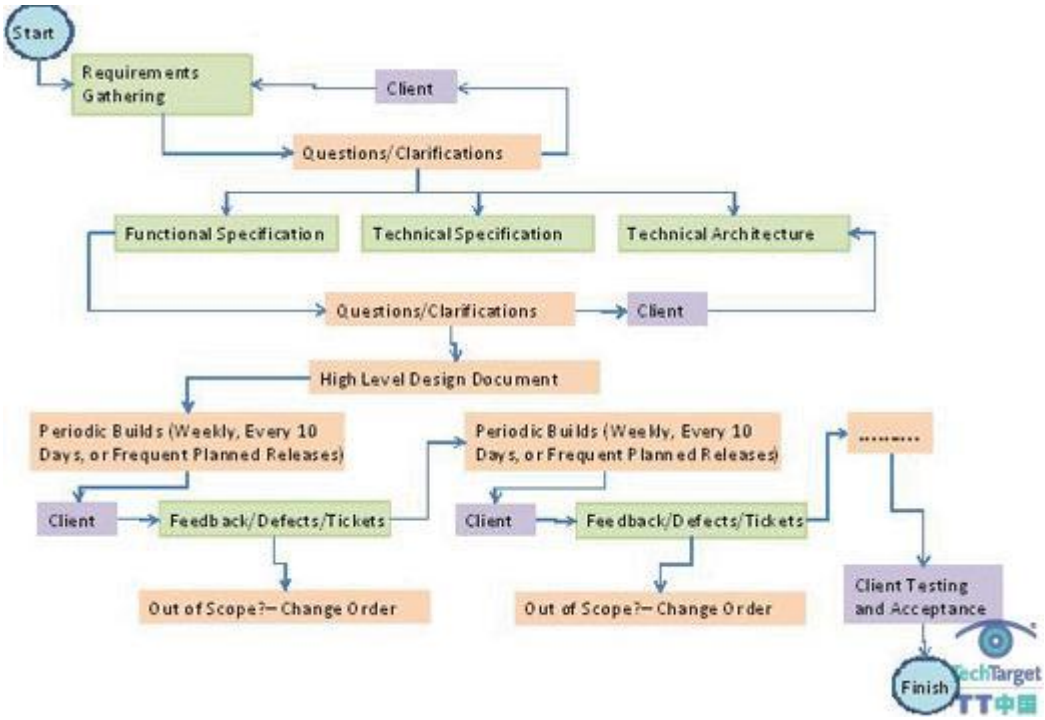
具有讽刺意味的是,在现实中，敏捷的短迭代允许需求改变被看成一个缺点。项目经理、外包公司、CIO 和其他人担心敏捷只会大幅度地增加掌握范围之外的因素，导致范围蔓延。

我发现另一个问题就是纯粹的敏捷开发方法，最好是所有的工作人员都在同一个地方。而且，面对面交流比通过视频会议、电话和即时消息产生的分歧要少。瀑布式开发在初始阶段的时间不是很密集，这可以帮助减少在分布式开发项目中的交流问题。

混合瀑布/敏捷开发的优势

混合的瀑布式-敏捷开发方法可以更好的适合分布式软件开发，尤其是在横跨多时区的开发。这会保证有一个正规的文件流程。也会确保正规的文档流程先有一个主要的预期，但是为了我们更正使之成为好的流程，敏捷开发方法应具有灵活性。最重要的是，我们不能等到项目的最后几周或者几天内，才发现一直在交流上有分歧。

在软件开发的工作中，90%的问题是因为客户和开发团队之间缺乏适当的沟通。如果你在早期就沟通，那么开发会更加顺畅。混合的瀑布式/敏捷方法强调沟通方面都要通过它的周期，如这张图所示。



混合瀑布式/敏捷开发模型精选

上述流程是一个指示性的流程。有些部分可能已经由最终用户的 IT 组执行了。例如，他们可能已经准备了需求文档、技术说明书和功能说明书。或者，他们可能只有一个需求文档。开发团队最先根据情况来选择适合的流程。

混合瀑布式/敏捷流程的关键组件

每个项目都需要一个详细的项目计划

在这个提示中，项目计划没有显示在流程图里，因为，有时你在项目投标之前就需要去完成它。同样，它有时也可能是交付的一部分。这个流程，我用微软的项目，但是，也有一些其它可用的管理工具。

在开发的每一个阶段，为客户列一个问题清单

从 RFP 文档到需求文档，在每一个阶段，开发团队必须生成一个回答客户问题的列表，收集那些他们不清楚或者不知道的问题的答案。更好的得到这些问题，尽早找出答案，而不是对任何事情的一个假设。

有些文件是可选的

例如，功能说明书、技术说明书和技术架构文档都是可选的，只能根据情况来选择并使用。如果你的团队正在拟订一项短期项目，设计网站的一部分，那么建立这些文件可能有些夸张了。在我们开始编码之前，这个流程展示了我们或者客户可能需要的所有文件的超集。

一个高等级的设计文档总是需要的

对于每一个项目，一旦通过详细的计划书阶段，开发团队会建立一个高等级的设计文档;即使它只有几页。这只是演示用户接口设计的实体模型等。做这件事可以确保开发团队可以完成各方面的开发工作，并在编码之前有机会提问，并澄清一些事情。

必须达到每周、每十天或者定期发布版本的最后期限

在混合瀑布式/敏捷模型中，项目计划分解开发和测试的可用时间到每周、每十天 -- 或者最大的期限 -- 每两周建立一个版本的周期。根据这一数据，就会有一个没有错误的版本。

这可能会为工作版本增加一些灵活性。的确，开发团队可以开一个没有客户存在的会议，决定建立一个工作版本；但是这个工作版本的发布不能因为任何理由而推迟。这是建立一个严格的工作和发布版本的原因。每一个工作版本会有一个正规的本地测试。这也保证了软件的测试比瀑布式模型测试的次数多，间接的提高了软件的质量！

确保请求改变是否需要一个变更指令

当从客户那里得到反馈时，建立一个缺陷或者标签的条目，首先测试的是，查看变更是否需要一个变更指令。一个简单的不需要变更指令就能修复缺陷和一个需要变更指令的漏洞之间有什么不同呢？像字体、颜色或者难以理解的简单事情都可以算作缺陷。如果涉及到超过四个小时的工作，就会考虑改变指令。这只是一个经验法则，可以按照具体项目有所改变。

选择敏捷、瀑布式或者混合敏捷/瀑布式

软件开发中纯粹的敏捷和纯粹瀑布式模型工作在特定的情况下，其他的情况都失败的很惨。对于分布式软件开发来说，无论它是外包或是内部使用，混合的瀑布式/敏捷方法提供了世界上最好的东西：在一个半正式的流程开始时，增加敏捷方法的灵活性。范围蔓延是许多软件开发方法的最忌惮的事情，在开始的时候提供一个半正式的规范方法，可以限制范围蔓延，在开发周期中，还允许一个超出范围之外的异常流程。

(作者： Nari Kannan 译者：刘志超 来源：TechTarget 中国)

原文链接：http://www.searchsoa.com.cn/showcontent_37212.htm

生命周期管理工具是敏捷开发的必需品吗？

敏捷团队使用各种各样的应用生命周期管理工具（ALM）。例如，我们可能选择故事发生流程图协助我们收集需求。通常包括很多便利贴以及一面墙或者桌子。我们可能关于模型或者设计使用白板或者大型纸张进行头脑风暴。又或者，我们可能使用更多正式的在线工具来实现。我们在选择这些工具协助我们实现目标最重要的是什么呢？交付高质量软件产品，符合客户需求。

在很多软件项目中，创建和改变管理工具是持续进行的，敏捷项目也不例外。很多敏捷团队已经自动化部署，一些部署持续进行。集成开发环境（IDE）是必须的，我建议敏捷团队中的每个人都使用，包括测试人员。

很多团队需要缺陷追踪，这一点就像任务板上的卡片一样容易实现，或者使用自定义缺陷追踪系统。产品订单可能在索引卡上、电子表格中或者是在线的订单管理工具中已经存在。

产品公司和开源开发社区已经用 ALM 工具加速了设计，尤其是敏捷项目。其中一些集成项目订单管理是通过冲刺或者集成管理和缺陷追踪来加速实现。

除了一些必须的东西，像源代码控制、构造管理和 IDE，我建议你的团队为不同的目标试用不同的工具。找出针对唯一环境的最佳工具。如果你认为你需要一个在线工具，就进行两到三个迭代，评估这个工具是否能解决你的问题。如果不能，就试试其他方法。不要试图同时解决所有问题。最好是设置一个目标，然后一步一步实现，试用短反馈环来确定是否进行正确。

（作者： Lisa Crispin 译者：张培颖 来源：TechTarget 中国）

原文链接：http://www.searchsoa.com.cn/showcontent_52619.htm

敏捷需求管理：利用云的六大优势

《敏捷宣言》声明指出，个人和交互高于流程和工具。由于开发项目的利益攸关者已经变得越来越分散，遍布在全球各地，甚至经常横跨了几个时区，基于云的开发环境已成为必备之选而非锦上添花。基于云的环境，与其说是采纳敏捷需求管理需要克服的又一道障碍，现在不如说它是一项资产。以下就是采用基于云的敏捷需求管理工具和流程的六大优势：

1. 基于云的协作工具中的频繁通讯与异步通讯

大多数敏捷的践行者更愿意把需求简述为故事，而非冗长的文档形式的规范。这些需求，在他们将其融入故事和 sprints 的时候，他们还期望能改变和包含若干的澄清和说明。基于云的协作工具对于敏捷迭代来说成为重要的附加品，在这种工具里面，利益攸关者之间大部分的通讯都是通过日常或阶段性的 stand-up 来进行的。它还以异步的方式创建永久性文字记录，可供所有利益攸关者浏览并添加自己的观点和建议。基于云的需求管理为敏捷开发提供了最重要的元素：在所有利益攸关者中间进行的频繁迅捷的沟通。

2. 基于云的工具可交换模型和框架

尽管敏捷开发的目的是为用户提供一种软件，让其尽可能早地跟 sprints 一起工作来获得反馈和改进建议，在任何需求或故事的最早阶段，不同的利益攸关者对于需求在自己的想法里面可能会持有不同的概念，这导致了沟通鸿沟。基于云的环境帮助团队分享各种模型和框架，对其提出意见和改进建议，甚至在所有的利益攸关者中间过一遍那些未经加工的草图都可以。这些都能够极大地缩短沟通鸿沟，甚至可以发生在一次迭代之内，免除了可能的在多次反复中传播故事的需要。

3. 基于云的测试数据存储和访问

基于云的测试数据存储与访问，如果有可能是在一个软件开发项目当中的话，由于得到开发团队的理解，使得预期需求与需求之间的鸿沟可以被分享、可以对其进行讨论，并最终消除鸿沟。基于云的测试数据与敏捷测试一起工作，有助于消除这些鸿沟，因为产品所有者能够验证自己是否被正确地理解了。

4. 基于云的敏捷需求

在某些使用敏捷方法论的公司（尤其是初创公司）里，业务模型本身能够快速改变，要不就是需求会随着时间而演变。如果利益攸关者在地理上分布各处，那么管理快速变更的最好途径就是基于云的环境。这使得利益攸关者从一堆匮乏描述的故事中抽取出需求来，然后插入若干的新故事，并根据优先级重新组织它们，只针对位于这堆故事最上面的那些添加细节描述，因为这些故事是下一次 sprint 或迭代的铺路石。相对于非云化的环境来说，基于云的环境对这种按需细化需求的方式的支撑要好得多。

5. 基于云的需求及测试数据集中化加速了测试进度

基于云的测试正在快速到来，尤其是在移动应用领域，开发和建模需要测试的应用有可能数以千计。一旦你把基于云的需求和基于云的集中化测试数据结合起来，加上测试环境也是基于云的，使得许多测试和质量保证可夜以继日地完成，可以由分布在各地的人并行完成。这也使得标准化的测试数据的使用可以由利益攸关者加以验证，无论他们身处何方。

6. 基于云的工作流缩短了所有利益攸关者之间的澄清与认可的周期

基于云的协作环境也总是会包含有工作流特性。有了它，可以极大地提高工作流转的速度，使得需求定义和理解的鸿沟可以在尽可能短的时间内被填平。甚至在协作的环境下，讨论区可能都不如代表着工作流的电子邮件那样得到相同的注意及如此之快回应。这帮助敏捷开发组为下一次 sprint 做好准备，在本次 sprint 完成之前即可准备就绪。

结论

基于云的敏捷需求管理工具正在快速演变。相对于利益攸关者可能无法准备好在任何时间都能访问到的一组集中化工具，它们显示出了若干明显的优势，尤其是如果他们在地理上跨越许多个时区的时候。通过云来管理敏捷需求从根本上消除了沟通的鸿沟，那种鸿沟可能存在于定义不足的需求以及会谈之间，而弥补这种鸿沟需要在你看到需求在 sprint 中实现之前发生。将其与基于云的测试数据和测试环境结合在一起，你就可以拥有一个非常强大的敏捷开发机制。

（作者：Nari Kannan　译者：杨华军　来源：TechTarget 中国）

原文链接：http://www.searchsoa.com.cn/showcontent_52888.htm

如何用云改善敏捷测试？

软件应用和现代业务模型如此紧密地交错在一起，现代业务模型开发需求更是空前高涨。但是挑战也随之而来，尤其是对于地理位置分散的团队更是如此。在这篇文章中，我们将找出如何使用云为地理分散的团队的测试管理提供更多的好处。

软件需求仍在持续

对于持续竞争优势的追求助长了全球对于软件开发的需求。对于有效交付开箱即用的软件，要保证最小缺陷，还要在双倍快速的时间中实现，需求成为一种逐步上升的压力。考虑到软件复杂度逐渐增加，所以只有少于三分之一的开发项目一开始会想着能够有圆满成功，这一点也不奇怪。

尽管协同定位团队可能成为有效软件开发的优先解决方案，在传统的非技术性国家中，技术性强、低成本的劳动力逐渐出现，为这种趋势加入了新的动态因素。最显著的在印度，但是东欧等地区也在郑家，现在能够提供熟练的 IT 资源，工作职位也从这个角度流入。

差不多 60%的敏捷测试团队地理分散

在最近的一项调查中，这种趋势被显著标出，报告中显示差不多 60%的敏捷测试团队目前是地理分散的。即使在敏捷团队，相近的工作也是规范越来越少，异常确不断的增多

但是尽管全球化为访问低成本资源和维护动力提供了机会，但是也带来了潜在的延迟和分解。最明显的就是试图利用和管理地理分散的资源，但为标准和敏捷测试方法带来了麻烦，通常我们假设开发和测试团队应该在同一个房间或者园区内。

核心难点：沟通

这项困难的核心点只有一件事情——沟通或者说缺少沟通。这种情况破坏了测试项目中摆脱时间表和注入成本以及低效的能力。

尽管地球村是个美好的想法，但远不是一种紧密的沟通，人们分散在世界各地，母语不同，具有不同的文化，而且在不同的时区工作。将这些因素混合，尤其是敏捷项目的测试管理，立即变成了一项相当复杂的任务，特别是敏捷测试需要在开发旁边运作，是需求驱动的。

数据转换安全性和可靠性复杂化

在分散的团队中，数据转换的安全性和可靠性更为复杂。企业希望其开发和测试团队成为一个体系，就必须扩展其网络，典型的就是远程的虚拟专用网络（VPN）。然而部署专线或者 VPN 到一个发展中国家，那里的基础架构仍在建设，是一种十分昂贵的选择。再加上配置服务器和设置数据库的需求，这就成为一种无法轻易实现的流程。

云端测试管理

这些令人不爽的状况导致越来越多的企业转向云端。公司的团队在全球各地运作，使用云提供的机遇，在需要的时候将遥远的资源带过来，同时去掉了让人讨厌的投资专有沟通基础架构的需求。这种工作方法不仅仅是交付了巨大的时间成本，也去除了在陆地和海上之间运行私有网络的嵌入成本。

此外，管理测试的开始和流程在云端立即变得简单，没有复杂的人员培训需求，团队成员可以简单并快速地通过熟悉的浏览器访问项目。

尽管基于云的系统可能不能应付所有复杂的软件开发，当和便宜的互联网沟通系统，像 Skype，连接的时候，云可以为测试管理提供相当完美的环境。甚至允许团队在 Scrum 的迭代框架中运作：进行短跑冲刺；主持会议；实时公职，最新的燃尽图；以及直接的协作思考和更快的缺陷分辨评论。

云也能很好地适应测试资产的存储。测试脚本、缺陷记录和结果分析都可以存在一个地方，需要他们的人可以轻易访问，不管他们在什么时区工作。项目变得更轻松和高效，在没有副本或者是通过更新时引入错误和遗漏而收到连累。

在一种更为分布式的世界中，相对简单的软件项目可以生成数以百计的测试，能够沟通和访问共享信息是某种巨大的进步。