

08 | 避雷策略：如何防止你的敏捷变为“小瀑布”？

2020-01-06 宋宁

说透敏捷

[进入课程 >](#)



讲述：宋宁

时长 14:19 大小 11.48M



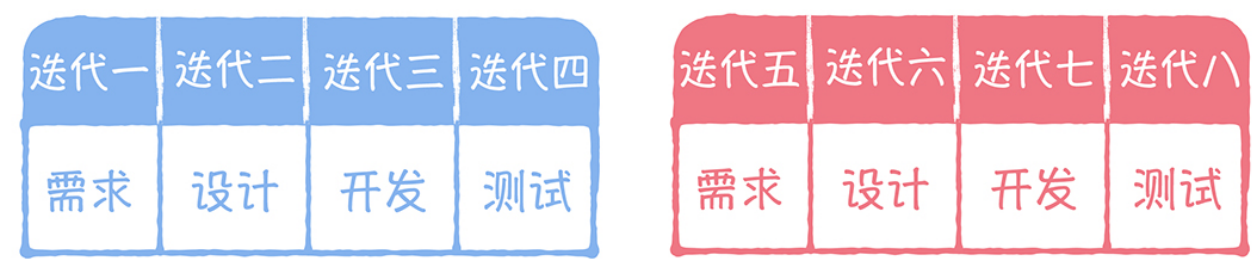
你好，我是宋宁。

也许在一起订阅我们这门课程的人中，很多已经走在敏捷实践的路上。然而却有很多人在做的过程中，不小心走入了敏捷的反模式。那么该如何检视你的敏捷方法是否正确呢？今天，我想以其中的一种反模式“小瀑布”来与你深入探讨一下。

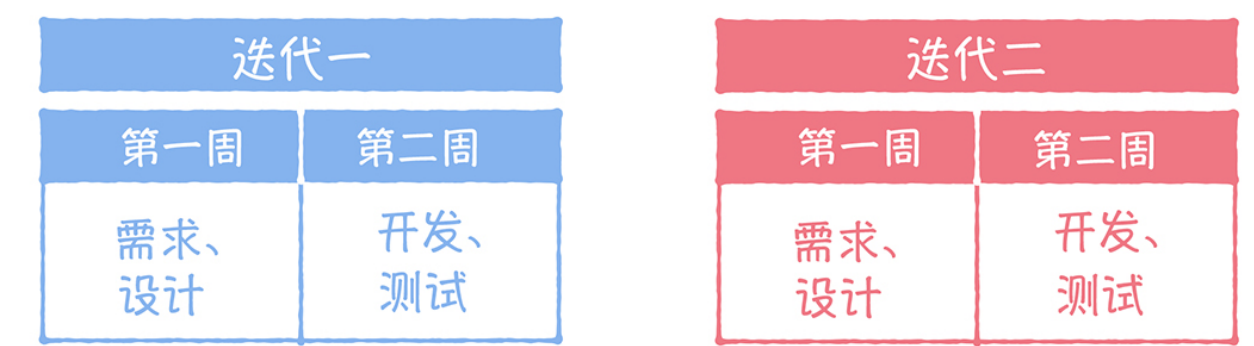
真敏捷，还是“小瀑布”？

在我提供敏捷咨询的过程中，常常会看到一些团队的敏捷实践过程出现下面这些情况：

把一个大项目分成若干个模块，仿照“敏捷”的做法，每四个迭代做一个模块（见下图）：第一个迭代做需求，第二个迭代做设计，第三个迭代做开发，第四个迭代做测试。这样四个迭代交付一个模块的内容，然后开始下一个模块的循环。做着做着，他们发现这种方式比以前的瀑布模式要好一点，但整体的节奏仍然缓慢，“敏捷”好像没那么大的效益。



有的团队觉得上面那种方式还是太慢，于是加快节奏。每个迭代周期也是 2 周，但不同的是，在一个迭代里完成一个大的功能（见下图）。第一周完成需求和设计，第二周完成开发和测试。迭代的次数增加了，但开发和测试却总感觉时间不够，每次迭代都特别赶，体验非常不好。



团队的感觉不好，然而却不知道自己做得到底对不对，就找身为敏捷咨询师的我来诊断。那么上面的做法究竟有没有问题？他们的敏捷活动做得到底对不对呢？

先来看一下他们目前的研发过程，整体来看，他们是按照“需求 - 设计 - 开发 - 测试”这个流程来做的。你是不是觉得似曾相识？没错，这其实就是瀑布模式。

但是他们做了一些改造，也就是引入了“迭代”这个概念。先把大项目或需求做一个模块拆分，然后一个一个模块做下去，和瀑布模式相比，这种方式有了一点进步。然而，究其本质，仍然还是瀑布，我们一般称它为“小瀑布”。

“小瀑布”同样具有瀑布模式的一些缺陷。比如说，很要命的一个问题就是浪费。我们看一下上面那两种做法就会发现，他们在研发中的所有步骤都是顺序进行的，这就意味着做第一步时，后面的步骤都处于等待状态。这样，BA（需求分析师）做需求的时候，开发测试人员在等待；开发人员写代码的时候，测试人员也在等待。这就造成了时间上的浪费。

所以从这个角度来说，**小瀑布依旧是瀑布，它并没有改变瀑布模式的宿命，它离真正的敏捷还有相当长的一段距离。**

那么真正的敏捷是什么样子呢？

以需求为例，团队会尽可能有效拆分需求，这样进入到迭代内的需求就可以是多个独立的小需求。小到什么程度呢？小到每个需求都可以在很短的时间内，比如 2~3 天内完成开发和测试，最长也不要超过一个迭代周期。

这样在开发人员写代码的时候，测试人员在写测试案例，或者在考虑使用自动化测试方案。由于需求拆分得足够小，所以很可能第一个小需求在迭代后的第二天就可以交付测试了，在测试人员测试这个需求的同时，开发人员继续开发下一个小需求，由此形成一个良好的循环。在这种情况下，大家都在热火朝天地工作，节省了很多等待的时间。

为什么你把敏捷玩成了“小瀑布”？

那为什么有的团队会把敏捷玩成“小瀑布”呢？我在接触了使用“小瀑布”的团队之后，才慢慢明白这背后的原因。

第一个原因是**有些团队其实并没有真正地理解敏捷**，团队遇到痛点，他们只了解了敏捷的皮毛，就拍脑袋炮制了一个所谓的“敏捷”流程。

比如说有一个 M 团队，他们团队的工作量很大，以前用瀑布太慢，完不成工作任务。他们领导说：“你们怎么做我不管，这个产品必须在 3 个月之内交付！”然而这些需求却远远超出团队的交付能力，所以当团队听到这个消息时一脸懵圈。团队中有一个机灵鬼说：“我听说现在别人用了敏捷后，交付得很不错，要不咱们也来试试？”此话一出，M 团队像是抓住了一根救命稻草，他们在很短时间内快速地上网查看别人怎么做敏捷，再按照自己的理解，把团队负责的产品按功能大卸八块，又给每个功能都定了小时间点，每个功能用一个小瀑布这么做下去。

他们在这里犯了什么错呢？他们误以为敏捷就是简单地换一个工作流程，只要套用一个不知所谓的敏捷流程就能成功交付工作，而完全忽视了敏捷自身的规律。

第二个原因是**需求太大**。团队知道使用敏捷后，需求要做拆分，也把需求做了拆分，然而由于拆分方法不当等原因，拆完后的需求还是很大，一个 2 周的迭代根本就做不完，或者只能灰头土脸地勉强做完。

比如有个 N 团队，他们每次拆分完需求后，需求的大小都要 2 周的迭代才能做完。这样他们每个迭代最多做一个需求，在研发过程中，也是按照做完需求分析再做设计、开发和测试这样一条线下来，所以虽说迭代频繁了，每一步也还是需要等待时间。

这里最大的问题，你看出是什么了吗？没错，就是需求太大，这导致团队在做敏捷时不能发挥敏捷的优势，将敏捷实践做成了不折不扣的“小瀑布”。

如何避免把你的敏捷做成“小瀑布”？

既然我们分析了问题和原因，那到底怎么做才能避免把敏捷做成“小瀑布”呢？我想针对上面 M 团队和 N 团队的情况，再给你一些建议。

先来看 M 团队，针对他们不了解敏捷就开始着急模仿敏捷流程的做法，首先我们可以给予他们知识上的宣贯，在使他们充分了解敏捷的基础上，对他们进行技能上的培养。

除此以外，不知道你有没有注意到一个细节，那就是他们预计的工作量要远高于他们的产能，这其实也是一个很严重的问题。其实如果工作范围没有发生变化，即便是用了敏捷的方法，或许也很难做到在 3 个月之内做完整个项目。

这是不是意味着这个问题就无解了呢？其实不是。

你应该还记得在敏捷的原则里，有一条是“我们最优先要做的是通过尽早的、持续的交付有价值的软件来使客户满意”，也就是说我们要先做有价值的东西。所以反应到这个项目中，我们其实可以**先把客户的需求拿来看一下，挑选好并先从有价值的、优先级最高的需求开始做。**

如果一定要卡 3 个月这个时间点进行交付，那么就按照 3 个月之内我们能做多少，来选定价值最高的需求优先做，因为在整个过程中我们是不断的把产品交付给客户的，所以我们有

很大可能在 3 个月内交付给客户一个令他们满意的产品。

你有没有看到，在这里我其实是把“3 个月做完项目”这个课题转换成“3 个月做一个让客户满意的产品”，而其实后面的那个课题才是客户最需要的。

所以在敏捷实践中，**我们工作的结束点不应该是把之前所有计划的工作做完，而是把客户需要的工作做完。**这些工作不一定是之前就被纳入计划的，但却一定要是客户最需要的。明白了这一点，敏捷才能被活学活用，而不是被误用成小瀑布。M 团队日后的工作就是按照这个原则来做的，他们的研发工作由此慢慢走上了正轨。

接着再来看看 N 团队，他们的问题是需求太大。这里涉及两个层面，第一个层面是拆分方法不当，导致大需求并没有被正确拆分成小需求；第二个层面是即使拆分方法得当，拆分后的需求仍然很大。

所以作为敏捷咨询师，第一步我先给他们讲解了需求拆分的方法。需求拆分的方法有很多，例如按照业务流程、按照业务规则的变化或按照数据的处理方式进行拆分等等。

你要注意，不管是使用哪种拆分方法，做需求拆分的目的，都是把大需求拆成一个个能够独立开发测试的小需求。只有这样，我们才能在迭代中同时做几个小需求，而不需要等待，并且在测试完成后，这些小需求也能独立上线。

N 团队之前是按照架构的层次来进行需求的拆分，例如把一个大需求拆成了 UI 层的用户故事、逻辑层的用户故事、数据访问层的故事。然而由于这个端到端的大需求一个迭代做不完，所以他们放到了几个迭代中。但是他们发现，每一个故事做完都不能单独测试，只能等着这一连串的故事全部做完以后才能一起测试。另外每一个故事没有独立的价值，也不能独立上线，这样就需要大量额外的等待时间。

在认识到这个问题之后，N 团队重新对他们现有的需求进行了拆分，保证了每个小需求的独立性，之后在两个周的迭代内他们能做好几个小需求。

但是 N 团队发现有的需求经过拆分后还是很大，无法在 2 周内做完，这种情况怎么办呢？这时就需要深度的挖掘一下背后的原因，采用相应的应对策略，

经过细致分析，大家了解到，他们的系统架构比较老旧，代码的耦合度较高，依赖性大，要完成一个需求甚至要改很多个系统，这对于产品交付来说明显是一个很大的障碍。

于是团队计划在现有的开发测试工作之余，对该产品进行架构演化，拆分微服务。为了能顺利开展这些工作，团队用 70% 的时间进行新需求的开发，用 30% 的时间进行架构演进。经过大概半年的时间，N 团队完成了既有架构的改造，拆分了微服务，现在他们的敏捷已经运转得越来越好。

总结

好，现在我来总结一下我们今天的课程内容。

在推进敏捷的过程中，有时候一不小心就会走入敏捷的反模式。综合上面 M 团队和 N 团队的情况，我们可以看到，如果团队只是套用敏捷流程，或没有做好需求拆分，敏捷很容易变成“小瀑布”。

此外，还有一些其他情况也可能让你步入敏捷的反模式，比如虽然导入了敏捷模式，却没有按照敏捷模式要求的角色职责进行人员匹配。举例来说，如果直接让一个技术经理同时扮演产品负责人和 Scrum Master，很可能就做不好敏捷，因为这两个角色要求的技能是完全不同的，技术经理是没有足够的能力和精力来同时担当这些责任的。

但其实走入了反模式并不可怕，重要的是我们一定要保持警惕的心。团队在发现问题后，需要沉下心来分析原因，只有将具体原因找出了，在正确理解敏捷原则的基础上灵活使用敏捷，并制定相应的对策，才能真正发挥敏捷的价值，敏捷才能继续正确地走下去。

思考题

现在，我想请你来思考一下，你在推进敏捷的过程中有没有犯过什么错误呢？我建议你思考一下，并结合今天的内容先自己想一下怎么解决，再来留言区跟我一起讨论一下吧。

点击参与 

打卡10天，带你 从原理到实战落地敏捷



扫一扫参与小程序话题



新版升级：点击「 请朋友读」，20位好友免费读，邀请订阅更有**现金**奖励。

© 版权归极客邦科技所有，未经许可不得传播售卖。页面已增加防盗追踪，如有侵权极客邦将依法追究其法律责任。

上一篇 07 | 填坑指南：填好这4个坑，快速做对敏捷

下一篇 09 | 内部教练：守护敏捷实践，求人不如求己


精选留言 (8)

 写留言



李永智

2020-01-13

做个一个项目，将小瀑布方式当做敏捷来看待，造成这个状况原因有二：第一，对敏捷的理解不到位，当时认为需求、设计、开发、测试还是要顺序进行，虽然这些工作是顺序的，但是开展工作时要做到能同步开展工作，这个对团队人员的要求还是蛮高的。第二，工作能力与工作量不匹配。当时工作量很大，但工期比较短，以当时的人员配比是无法完成的。虽然当时也安排了优先级，不过是按照功能进行拆解的，没有考虑有些业务场景...
展开 



你的美

2020-01-11

敏捷和微服务有点天生就是好兄弟的感觉。说起这个小瀑布，就像那个系统拆分做成了一个一个小单体。



无需昵称

2020-01-11

老师好，针对老师的这句话“我们工作的结束点不应该是把之前所有计划的工作做完，而是把客户需要的工作做完。”，在实施固定总价合同的客户项目时，如何比较好的控制项目的范围和成本呢？有时候客户和我们在项目初期都没有办法完全界定清楚，什么是客户需要的工作？

展开 ∨



再不吃胖我们就老了

2020-01-10

很难相信居然还有小瀑布这样的团队，游戏行业策划还能闲等着下一个迭代，早被喷死了。



Raymond吕

2020-01-09

老师说的大瀑布中小敏捷的累，我们正在踩。

我们公司内部的工具开发，面向开发做小工具，可能是开发提的需求一般都比较明确，所以效率并没感觉降低。老师说导入敏捷要先找到组织当前的痛点，如果现在并没有觉得“痛”，怎么办呢？

展开 ∨



落地得分

2020-01-09

我们是创业公司，主要工作就是主业务的日常迭代，都不是很大的需求。正常开发也是一周左右可以上线。任务故事根本无需拆分，这样怎么去实施敏捷呢？

展开 ∨



吴言乱语

2020-01-07

在实际工作中，因为涉及外部团队的配合，所以需求拆分可以按照内部测试单元和外部测试单元进行分类，从而更好的评估外部依赖性，规避项目风险。

展开 ∨





louis lam

2020-01-07

如何拆分用户故事是敏捷开发一个很关键的部分，而评估是否是一个好的用户故事标准就是能否独立进行上线，如果做不到独立上线，那这个还不叫用户故事，测试也没办法并行进行，也不是真正的敏捷开发。另外敏捷开发跟现在到微服务架构是相辅相成，敏捷开发非常适合微服务这种开发模式，微服务能够提高敏捷开发的效率。总之，敏捷开发的精髓应该是团队至上，小步快跑，快速迭代，拥抱变化。不知道对于敏捷开发这样的理解有...
展开

作者回复: 赞一个！优秀用户故事的标准业界有通用的认识，公认的是符合INVEST原则。在《用户故事与敏捷方法》一书中有介绍，可以去看一下。

