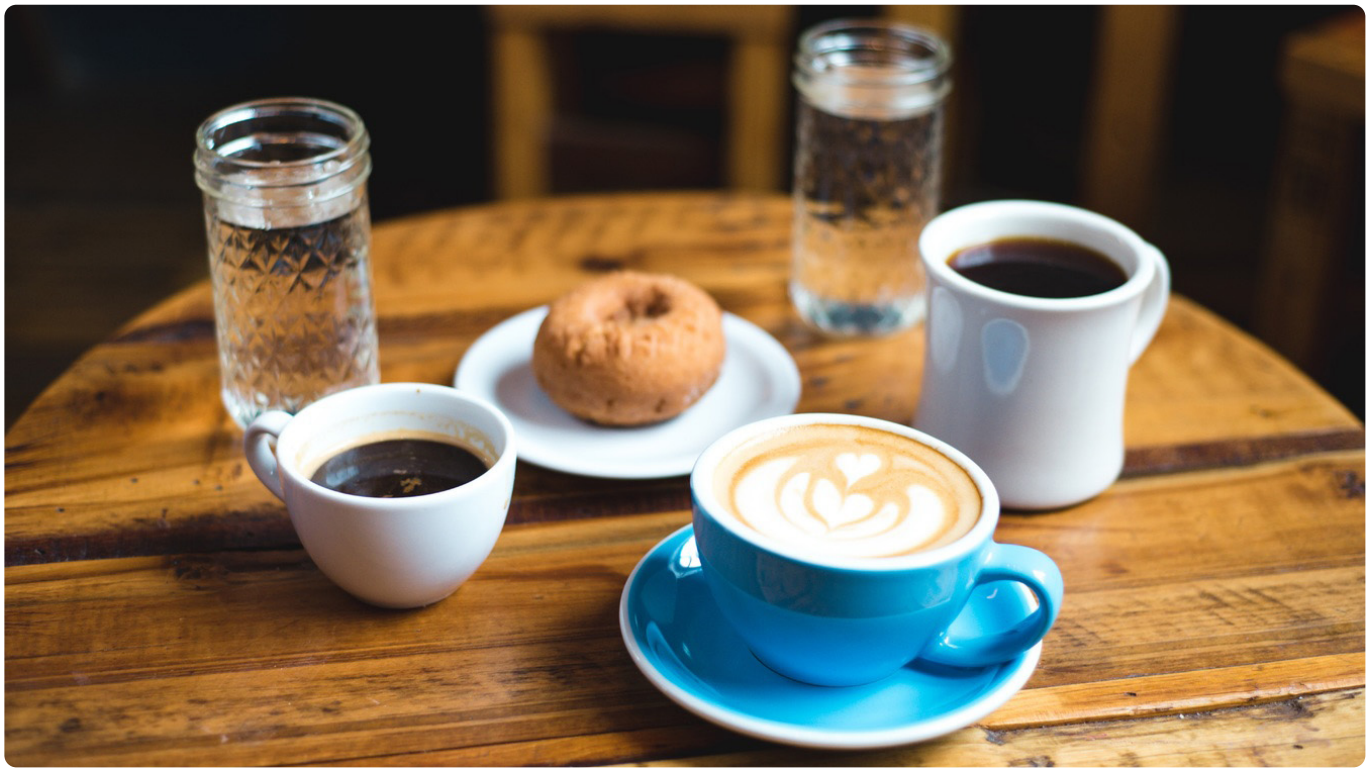
02 | 老生常谈：你真的知道敏捷到底是什么吗？

2020-01-06 宋宁



讲述：宋宁
时长 17:52　大小 16.37M

你好，我是宋宁。今天这节课我要给你讲一下到底什么是敏捷。

当谈到敏捷时，大家往往是仁者见仁，智者见智，有各种不同的理解。然而这里面，有不少是对敏捷的误解，比如在我平时的咨询过程中，会听到一些团队成员这样说敏捷：

> 敏捷来了，太好了，我们只负责开发软件就够了，再也不用做文档，也不用做设计了；
>
> 敏捷就是快，原来要 6 个月才能完成的项目，用了敏捷后，周期就可以缩短到 3 个月了；
>
> 敏捷就是加班，用了敏捷后，由于在迭代开始时许诺要做的事到迭代结束时一定要完成，所以我们比原来加班更严重了；

Scrum 就是敏捷，敏捷就是 Scrum，这俩是同义词；

敏捷是自由的、无约束的，不需要那么多条条框框，跟随自己的心来做就好了。

如果你带着这些对敏捷的误解去做敏捷，很可能会做得一塌糊涂。作为过来人，我想我应该在给你讲怎么做敏捷之前，先给你捋一捋到底什么是真正的敏捷，以便你能正确地理解敏捷。

在我看来，**大家之所以对敏捷有那么多的误解，归根结底，是忘记了做敏捷的初心，忘记了敏捷的价值观和它的基本原则，只把注意力集中在怎么做上。**

所以要想真正了解敏捷，需要从敏捷的价值观、原则以及具体的方法和实践上对它有全方位地认识，只从任何单一的方面去谈敏捷，都像盲人摸象，是片面的。

敏捷的价值观：正确理解敏捷的初心

我们先来看一下敏捷的价值观。2001 年，17 个轻量级方法论者在美国的犹他州，发布了敏捷宣言，在里面阐明了敏捷的 5 条价值观，我在下面也给你列出了这 5 条价值观：

1. 个体和交互**胜过**过程和工具
2. 可以工作的软件**胜过**面面俱到的文档
3. 客户合作**胜过**合同谈判
4. 响应变化**胜过**遵循计划
5. 虽然右项有价值，但我们更重视左项

请注意这 5 条价值观中的最后一条：“**虽然右项有价值，但我们更重视左项。**”这一条其实是对前 4 条价值观的解释说明，很多人在传递敏捷价值观时其实只讲了前面 4 句，而忽略了最后这一句，很大程度上，这导致了大家对敏捷的误解。

结合这一条，前 4 条价值观中“胜过”一词的意思是：**与每一条中右面的内容相比，左面的内容是敏捷更加重视的，但要注意，这并不表示让你停止做右面的内容。**

以开篇的中一个误解“敏捷来了，我们只负责开发软件就够了，再也不用做文档，也不用做设计了”为例，我们来看看对它的误解在哪里。

在敏捷价值观的第 2 点里，敏捷认为“可以工作的软件比面面俱到的文档更加重要”，然而这并不是说我们就完全不要文档了。对这句话的正确理解应该是：**敏捷重视可工作的、有价值的软件，减少一切不必要的文档。注意，是不必要的文档，而不是所有文档。**

那么对于一个项目来说，什么样的文档是没有必要的文档呢？比如说一些交接类的文档，开发人员开发完毕要提交给测试部门测试，还要写一个提测单，再一级一级批复，我认为这样的文档就是可以省略的。因为在敏捷里，开发人员和测试人员是在一起工作的，所以提测的工作不需要走如此麻烦的申请与审批，开发人员做好了需要提测时，直接在软件上点击一下“提交”，并告知测试人员就足够了。

什么又是有必要的文档呢？比如那些写着重要设计方案的文件，如果在后期遇到问题，其他同事就要回头查看文件，找出问题所在；或者是系统后期开发完成、需要转交给其他人维护时，若想知道这个系统当初是怎么做的，也需要去查看当时的系统设计文档，所以这类设计方案是需要保留下来的。我想你可以根据自己的项目情况，考虑哪些是重要的文档，哪些是不重要的。

所以说，**敏捷的价值观并未否定或贬损“右项” 的价值**，“流程和工具”、“ 详尽的文档”、“ 合同谈判” 以及“ 遵循计划” 这些右面的内容也很重要，在敏捷里并不是完全不做。比如在敏捷中也是有计划的，只不过计划的方式与传统瀑布模式的计划方式不一样罢了。

敏捷的价值观体现了敏捷的初心，只有正确理解它，你才能更深层次地理解敏捷。敏捷的初心是通过一系列方法来让我们的研发工作更加灵活、有序和高效，所以敏捷的价值观重视人的能动性，强调人与人之间的协作，也更重视对变化的应对，这些都是为了能够更好、更灵活地组织和管理研发工作，但如果“流程和工具”、“ 详尽的文档”、“ 合同谈判” 以及“ 遵循计划” 同样能让研发工作更有序和更高效，那敏捷是不反对的，也不会放弃不做，这才是敏捷的真谛。

敏捷的原则：正确理解敏捷的基石

上面我带你重新理解了敏捷的价值观，但对于敏捷来说，只有价值观还不够具体，为了能更具体地指导工作，由敏捷的价值观又引出了 12 条敏捷原则，在下面我也给你列出了这 12 条原则，方便的话，你可以打开 App 看一下：

1. 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。

2. 即使到了开发的后期，也欢迎改变需求。敏捷过程利用变化来为客户创造竞争优势。
3. 经常性地交付可以工作的软件，交付的间隔可以从几个星期到几个月，交付的时间间隔越短越好。
4. 在整个项目开发期间，业务人员和开发人员必须天天都在一起工作。
5. 围绕被激励起来的个体来构建项目。给他们提供所需的环境和支持，并且信任他们能够完成工作。
6. 在团队内部，最具有效果并且富有效率的传递信息的方法，就是面对面的交谈。
7. 工作的软件是首要的进度度量标准。
8. 敏捷过程提倡可持续的开发速度。责任人、开发者和用户应该能够保持一个长期的、恒定的开发速度。
9. 不断地关注优秀的技能和好的设计会增强敏捷能力。
10. 简单——使未完成的工作最大化的艺术——是根本的。
11. 最好的构架、需求和设计出自于自组织的团队。
12. 每隔一定时间，团队会在如何才能更有效地工作方面进行反省，然后相应地对自己的行为进行调整。

这 12 条原则是正确理解敏捷的基石，所以在这里，我想结合开篇那几条对敏捷的误解来为你解读敏捷的原则。

针对“敏捷就是快”这个误解，你要注意的是，在敏捷的原则里，可从来没有说过“敏捷就是快”这句话。所以你需要正确理解“快”这个词。

如果你将“快”理解为采纳了敏捷以后，你的代码编写速度就立马加快了，那是非常不现实的。虽然在敏捷中我们也会使用一些方法来训练整个团队的代码编写能力，但是这并不意味着用了敏捷后，程序员敲代码的速度就得到了显著提升。况且即使敲代码的速度加快了，项目整体上线的速度就也一定能快吗？

那么敏捷在交付上会带来什么变化呢？你可以先看一下敏捷原则里的第 1 条和第 3 条，总体的意思是：针对交付，敏捷希望能够尽早交付可工作的软件给客户，而不是最后一次性交给客户，并希望交付的间隔时间越短越好。所以说，**敏捷中的“快”其实指的是反馈更快，反馈更及时。**

这样，我们就能更快、更准地得到用户的真实反馈，尽早修正，为用户创造有价值的产品和竞争优势。在这一过程中，客户也可能更早一点想清楚想要什么样的产品，你也能比预想时间更早达成客户的目标。这里你也要注意，是有“可能”，而不是“一定”。甚至，由于客户能够早些看到产品，通过修正，也许在某一个时间节点上，客户觉得目的达成了，项目可以结束了，这或许比客户预想的上线时间还要早。

但这绝不是通过快速写代码来完成的，而是通过不断检视客户的需求，总是优先做最有价值的事，减少浪费来达到的。在瀑布模式的项目管理过程中，我们把原计划的事情全部做完，项目即宣告结束；而在敏捷的原则下，客户的目标达到才意味着项目可以结束，这是因为敏捷是以业务价值和业务目标为导向的，根据业务价值和业务目标，短迭代使客户对自己的需求更清楚了，有可能做的事情更少了，所以时间才减少了，交付也更快了。

同样，**“敏捷就是加班”**这个理解当然也是有失偏颇的。你可以回到敏捷原则里，重新看看第 8 条：“敏捷过程提倡可持续的开发速度。责任人、开发者和用户应该能够保持一个长期的、恒定的开发速度。”

你可以清晰地看到，敏捷强调“可持续的开发速度”，这指的是团队能一直以稳定的开发速度持续下去，而不是为了加速开发，本次或几次迭代一通加班，透支团队成员的健康，后面又由于员工身体不支等原因导致接下来的迭代开发产能下降。试想如果天天加班，员工能否能一直保持高昂的斗志和较高的工作效率？能否保证一直以这样的开发速度开发下去？我想这是不行的。

那怎么才能做到保持“可持续的开发速度”呢？在咨询过程中，我看到能做到这一点的开发团队，通常都是这样做的：

1. 他们在迭代开始的时候，不会过度承诺，也就是能完成多少工作，就承诺多少工作。
2. 严格遵守纪律。他们在迭代开始之后，原则上不再接受需求的增加，如果一定要往他们的迭代待办事项列表里增加其它需求，就一定要同时从其中拿走等量的需求。

这么做有什么好处？我认为这样做，可以使开发团队聚焦在自己的事情上，不受干扰，效率更高，从而形成稳定的开发节奏；而稳定的开发节奏可以加强我们的预见性，这样我们预计的上线时间才可能是精准的。

所以，如果你的团队采纳了敏捷后加班更严重了，那么我建议你参照上面的做法来自我检视一下，看看你的团队是否在一个迭代中承诺的事情太多了，也就是工作范围是否太大了，如果是的话，那你可以结合团队的实际产能，根据需求条目的优先级来进行调整；此外，你的团队是否遵守了纪律，如果不遵守纪律，那额外的加班肯定不可避免。

现在，我们回过头再来看看敏捷原则里的内容，会发现它和敏捷宣言一样，重视研发各方的协作，并强调了持续改进、响应变化。只不过在这 12 条原则里，对敏捷重视的价值做了更细致的阐述，涵盖了软件项目管理的所有基本流程，而且这些流程很具体，让大家有了更可参考的标准，将敏捷的价值观落实到具体的、可操作的原则之上。

因此正确理解敏捷的原则，并以此为基准去行事，才能在后面具体的敏捷实践中不偏离，最终取得敏捷项目的成功。

敏捷的方法与实践：正确落地敏捷的基础

但很显然，只有价值观和原则，敏捷是不能落地的，你还需要一系列的敏捷方法和具体实践。

在提出敏捷时，建立敏捷联盟的 17 位大师所创立的敏捷方法包括：极限编程、Scrum、特征驱动开发、动态系统开发方法、自适应软件开发、水晶方法等等，这些方法被统称为敏捷方法。到现在还有很多关于规模化敏捷的方法，比如说 SaFe 和 Less；也有更多技术实践，比如说测试驱动开发。可以这么说，凡是符合敏捷价值观和原则的方法论，都可以归到敏捷的大伞下。

怎么样，敏捷是不是包罗万象？但敏捷的方法和实践虽然很多，你一定要结合自己的需求来选择。所以在这里我想和你强调的是，**这些方法从共性上来说都遵守了敏捷的价值观和原则，不同的是它们针对了不同的应用场景**，比如说 Scrum 在新软件开发中更好用，而看板在维护类的软件开发中更胜一筹。

所以开篇我们提到的 “Scrum 就是敏捷，敏捷就是 Scrum”这一说法，也是相当片面的，是对敏捷的误解。敏捷还有很多如上面讲到的方法，如果只认识这俩，那你在采纳敏捷时，无疑就受到了限制。

说到敏捷落地的方法，我们可以回头看看开篇的那个误解“敏捷是自由的，无约束的”，我想以敏捷的方法 Scrum 为例，来谈一下为什么这个说法是不对的。

Scrum 框架看起来很简单，很多人以为它不过就是“三三五五”：3 个角色（产品负责人、团队、ScrumMaster)，3 个工件 (产品待办事项列表、迭代待办事项列表、燃尽图），5 个会议（迭代计划会议、每日站会、迭代回顾会议、迭代评审会议、产品 Backlog 梳理会议），5 个价值（承诺 、专注、开放、尊重、勇气），以为只要把上面这些事情都照搬过来做完就万事大吉，就能用好 Scrum 这一方法了。但其实 Scrum 也是有约束条件的，如果不按照这些约束条件来使用它，是用不好这个方法的。

关于 Scrum 的约束条件，这里我举最重要的两条来说明：

1. 迭代计划会议开始前，产品负责人需要准备好需求条目，使需求达到准入标准；
2. Scrum 讲究时间盒，包括迭代的周期、各个会议，这些都要遵守时间盒的约定。

如果不遵守第 1 条约定，你会发现你的团队即使用了 Scrum，仍避免不了研发节奏被打乱的命运；如果不遵守第 2 条约定，你会发现你的团队被耗在各个会议上，会议效率又很低，团队成员很快就会感到厌烦。所以说，Scrum 是有纪律的，如果不遵守纪律，自由自在无约束，那么使用它注定是痛苦的，也达不成既有目的。

从上面 Scrum 的例子我们可以看出，了解敏捷的方法和实践，不能只了解它的表面，更要深度了解它背后的规则和深意，只有这样才能正确地应用它，并让它为我们的研发管理服务。

针对敏捷的每一种方法我建议你在使用前问自己 3 个问题，从下面几个方面去深入了解它：

1. 这个方法能解决什么样的问题？
2. 有没有使用前提？
3. 有没有相应的使用规则？

此外你还可以借鉴别人是怎么用这些方法的，看他们在使用过程中有没有遇到什么坑，如果有又是怎么避免的。我希望通过这样的自我提问，能让你深度了解敏捷，并在日后能正确地使用敏捷的方法。

总结

说到现在，你是不是已经明白到底什么是敏捷呢？我希望通过这篇文章，你能够了解什么是真正的敏捷，并能够从人们对敏捷的看法中分辨出哪些是对敏捷正确的理解，哪些是误解。综合上面内容，我来总结一下，希望能给你带来帮助。

一句话，**敏捷 = 价值观 + 原则 + 一系列符合价值观和原则的方法。单纯说敏捷是一种方法，肯定是片面的；只强调敏捷的价值观和原则而不重视方法也是不对的，因为那样敏捷就飘在空中，不能落地了。**

因此对于敏捷，需要从敏捷的价值观、原则、具体方法和实践上对它有全方位的认识，只从任何单一的方面去谈敏捷，都是片面的。更重要的是，做敏捷不能只关注具体的实践和方法，还要时时刻刻记住做敏捷的初心，不能偏离了它的价值观和原则。

思考题

结合上面的内容，我想请你来思考一下，你还知道有哪些对敏捷的误解么？请对照着前面提到的敏捷的价值观和原则来检视一下吧。

上一篇　01 | 灵魂拷问：如何利用敏捷思维更好地解决实际问题？

下一篇

精选留言 (16)

写留言

jinny
2020-01-07

敏捷中的‘快’是指应对及时，反馈及时，就像马歇尔说过的那句话：一个及时的中庸决策，比不及时完美决策要好。

展开

作者回复: 👍

1 8

桃子-夏勇杰
2020-01-07

1. 敏捷就像健身，怎么看都对，但是，背后都需要强大的自律。
2. 敏捷就像健身，怎么看都对，但是，背后都需要强烈的改变意愿和一点的自律。
3. 敏捷就像健身，上了点年纪，有了点经历，还不放弃的，才会追求。

大家觉得哪句更能激励人尝试敏捷?

展开

1 3

吃饺子不吐饺子皮
2020-01-06

我是一个项目的团队成员，目前项目中存在大量加班情况，我想讲敏捷思想融入项目开发中，但我不知道这会不会影响其他成员的工作习惯，从而导致计划落空。所以想问下团队成员如何自行实施敏捷开发。

作者回复: 首先恭喜这位朋友，已经有意识想导入敏捷了。关于大量加班的问题，建议分析一下背后的真实原因。比方说到底是什么原因导致的？是需求反复更迭？还是需求范围没界定好，团队承诺过多？还是本身人员效率不高？等等，然后有针对性地采取措施。另外敏捷的导入和真正使用它达到既定效果不是一件简单的事情，因此需要各方面的配合。因此建议先跟团队领导商量，让他们看到敏捷的好处，得到他们的支持。对于团队成员，也需要跟他们树立信心，可以先从简单的实践着手，让大家看到一些益处，然后就自然而然地想跟着做了。我们的整个专栏都有这样的思想，可以看看后面实战篇的方法。

4 3

Twinkle
2020-01-06

怎么去培养团队每个成员的敏捷思维

展开

> 作者回复: 谢谢，专栏可以给他一些好的观点和思维方式，另外专门的布道也是非常重要的，专栏后面的09里的好的敏捷教练也会负责一直帮助团队理解敏捷背后的原因，帮助他们不断地成长。人其实是非常有意思的，如果他不明白为什么，很难转化成态度和行为的改变

1 2

小明
2020-01-10

老师，有个问题，需求在一个迭代里不加，可是遇到，产品设计不合理的情况，开发过程中发现，导致的不得不改的需求怎么办，自己怎么规避这种问题，希望老师能看到

1

escray
2020-01-08

温习了一下敏捷宣言和12条敏捷原则，如果去看英文原版的话，会发现其实宣言只有四条，而作者提到的第五条，其实应该是一个备注。不过，如果按照五条来宣讲的话，可能会更容易被接受。

在原则里面，我觉得比较重要的是“被激励起来的个体”，如果没有这个，那么其他的...

展开

1

Aaron(健廷)
2020-01-06

醍醐灌顶阿，谢谢老师

展开

1

gevin
2020-01-13

敏捷方法，我一直都觉得分为“敏捷的软件开发方法”和“敏捷的项目管理方法”两类

赵欣光
2020-01-13

今年，整个事业部都要开始推行敏捷，首先就是要把事业部下的每个部门的组织结构划分不再以室为单元，而以小组划分。我个人是没有搞懂敏捷的，所以来学习这个课程，但感觉不到公司的这种做法与敏捷有什么相关性。继续学习的同时，也拭目以待吧。

展开


小老鼠
2020-01-12

1，迭代次数增加，回归测试肯定增大，势必会引入自动化测试，但现在好多企业自动化建立不起来，咋办？2，在几次迭代后客户发现他们的需求变多了，这种情况如何处理？


曾泽伟
2020-01-09

不管是敏捷还是瀑布，文档还是很重要的，但我碰到很多公司和人告诉我，互联网公司不写文档的，我有时候真的是比较无语，什么文档都没有，代码注释也没有，这和是不是互联网公司有什么关系，重要的文档（设计理念、架构思路等等）是一定要有的，文档可以简易不可以没有，可是很多人不懂，真累

展开


那土
2020-01-09

个人敏捷与团队敏捷，这个老师有讲解莫？

展开


Raymond吕
2020-01-09

天下武功唯快不破，敏捷的快我理解是信息流的快，始终保持对当下的目标对齐，贯穿整个开发过程。


y欧尼酱
2020-01-08

交付快，反馈快，目的是为了满足客户的需求，保证项目符合客户的要求，同时让开发人员少走弯路，避免带来沟通和需求理解错误的试错成本。

展开

Tiger
2020-01-07

我觉得我们项目组里还有个敏捷误区，PM说敏捷交付就是迭代交付。就跟说敏捷就是Scrum一样的感觉。

作者回复: 是的，很多朋友都理解不全面，所以专门提了这一点

吴言乱语
2020-01-07

逐条检视了一下：
针对第二条：传统做法是不欢迎在开发后期，甚至是任何阶段，改变需求；但是调整思考方式，敏捷的最终目标是业务目标和业务价值，需求变更若可以为客户带来更好的市场竞争力，欢迎变更。但同时开发团队不过度承诺的原则，要将变更量化，替换原有需求，保证稳定的开发速度。...

展开