

04讲接到需求任务，你要先做哪件事



我们书接上文，继续讲程序员小李的故事。这次小李接到一个新的需求，让他开发一个单点登录的服务，经过几天的奋战，他顺利地写完了所有的代码。正好产品经理小王路过他身边，顺便问了他一下。

小王：单点登录做得咋样了？

小李：做完了，我给你演示一下。

小李演示了一遍自己做的功能，小王看上去很满意。

小王：不错。不过，怎么没有支持验证码？

小李：为什么要做这个？

小王：这不就是登录的一部分吗？

小李：哪里规定要做验证码了？

小王：现在做登录哪有不用验证码的？

我想你已经嗅到了双方谈话的火药味，这个时候如果双方都不能很好地控制自己的情绪，那接下来一场体力的较量可能就一触即发了。

为什么双方会有这么大的分歧呢？其中一个重要的原因是，开始实现这个需求之前，任务双方都没有清晰地定义好边界，没能把需求描述清楚。

需求描述的问题

在软件开发中，程序员做什么一般都由需求来定义。我们都知道，需求是软件开发的一个重要组成部分，但你可能并没有仔细想过，不同的需求描述方式，可能会影响我们程序员对需求的理解。

因为信息的传递是会衰减的，你不可能把你理解的信息100%传递给另外一个人，而这中间，如何传递，也就是如何描述将直接决定衰减的比例。

很多公司的软件开发模式是基于功能列表的，这个列表“规定”了程序员要做的功能，各个组从产品经理那里领来开发列表，然后“照单抓药”开始写代码。但是，通常这种功能列表只是一些简单的描述，你并不能看到全局。

很多团队的一个状态就是，程序员们都知道要开发的功能是什么，但这个功能是谁在什么样的场景下使用的，很多人却回答不上来。如果你去问他为什么要开发这个功能，他通常会说：这是功能列表里规定的。

这种功能列表式的需求描述方式，将一个完整的需求敲成了碎片。只有所有功能全部开发完成，对接在一起的时候，才是“破镜重圆”的时刻。

也就是说，不到最后一刻，大多数人并没有一个完整的图景，这就相当于看不到完整的“终”。顺着这个思路做下去，你会在最后关头遇到许多意料之外的问题，其结果必然是手忙脚乱。

根据这种基于功能列表的需求描述，每个组在安排工作的时候，都会按照自己的理解进行功能排列。

所以，当你的组完成了一个功能时，这个功能却可能上不了线，因为你还要依赖于其他组的工作，而这个组不巧，却刚好把相关的功能开发排在了后面。

这还只是两个组之间有依赖的情况，如果需要多个组协同，可以想象，状况会多么糟糕。

所以，当我们对产品经理说“时间不足，砍掉一些需求吧。”得到的答案肯定是，“对不起，做不到，因为需求已破碎，没办法调整。”

因此，一些新的需求描述方式也就应运而生，这其中，用户故事（User Story）是我最喜欢的一种方式。它是站在用户的角度来描述了一个用户希望得到的功能，关注用户在系统中完成一个动作需要经过怎样的路径。既然它是“故事”，它就需要是一个完整的场景，可以讲述出来。

“用户故事”有什么用？

我们先来以用户密码登录为例，看看用户故事长什么样？一个完整的用户故事大致包含以下几个部分：

- 标题，简要地说明这个用户故事的主要内容，比如：注册用户使用用户名密码登录。
- 概述，简要地介绍这个用户故事的主要内容，一般会用这样的格式：
As a (Role) , I want to (Activity) , so that (Business Value) .
意思就是：作为一个什么角色，要做什么样的事，以便达成一种怎样的效果。其中最重要的是，告诉别人为什么要做这件事，虽然只有一句话，却往往是很多人欠缺的思考，只知做，不知为何做。
举个概述的例子：作为一个注册用户，我想要通过用户密码登录，以便我可以使用注册用户才能够使用的服务。
- 详述，详细地描述这个用户故事的完整流程，我们会把操作流程、用户界面等信息都放到这里。
比如：用户使用正确用户名和密码登录，就可以登录成功；如果密码不正确，则登录页面提示用户“用户名密码不正确”。基本上，看到这个部分，程序员就可以在心中描绘出这个用户故事的样子了。
超出范围的部分，比如：第三方登录不在范围内，这个部分主要是限定人们不要进一步发散。
- 验收标准，这个部分会描述一个正常使用的流程是怎样的，以及各种异常流程系统是如何给出响应的，这是程序员常常会欠缺的思考。它会把详述中很多叙述的部分变成一个具体的测试用例。比如，下面我给出的两个验收用例：
正常场景：给定一个注册用户张三，其用户名是zhangsan，密码是foobar，当张三使用zhangsan 和 foobar 登录系统时，可以成功登录，登录成功后，跳转到用户中心。
异常场景：给定一个注册用户张三，其用户名是zhangsan，密码是foobar，当张三使用zhangsan 和 wrong 登录系统时，登录失败，在登录页面上提示“用户名密码不正确”。

在前面的例子中，小张和小王之所以会对需求是否完成产生分歧，是因为大家对于需求完成的定义不同。对于这种情况，我们能怎么办呢？

这个模块的主题是“以终为始”，现在你看到了用户故事是如何描述需求的，你或许已经知道我要说什么了，没错，这里非常关键的一点就是“验收标准”。很多人学习用户故事，认为最重要的是记住“As..., I want to ..., so that ...”这样的需求描述方式。

在我看来，无论采用哪种需求描述方式，这部分也都是能说清楚的。那我们要从用户故事中学到什么呢？我认为就是用户故事的关键点：验收标准，它可以清晰地定义出需求边界。

验收标准非常重要的一环是异常流程的描述。大部分程序员都擅长解决正常流程，而异常流程则是最容易忽略的，也是产生扯皮的关键环节。既然容易扯皮，我们就在一开始把它定义清楚。怎么才算做完需求呢？验收标准说了算。

采用用户故事之后，我经常在写完了主要流程之后，再去看一下验收标准，为自己的开发查缺补漏。因为我知道，那是标准，达不成就不算任务完成。

当我们说自己开发完成，可以交给测试人员测试时，我们需要照着验收标准给测试人员演示一遍，证明我们的系统确实能够跑通。这之后，测试人员才会把系统接手过去，做更系统的测试。

验收标准给出了这个需求最基本的测试用例，它保证了开发人员完成需求最基本的质量。如果你了解 BDD（Behavior-Driven Development，也就是“行为驱动开发”），就可以按照验收标准中给出的内容编写验收测试用例了。

在实际工作中，许多产品经理把需求交给开发人员之前，很多细节是没想清楚的，那种功能列表式的需求常常只包含了正常路径，那些缺失的细节就是在后续的过程中，由开发人员补全的。用户故事就是一种固定的格式，让他们把这些应该想清楚的问题想清楚。

如果你的团队采用用户故事的格式进行需求描述固然好，如果不能，在功能列表中，补充验收标准也会极大程度地改善双方协作的效率。

你的角色

或许你会有这样的疑问，如果产品经理通过用户故事的方式，将需求实现细节都描绘得清清楚楚，那我们程序员的发挥空间在哪里？请注意，验收标准所给出实现细节应该是业务上的，程序员在这种问题上思考才是真正意义上的浪费时间，我们的发挥空间应该是在技术实现上。

然而，在现实情况中，很多团队做不到这种程度。

你会发现，我们在开发中之所以会“丢三落四”，很重要的一个原因是，在开发一个功能特性的时候，因为一些环节的缺失，我们不得已扮演了很多的角色，其中之一就是产品经理。你是一个专业的程序员，但大多数情况下，你却只是一个业余的产品经理，“丢三落四”就在所难免了。

或许你会说，我在一个小公司工作，公司没那么多的人，没有专门的产品经理，只有我们几个“全世界都缺”的程序员，需求都是老板扔给我们的，谁来帮我们写验收标准呢？

没办法，答案只能是你自己。虽然你名义上是程序员，但当拿到一个需求的时候，你要做的事不是立即动手写代码，而是扮演产品经理的角色，分析需求，圈定任务范围。相信我，事前分析绝对比你拿一个写好的系统给老板，而他却告诉你这不是他想要的，好太多了。

另外我想提醒你注意的是，**扮演不同角色的时候，我们的思考模式是不同的。**还是以开开发用户名密码登录为例，你想到的可能是：输入正确的用户名和密码可以正常登录，输入错误的用户名和密码不能登录，而且给出提示。

如果你只扮演开发人员的角色，想到这些就算不错了。但如果你扮演的是产品经理的角色，会从产品的角度进行思考，也就会

看到不同的内容，比如：

- 登录是否需要验证码
- 是否需要第三方登录
- 用户名和密码的长度在系统内是否有限制
- 密码是否需要满足一定的规则
-

我知道，如果让你来填写，这个列表会更长。可能这并不是我们都需要完成的功能，但站在分析的角度，这都是我们要考虑的问题，一个登录功能，绝不仅仅是用户名和密码校验那么简单的。我们能想到这些，仅仅是因为我们正在扮演一个不同的角色。

所以，如果你要兼顾开发人员和产品经理两个角色，建议你先扮演好产品经理的角色，多花点时间把验收标准制定好，再回到开发人员的角色上去写代码。毕竟，**最好维护的代码是没有写出来的代码。**

总结时刻

总结一下今天的内容。需求，是软件开发中的一个关键环节，一旦需求理解出现问题，势必会造成大量的浪费。传统的功能列表只是简单罗列了要实现的功能，丢失了大量的上下文，会导致团队成员对于需求“只见树木不见森林”。

而在比较大的团队中，更是会将一个功能分拆到多个小团队中，每个人看到的只是功能碎片。于是，后来产生了其他的需求描述方式，比如用例和用户故事。

在实际的开发过程中，大量的分歧来自于对“需求完成”的定义。当我们把“以终为始”的原则应用在需求领域中，就会注意到，用户故事有一个非常重要的组成部分是验收标准。

验收标准不仅仅描述出了正常流程，也会关注到异常流程的处理，它也是我们验收测试用例的起点。一旦事先定义好验收标准，大量的扯皮工作就随之烟消云散了。

理解了验收标准的作用，即便我们不使用用户故事来定义需求，依然可以把用户故事中的关键点应用到自己的实践中，在功能列表的每个功能定义中，增加验收标准。

如果今天的内容你只能记住一件事，那请记住：**在做任何需求或任务之前，先定好验收标准。**

最后，我想请你回想一下，在实际工作中，你是如何澄清你的需求，或者因为需求不清晰给你造成了哪些困扰？欢迎在留言区写下你的想法。

感谢阅读，如果你觉得这篇文章对你有帮助的话，也欢迎把它分享给你的朋友。

10x 程序员工作法

掌握主动权，忙到点子上

郑晔

火币网首席架构师
前 ThoughtWorks 首席咨询师
TGO 鲲鹏会会员



新版升级：点击「 请朋友读」，10位好友免费读，邀请订阅更有**现金**奖励。

精选留言



liu

程序员的核心职责是如何实现产品功能，怎么实现功能；前提是理解产品功能，需要实现哪些功能。有些项目经理，产品经理与程序员角色混淆。你同他谈功能，他同你谈技术实现，你同他谈技术，他同你谈产品（需要实现哪些功能）

2019-01-02 08:59

作者回复

你能分辨出你们讨论的不是一回事，这是很强的能力。其实，与任何角色沟通都一样，能够识别出双方讨论的不是一回事，及时制止，会大幅度提高沟通的效率。

2019-01-02 22:15



彩色的沙漠

老师您好

一个产品经理没有想清楚需求的前提给我们开产品需求讲解会，每次就会造成开发和测试的轮番轰炸产品细节，产品经理的回答就是我去再想想，第二次再开产品需求会的时候还会遇到新的产品细节，产品经理还会回答我去再想想。看来这个产品经理就是没有定义好验收标准，只是想好了市场部提给他的需求。

那么问题了

1、“验收标准给出了这个需求最基本的测试用例，它保证了开发人员完成需求最基本的质量”

这个验收标准给的太基本还是会在后期开发中浪费时间，尽可能的详尽很考验产品经理的职业素养以及时间成本，如何去衡量这个最基本的验收标准？

2、团队开发中需求变更在所难免的，产品变更需求如何同步给大家最有效，以及需求变更的痕迹如何可追踪。请问老师在团队开发对于这种问题有什么最佳实践，谢谢！

2019-01-03 11:40

作者回复

这种现象可能有两种原因，一是产品经理没想清楚，二是问题复杂到产品经理想不清楚。实际上，这是一个问题，产品经理缺乏好的工作方式，他需要将产品特性做分解，分解一个个的小需求去实现。另外，变更不可怕，大变更才可怕。这其实是任务分解模块要讨论的内容，敬请期待。

2019-01-03 12:40



Monday

项目都快到截止日期了，心里总是没底。因为开发，项目负责人，产品经理，测试工程师。。。都是自己。看完此篇才知道验

更新请加微信1182316662 众筹更多课程188

收标准如此重要。现在的情况是没有明确的验收标准，，，怎么做才是最好？

2019-01-03 08:34

作者回复

先有验收标准，按照自己的理解，怎么有助于提高怎么来。

2019-01-03 11:36



王小勃

打卡

开始之前就定义好验收标准

2019-01-02 11:50

作者回复

打卡的方式挺好的，加油

2019-01-03 19:05



喜悦

今日概念：

1. 开发功能列表：对开发标准做简易描述的列表，容易丢失信息；
2. 用户故事：用户故事是一种分析需求的方法，能将各个功能串联起来以便做场景化的思考。最重要的是它能确定验收标准，这将成为后续开发的准绳；
3. 验收标准：规定以终为始开发的结果，可以使用 DoD 来制定；

今日总结：

开发中的许多问题都是由于思考不够深入，沟通不够细致造成的，用户故事能将沟通中丢失的关键环节暴露出来，（切换角色至项目经理）走通用户故事之后再制定验收标准，这时我们的开发流程就变得可控了；

2019-01-04 22:49



大帅哥

产品经理太强势，需求总是列出几个功能点，用户故事和验收标准完全说不清楚，每次需求评审耗费大量时间讨论，结果功能做出来了，又说不是产品经理需要的，由此可见推动制定验收标准是何等重要

2019-01-04 08:29

作者回复

每个人都需要考虑一下，令自己信服的到底是态度，还是道理。做出的东西不是需要的，需要怎么让它变成需要的呢？不去面对这个问题，问题永远解决不了。

2019-01-04 09:50



davidce

请作者如果能顺便介绍一下每节课内容相关的国内当前行业现状，就像回复评论中的那样就好了。

2019-01-03 02:07

作者回复

好建议，后面的内容我适当调整一下。

2019-01-03 09:27



光明

【需求不清晰】带给我最大的问题就是 经常返工！不光是需求上面，在领导分配的任务（也算是需求吧）上面理解出现偏差，最终做出来的东西不符合领导要求，这和上一篇的 DoD 有共同的地方，什么是完成？什么是验收标准？经历过，踩过坑 才知道【以终为始】的重要性！

2019-01-02 19:41



L

突然觉得，这个东西很适合产品看啊。。。

2019-01-02 09:58

作者回复

欢迎把文章转发给你们的产品同事！

2019-01-02 13:32



小可

我说两点感受

1.有不少开发，纯粹为了完成任务而完成任务，从来不考虑自己开发的功能到时候怎么用，开发完随便点点就提交测试了，我要是测试已经崩溃了

2.另外一个，遇到不专业的专职产品经理，给出的全是需求矩阵，两句话描述一个功能，开发的时候思考这些细节真是浪费时间

2019-01-02 08:56



KEN

定义好边界，很好得防止范围的扩散。定义好验收标准防止开发和产品之间互相扯。

2019-01-08 23:11



Xunqf

就像老师文中提到的，产品只考虑正常逻辑，异常是不做考虑的，然后交付设计，设计也不会出异常页面的UI，开发补充的异常提示信息和异常界面又往往不是产品想要的，往往容易在验收的时候反复修改。而且有些细节问题往往是在做的过程中才发现的，这个时候做一点就去和产品确认一下又非常影响开发效率，老师有什么比较高效的方法吗？

2019-01-07 23:37

作者回复

开发效率是什么？我反反复复强调地一个观点就是，开发效率不是写代码的效率。细节不确认就动手，后期还会反复，更浪费时间。

有些细节确实是只能在开发中发现的，所以，最好的办法是，让产品经理与开发团队坐在一起，随时确认。退而求其次，每天确认一次。

2019-01-08 09:06



雷小鸿

最好维护的代码是还没写的代码。的确意味深长。

2019-01-07 14:04



L

通过验收标准反摊需求？

2019-01-05 16:20



oddrock

本小节关键:用户故事的核心是验收标准

2019-01-04 08:17



到道可道

好像现在作为一个开发，去想产品功能的实现细节及业务逻辑实现，是业内的常态。

2019-01-04 01:38

作者回复

常态和合理是两件事，我在专栏之初就在说，直觉大多是错的。错的东西就应该是努力改变的。如果就这么认命了，我们为什么要学习呢？

2019-01-04 08:03



MarksGui

产品经理往往考虑不到很多异常情况，因为很多产品没有技术背景。作为项目经理，后续我应该承担起这部分工作的责任。第一划清需求边界，避免天马行空的讨论。第二着重考虑异常流程。第三定义验收标准，然后转交给测试。最后真心希望作者能给出国内大公司的一个开发流程和一些工具（比如腾讯的tapd就不错），这样能让很多小公司的人员也能学习大公司的先进管理模式。不胜感激。

2019-01-03 23:34



大彬

在华为，我们部门是没有产品岗位的，需求是se传递给开发人员的，se大多也是从开发升级上去的，所以需求看起来还算“顺畅”，拿到需求，就要先看，然后拉上se和测试，让se再给我们介绍一遍，解决开发和测试的疑问，这个环境是公司要求不能少的，所以都搞清楚再开始各自的工作，最后才能无缝连接起来

2019-01-02 14:04



James

信息的传递是有衰减的，所以你不可能百分之百的把你的需求传递到位。

简单的功能列表或者是需求列表是缺失上下文的，是碎片化的，不能很好的看到全局，对程序员来说，要想真正的理解需求是有困难的。很多时候并不知道这个要开发的功能是谁在什么场景下使用的。一定要让程序员知道为什么要开发这个功能。

如果没有用例或者用户故事加以说明，那么很有可能就导致理解的差异。

与此同时，还将很多产品该想清楚的工作转移到了程序员身上，程序员就相当于半个产品经理，而程序员应该做的是技术实现

，而不是过多的花时间思考业务。

对任务完成定义的不同，这就是扯皮的重要起因。

所以，关键的一点，一定要编写验收标准，程序员和产品根据验收标准检验是否完成任务。

作为一个研发人员，非常有共鸣！

2019-01-02 09:54



Alexdown

今天的内容深有体会，也帮我理清了一些事情——程序员的自信是通过完成一个一个功能/任务 / 项目逐步建立起来的，每当为了一个功能反复扯皮、返工再加上领导的负面评价，我都对会产生自我怀疑——我怎么总做不对？总达不到要求？我真的很差吗？看完这篇文章心里的别扭感，挫败感减轻许多

现在理性分析一下，我不该承担全部的责任，这里有公司或团队文化、工作流程的问题，按工程师招进来大部分时间却做产品经理的事，美其名曰“全栈”——只要给个功能列表，其他就不用管了。

说实话理想很丰满现实很骨感，如果用这篇文章的内容去反驳领导后果会很惨，因为他的头脑中已经固化了他认为对的流程，再加上他在公司的地位是不容你反驳的

我发现虽然名叫程序员但要扮演多重角色，主角是工程师，副角色产品经理+QA+运维等等，副角色扮演的不好即便是因你不适合该角色，你仍会被评价为“不好的程序员”

既然找到问题的根源那就对症下药，有意识地锻炼自己扮演副角色的能力，能提高多少算多少

不知老师和其他小伙伴们怎么想^_^

2019-01-02 09:47

作者回复

你提到的这些问题，后续更新的内容会给出答案的噢~

2019-01-03 19:10