

## 17 | 程序员也可以“砍”需求吗？

郑晔 2019-02-06



00:00

讲述：郑晔 大小：11.63M 时长：12:41

1.0x v

你好，我是郑晔。

我们前面讲的任务分解，主要是在讲开发任务的分解。今天我们换个角度，看看需求的分解。是的，需求也要分解。

有一次，我和一个做开发的同事聊天，他给我讲了他近期的烦恼。

**同事：**我们现在就是需求太多，开发的人太少，再这么干下去，哪天觉得自己抗不住了，我就拍拍屁股走人。

**我：**你没尝试着砍砍需求？

**同事：**怎么没尝试？产品的人都不同意。这批功能他们都说是关键功能。

**我：**你有没有尝试把需求拆开了再砍呢？

**同事：**还可以这样？

同事很惊讶，我一点都不意外。我们都是在说需求，但彼此对需求的理解却是大不相同。我先来问个问题，提到需求这个词，你会想到什么呢？

以我们用了好多次的登录为例，如果我问你这个需求是什么，大多数人的第一直觉还是用户名密码登录。

**基本上，闯入你脑海的需求描述是主题（epic），在敏捷开发中，有人称之为**主用户故事**（master story）。**

如果你对需求的管理粒度就是主题，那好多事情就没法谈了。比如，时间紧迫的时候，我想砍需求，你问产品经理，我不做登录行不行，你就等着被拒绝吧。

但是，如果你说时间比较紧，我能不能把登录验证码放到后面做，或是邮件地址验证的功能放到后面，这种建议产品经理是可以和你谈的。

这其中的差别就在于，后者将需求分解了。

大多数人可以理解需求是要分解的，但是，分解的程度不同，就是导致执行效果差异极大的根源。

以我的经验而言，**绝大多数问题都是由于分解的粒度太大造成的，少有因为粒度太小而出问题的。**所以，需求分解的一个原则是，粒度越小越好。

## 需求要分解

“主题”只是帮你记住大方向，真正用来进行需求管理，还是要靠进一步分解出来的需求。这里的讨论，我们会继续沿用前面专栏文章中已经介绍过的需求描述方式：用户故事，它将是我們这里讨论需求管理的基本单位。

如果你的团队用的是其他方式描述需求，你也可以找找是否有对应的管理方式。

上一个模块介绍“以终为始”，我们对用户故事的关注点主要在：用户故事一定要有验收标准，以确保一个需求的完整性。而在“任务分解”这个模块，我们看用户故事，则主要关注它作为需求分解的结果，也就是分拆出来要解决的一个个需求点。

在前面的讨论中，我们已经知道了用户故事的“长相”，但更重要的问题是，划分需求的方式有无数种，就像一块蛋糕，你可以横着切，也可以竖着切。如果你一刀不切，那就是拿着主题当用户故事。你也可以快刀飞起，把主题切碎。

每个人都会有自己喜欢的拆分方式，我相信知道拆分的重要性之后，你总会有办法的。这里，我主要想和你聊聊怎样评判拆分结果，毕竟我们要把它当作需求管理的基本单位。

只有细分的需求才能方便进行管理。什么样的需求才是一个好的细分需求呢？我们先来看看用户故事的衡量标准。

评价用户故事有一个“INVEST 原则”，这是六个单词的缩写，分别是：

- **Independent, 独立的。**一个用户故事应该完成一个独立的功能，尽可能不依赖于其它用户故事，因为彼此依赖的用户故事会让管理优先级、预估工作量都变得更加困难。如果真的有依赖，一种好的做法是，将依赖部分拆出来，重新调整。
- **Negotiable, 可协商的。**有事大家商量是一起工作的前提，我们无法保证所有的细节都能 100% 落实到用户故事里，这个时候最好的办法是大家商量。它也是满足其它评判标准的前提，就像前面提到的，一个用户故事不独立，需要分解，这也需要大家一起商量的。
- **Valuable, 有价值的。**一个用户故事都应该有其自身价值，这一项应该最容易理解，没有价值的事不做。但正如我们一直在说的那样，做任何一个事情之前，先问问价值所在。
- **Estimatable, 可估算的。**我们会利用用户故事估算的结果安排后续的工作计划。不能估算的用户故事，要么是因为有很多不确定的因素，要么是因为需求还是太大，这样的

故事还没有到一个能开发的状态，还需要产品经理进一步分析。

- **Small, 小。**步子大了，不行。不能在一定时间内完成的用户故事只应该有一个结果，拆分。小的用户故事才方便调度，才好安排工作。
- **Testable, 可测试的。**不能测试谁知道你做得对不对。这个是我们在前面已经强调过的内容，也就是验收标准，你得知道怎样才算是工作完成。

“INVEST 原则”的说法是为了方便记忆，我们这里着重讨论两个点。

第一个关注点是可协商。作为实现者，我们要问问题。只是被动接受的程序员，价值就少了一半，只要你开始发问，你就会发现很多写需求的人没有想清楚的地方。

在我的职业生涯中，我无数次将需求挡了回去，不是我不合作，而是不想做一些糊涂的需求。我之所以能问出问题，一方面是出于常识，另一方面就是这里说的用户故事是否有价值。**用户故事，之所以是故事，就是要讲，要沟通。**

还有一个更重要的关注点，也是这个模块的核心：小。无论是独立性也好，还是可估算的也罢，其前提都是小。只有当用户故事够小了，我们后续的腾挪空间才会大。

那接下来就是一个重要的问题，怎么才算小？这就牵扯到用户故事另一个重要方面：估算。

## 需求的估算

估算用户故事，首先要选择一个度量标准。度量用户故事大小的方式有很多种，有人用 T 恤大小的方式，也就是 S、M、L、XL、XXL。也有人用费波纳契数列，也就是 1、2、3、5、8 等等。有了度量标准之后，就可以开始估算了。

我们从分解出来的用户故事挑出一个最简单的，比如，某个信息的查询。这个最简单的用户故事，其作用就是当作基准。

比如，我们采用费波纳契数列，那这个最简单的用户故事就是基准点 1。其他的用户故事要与它——比较，如果一个用户故事比它复杂，那可以按照复杂程度给个估计。

你或许会问，我怎么知道复杂程度是什么样的呢？这时候，我们前面讲过的任务分解就派上用场了，你得在大脑中快速地做一个任务分解，想想有哪些步骤要完成，然后才好做对比。

所以，你会发现，任务分解是基础中的基础，不学会分解，工作就只能依赖于感觉，很难成为一个靠谱的程序员。

**估算的结果是相对的，不是绝对精确的，我们不必像做科研一样，只要给出一个相对估算就好。**

同一个用户故事，不同的人估算出的结果可能会有差别。怎么样尽可能在团队中达成一致呢？这就需要团队中的很多人参与进来，如果团队规模不大，全员参与也可以。

如果多人进行估算，你就会发现一个有趣的现象，针对同一个用户故事，不同的人估算的结果差异很大。

如果差别不大，比如，你觉得 3 个点，我觉得 2 个点，我们协调一下就好。但如果差异很大，比如，你认为 2 个点，我认为 8 个点，那绝对是双方对任务的理解出现了巨大的差异，这个时候，我们就可以把刚才在脑中进行的任务分解“摆”到桌面上，看看差异在哪。

通常情况下，是双方对需求的理解出现了偏差，这时候负责用户故事编写的同事就要站出来，帮助大家澄清需求。所以，**一般来说，估算的过程也是大家加深对需求理解的过程。**

估算还有另外一个重要的作用：发现特别大的用户故事。一般而言，一个用户故事应该在一个迭代内完成。

比如，你预计大小为 1 点的用户故事要用 1 天完成，而你团队的迭代周期是两周，也就是 10 个工作日，那 13 点的任务是无论如何都完不成的。那该怎么办呢？很简单，把它拆分成多个小任务，这样一来，每个小任务都可以在一个迭代中完成了。

所以，一般来说，用户故事有可能经过两次拆分。一次是由负责业务需求的同事，比如，产品经理，根据业务做一次拆分。另外一次就是在估算阶段发现过大的用户故事，就再拆分一次。

当我们有了一个合适的用户故事列表，接下来，我们就可以安排我们的开发计划了。只要厘清用户故事之间的依赖关系，安排工作是每一个团队都擅长的事情。

我在这里想回到我们开头讨论的话题。我们常说，需求来自产品经理，但需求到底是什么，这是一个很宽泛的话题。到这里，我们已经有了一个更清晰更可管理的需求，用户故事。这时候我们再说需求调整，调整的就不再是一个大主题，而是一个个具体的用户故事了。

许多团队真正的困境在于，在开发过程中缺少需求分解的环节。在这种情况下，需求的管理基本单位就是一个主题，既然是基本单位，那就是一个不可分割的整体。团队就被生生绑死在一个巨大的需求上，没有回旋的余地。

如果团队可以将需求分解，需求的基本单位就会缩小，每个人看到的就不再是“铁板”一块，才能更方便地进行调整，才会有比较大的腾挪空间。

## 总结时刻

软件开发中，需求管理是非常重要的一环。在需求管理上常见的错误是，需求管理的粒度太大，很多团队几乎是在用一个大主题在管理需求，这就让需求调整的空间变得很小。

结合用户故事，我给你讲了一个好的需求管理基本单位是什么样子的，它要符合“INVEST原则”。其中的一个关键点是“小”，只有小的需求才方便管理和调整。

什么样的需求才算小呢？我给你介绍了一种需求估算的方式，每个团队都可以根据自己的特点决定在自己的团队里，多大的需求算大。大需求怎么办？只要再进行分解就好了。

如果你对用户故事这个话题感兴趣，推荐阅读 Mike Cohn 的两本书 [🔗 《User Stories Applied》](#) 和 [🔗 《Agile Estimating and Planning》](#)。

如果今天的内容你只能记住一件事，那请记住：**想要管理好需求，先把需求拆小。**

最后，我想请你分享一下，你的团队在需求管理上还遇到过哪些问题呢？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。

© 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。

该试读文章来自付费专栏《10x程序员工作法》，如需阅读全部文章，  
请订阅文章所属专栏，新人首单¥19.9

立即订阅



慢慢

由作者筛选后的优质留言将会公开显示，欢迎踊跃留言。

Ctrl + Enter 发表

0/2000字

提交留言

### 精选留言(18)



Kăfkă<sup>2020</sup>

需求管理中印象最深的是刚进入开发行业时的一段经历。当时要显现某种单据，单据本身还嵌套子单据，原始需求是单据按层次显示，但某变量等于某个值时，某层次的单据就隐藏，但它的再下一层子单据需要显示。解决方案简单可以理解成页面要用递归方式才能正确显示。当时一心想显示自己牛鼻，因为无法用通常的页面模版机制，就自己写工具递归生成页面代码，看起来代码很巧妙，但别人难以维护。后来有段小插曲，若干年后项目移交到另外一个研发中心时，还需要有人专门去讲这段代码，否则接手人很难理解。再说回到之前，开发完后有一次和业务分析员再次聊起，说这个很不好实现但我啃下

来了等等，他说那可以不用隐藏，显示空白层也可以，真如果这样实现就简单多了，也好理解多了。事不大，但在职业生涯里印象深刻。需求不光是拆解，更可以讨论后寻找简单解决方案，而不是用自以为牛鼻的代码实现。以更合理的成本实现需求交付价值，这其实是用户故事里Negotiable的意义所在

作者回复: 多谢大年初二的学习与分享!

程序员的锤子是代码，到处找钉子是直觉。

2019-02-06



13



## 丁丁历险记

1 作者观点。绝大多数问题都是由于分解的粒度太大造成的，大块的需求不能谈取舍，但小的就比较合理。

读后思考，（yy 个例子，和人讲把头发剃光是不好谈的，但是谈谈把哪里剃一下，可以换来xxx 这个是好谈的）

2 INVEST 介绍。

Independent, 独立的

Negotiable, 可协商的

Valuable, 有价值的

Estimatable, 可估算的

Small, 小

Testable, 可测试的

读后思考，天啊，这么多的单词，这么详细的解释（罗里吧嗦），我是记不住的，自己yy 个故事去记忆吧。（为了方便，直接借用作者昨天的例子来扯需求好了）

google 整理术告诉我们，长期的记忆是一个encoding 过程，使用故事是一个非常好的套路（作者也同样说了这点，只是没站在大脑认知学的维度来谈而已）

step1: 在漫长的等待后，产品的用户登录登出需求出来了（出来了，出来了....）。做为开发，我第一时间对他的需求做了梳理含以下内容, 基于(Independent 独立性的)分解如下 并且分解得很(small 小)

手机号密码登陆（判断（不能为空，手机号格式，密码长度校验））

记住密码

第三方登录（微信扫码，微博登录，qq 登录）

登录时长为2小时

.....

登出

step2: 我拉来了产品,协商。（便于简化，这里只谈记住密码，和第三方登录）这写说明了，分解的粒度是（Negotiable 可协商的）

2.1记住密码这事，其价值是用来方便用户的（Valuable 价值判断），但现在，chrome,ff,360se等浏览



器 早以集成了这些功能，花了不少的时候，做出来的东西，并未额外给客户带来更多价值，所以，在我们这个mvp 版本中，这个先砍掉，我们多些思考，找出做这里的价值，再来开发如何。。。 (small)

2.2 第三方登录这事，（我们的客户特征，是一定有微信的，（在其它应用场景中，是直接使用了微信的例如群发通知，微信课评等...））所以，微博登录，qq 登录，对我们来说，是做了多余的事。

关于微信登录呢，在沟通说到 平台需要收集到用户的手机号，顺带方便客户手机登陆，所以在首次扫码时，提示下客户去做手机绑定，和直接使用。（毕竟我们不是流氓平台，动不动强制用户绑了手机才能用）

这个沟通，我们先砍了价值 我们通过沟通发现，我们把微信登录后手机号绑定这个需求，单独给提了出来 (small) 并且把微信登录这事 (Testable 可测试的) 的需求出来。

step3 梳理玩需求后，开始估算进度了。

微信登录这事吧 先拍脑袋 2 days (Estimatable) （我讲下如何拍出来的，我看了下文档 [https://developers.weixin.qq.com/doc/oplatform/Website\\_App/WeChat\\_Login/Wechat\\_Login.html](https://developers.weixin.qq.com/doc/oplatform/Website_App/WeChat_Login/Wechat_Login.html) 难度适中，考虑到开发之前没有做过类似开发，但这个早已被大量网站给实践出来了，先初步给两天吧。（基于dod 原则，我和程序员聊了下，给这么多时间，主要是 1 完成pc 端扫码登录的任务，并测试完成【紧急】。2 熟悉知识和习惯一种与第三方协同的开发模式【重要】我为认工程师不能只限于事务（完成1），还需要能力成长（完成2）（再讲个个人习惯，我会再预留半天时间，程序员写崩了后，我去快速填坑，小概率事件，做一个合适的预留，这里我多扯一句，要基于库尔贝勒交叉熵去适当预留资源，项目开发需要鲁棒性，别让极小概率事件把整个项目搞崩）其实这个 2days 的结论，也是由于分解的足够独立，且粒度足够小，才可以做出来估算。所以，我在后续估算工做量发现蒙b 时，就会去思考是否分解到位了。

3 作者推荐了两本书《User Stories Applied》和《Agile Estimating and Planning》。

做以下操作，复制 User Stories Applied 打开某东，搜索 User Stories Applied 先加入购物车

由于价格很感人 User Stories Applied ¥571.00 Agile Estimating and Planning ¥620.00

我登上download.csdn.net (vip ) 下载了对应的pdf，然后再上传至百度网盘（还是vip）

并分享出来 <https://pan.baidu.com/s/1jWbXqUX2kXJGJcyurGDh4w> 提取码：e28v

2019-11-09



11



行与修

我对需求的理解分两个层面：用户需求和开发需求。在尽可能全面了解客户意图的基础上突出价值，通过开发需求框定范围。以往会形成两份不同的文档，主要是考虑到双方的知识背景不同而有针对性的准备，至于开发需求的误差程度往往得不到客户的有效确认，因为客户不习惯以程序员思维和工作模式去阅读开发文档，常常会给基本上是这样、差不多吧、先这么着看看这样的反馈。所以我在想应该可以用一种更“活泼”的方式去提高双方的沟通效率，如果能够以故事的方式去撰写用户故事，把问题拆分说透，用一个能让对方身临其境界的场景故事去沟通，而非格式化的冷冰冰的文档去消除双方认知上的不足与分歧，这样除了可测试之外其他方面应该都能兼顾了吧。

作者回复: 程序员喜欢活泼点的东西 :)

2019-02-06



4



陈斯佳

愿上帝赐予我能拆分需求的智慧.....

作者回复: 嗯, 得练。

2019-07-31



3



WL

可衡量才可以讨论, 跟老师学习受益匪浅

2019-02-12



3



helloworld

进行开发之前先把需求理解好, 也就是要做什么, 然后进行需求的拆分, 也就是怎么做, 到这时候有些用户故事肯定会由于某些原因做不了, 就砍掉, 再然后对需求进行排期。所以拆分需求的过程也是梳理项目的过程。

作者回复: 这个理解的角度很不错!

2019-03-01



2



稳

老师, 今天的内容, 能不能理解为需要先将用户故事切分的足够小, 然后以此来需求合理性、工时评估、需求的降低标准等事情呢?

作者回复: 先将需求分成用户故事, 只有到达可以评估的大小, 你才能更好地理解, 才能开始做

后续的工作。

2019-02-06



👍 2



Jerry Wu

工作中，我也砍过不少的需求，和老师的思路类似，就是拆细需求，一个个地去砍。

比如，上节课的登录注册。我会这样说，如果完全按着这个需求来的话，肯定是完不成的，即使加班加点，最后也是一堆bug。

不如这样，图像验证码前期就够用了，短信验证码放到后一个版本吧？还有前期用户量不多，分布式也就放到下一个版本吧？

这样的话，我们的工作就是XXX，在这个时间内，还是能赶出来的。

作者回复: 是这个味道。

2020-04-22



👍 1



Phoenix

要看公司，很多公司很多需求是老板提的，老板和产品都是强势方，这种情况是没法砍需求的

作者回复: 你“认为”没法砍，那就真的没法砍了。

2019-02-15



👍 1



秦奋

“很简单，把它拆分成多个小任务，这样一来，每个小任务都可以在一个迭代中完成了。”这句话没看懂，拆分完之后需求并没有减少，为什么说小任务可以在一个迭代中完成呢？

2020-12-25



👍



一个帅哥

拆分业务需求有3个好处：砍掉实现成本高及体验不合理及价值不高的小点；方便后面的开发任务的排期；加深对需求的理解

作者回复: 这个理解很到位!

2020-06-14



一个帅哥

分解有两个维度：业务需求分解；开发任务分解

作者回复: 有了这个角度，就可以理解《软件设计之美》里提到的分离关注点了。

2020-06-14



Jerry Wu

这节课的核心问题：怎么砍需求？

把需求拆细，一个个地砍。

2020-04-22



Json Dumps

需求分解了，技术方案设定，还是不好确定开发时间。比如技术方案中遇到你不会的技术，或者需要预先熟悉现有系统的实现

作者回复: 后面讲了怎么面对不熟悉的技术，继续学。

2020-04-17





AdonisPeng

砍需求非常不错，扩展我的知识的广度。谢谢！

作者回复: 思路不受限，能做的办法就很多。

2020-03-25



小马哥Mar

开发估期不应该靠直觉，而应该基于可控的需求任务量，而要需求任务量可控，这个需求就应该足够小，足够明确

2019-08-20



陈斯佳

拆分需求，越小越好，这个是要不断打磨的一项技艺。做一个好的程序员，甚至一个好的项目经理，这应该是必不可少的技能。其实，这个方法都可以应用在我们生活当中。把我们生活看作一个个项目，把自己当作项目经理，去拆分一个个需求，慢慢的把它实现。

2019-05-18



Sudouble

这一节里介绍的用户故事，感觉和软件工程里的基于场景的需求建模很相似。明确需求是重中之重

作者回复: 用户故事只是描述需求的一种方式而已。

2019-02-15



