04 | 流程优化：怎样才能让敏捷、精益真正为我所用？

2019-08-30 葛俊

研发效率破局之道

进入课程 >

讲述：葛俊

时长 17:06　大小 15.67M

你好，我是葛俊。今天我们来聊聊怎样从流程方面来提高研发效能。

从这一篇文章开始，我们就正式进入研发流程模块了。在第 1 篇文章中，我与你强调了软件开发的最大特点在于它是一条非常灵活的流水线，因此提高研发效能的第一步就是优化流程。

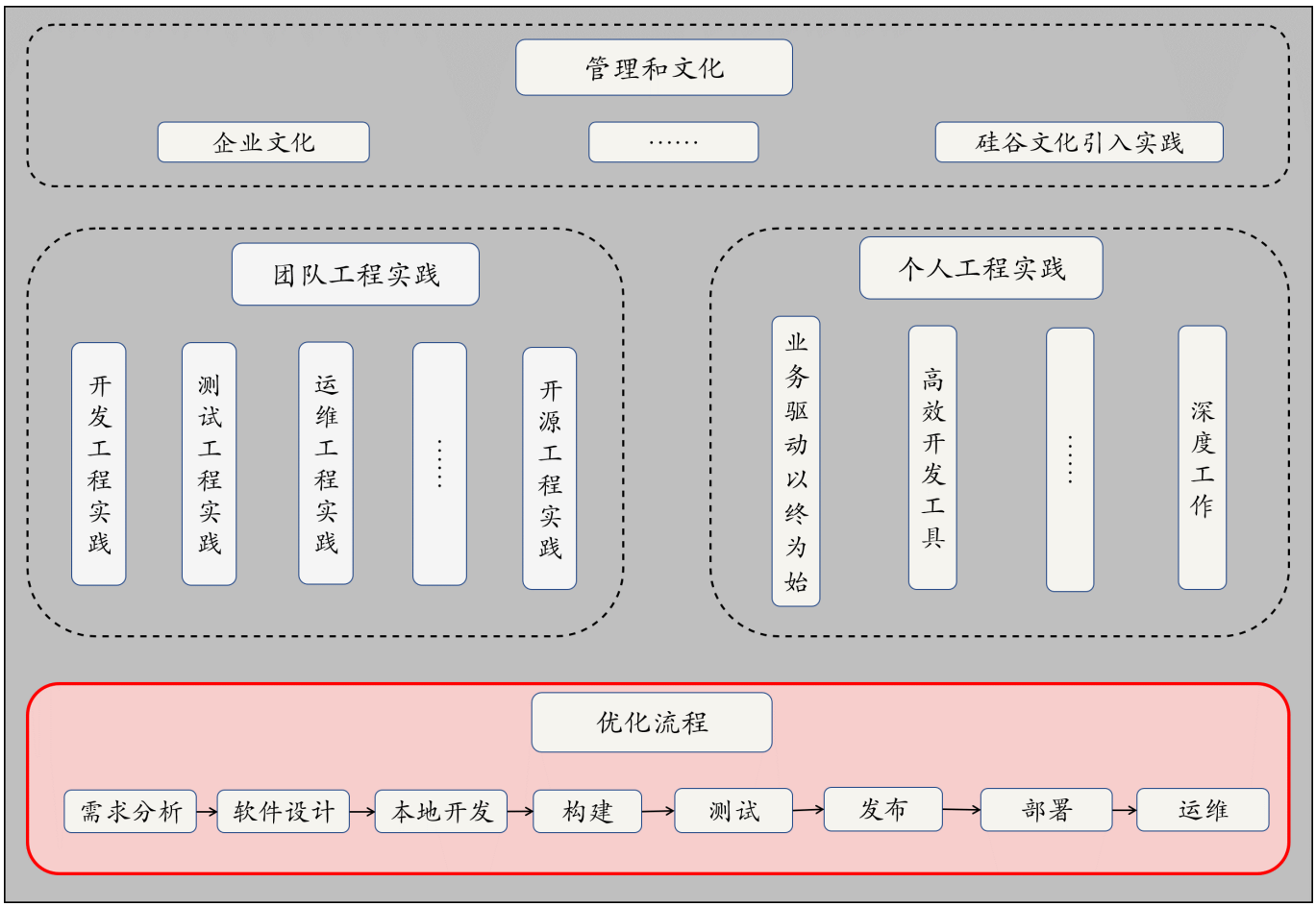

图 1 提高研发效能的第一步就是优化流程

因为优化流程会涉及方法论，所以我会先和你介绍高效实践方法论的关键要素。然后，我会按照目标、原则和实践 3 个层次，从抽象到具体，给你逐步讲解如何优化流程。

需要说明的是，在这一篇文章中，我不会与你大量讨论实践细节，而是把这些内容放在了后续的文章中。

如何实践方法论？

其实，在研发流程上，最不缺的就是方法论，从敏捷到精益再到看板，层出不穷。但，实施效果却不理想。尤其是敏捷，无论在硅谷还是在国内，绝大部分使用敏捷的团队实施得都不理想，导致这个概念的争议很大，Scrum 有时甚至成了贬义词。

相比之下，像 Facebook、Google 等高效能公司，并没有强调使用 Scrum、看板等工具，研发效能却很高。**这是不是说敏捷、精益这些方法论本身就有问题呢？**

事实上，虽然 Facebook、Google 这些公司没有明确提及敏捷、精益、看板这些方法论，或者没有严格地去使用 Scrum 等框架，但在开发流程中却实实在在地应用了这些方法论的

精髓。所以说，**方法论实施效果不好，关键在于没用好。**

在学习方法论的时候，我推荐使用类似美国著名作家、企业顾问西蒙斯 · 涅克（Simon Sinek）总结的 Why-How-What 黄金圈法则。

参见下图：最中间的一个圆是 Why，也就是这个方法论的目标，是要解决一个什么问题；第二个圆是 How，也就是这个方法论的基本原则、指导思想；最外层的圆是 What，也就是这个方法论的具体实践。

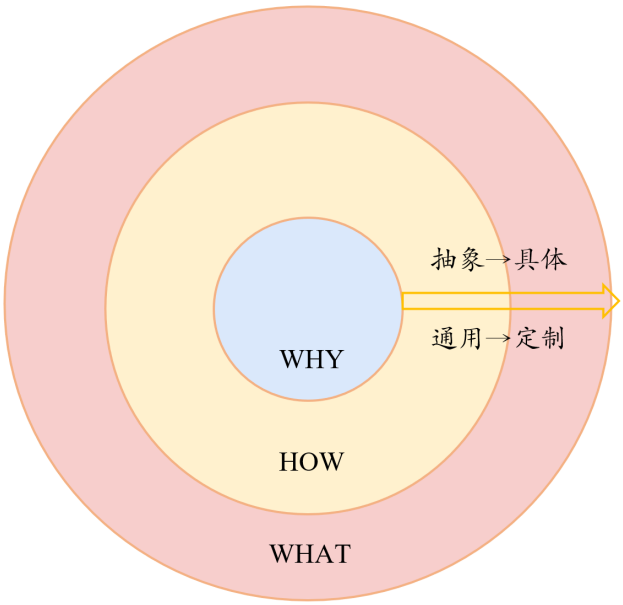

图 2 Why-How-What 黄金圈法则

这 3 个圆圈，从内向外，是一个从抽象到具体，从通用到定制的过程。

在使用一个方法论的时候，一定要从内往外看。

> 中心的目标一般错不了。比如，敏捷的目标就是快速应对变化。
>
> 原则的通用性就差一些，你需要在理解的基础上挑选适合自己的。比如，敏捷中有一条原则是“面对面交谈是最好的沟通方式”，就不一定适合你的团队。
>
> 具体的实践，就更要小心，切忌生搬硬套。

所以，我们必须首先深入理解这个方法论的目标和基本原则，然后根据原则因地制宜地选择具体实践。在选择具体实践时，往往需要在已有实践上做些修改才能达到效果，否则事倍功半。

以最容易出现问题的 Scrum 为例。敏捷的目标是快速应对变化，而 Scrum 就是用来服务这个目标的。但是，很多团队在使用的时候，严格照搬 Scrum 的具体方法，而“严格照搬”本身就已经违背了敏捷的目标。

相比起来，Facebook 的众多团队，“严格”使用 Scrum 的很少，却一直在大力优化管理、开发等流程来快速应对变化，最快找到并满足用户的最新需求。具体来说，他们很早就引入了 A/B 测试、灰度发布、每周定时全量代码部署等实践。这些都是和敏捷方法论相吻合的，也是 Facebook 业务成功的关键技术支撑。

在引入实践的时候，我的建议是逐步优化已有的开发流程和框架，甚至只给出原则，让团队成员逐步摸索并最终找到合适的方法。比如，你可以把一个核心原则通知给团队：“流程中总是有一个核心瓶颈。找到它，并解决它”。通过这样一个基本原则的指导，让具体实践逐步形成，对现有工作的影响，以及受到的阻力都会比较小。

了解了怎样使用方法论，接下来我和你一起看一组提高研发流程的目标、原则和实践。其实，这些并不是一套全新的方法论，而是基于 Facebook、Google 等高效能公司的实践，并结合对精益、看板、敏捷等方法论的思考，我得出来的一个总结。

目标一：寻找用户价值

以终为始地来看，我们最终目的是产生用户价值，生存下去。为此，我们经常需要主动去寻找最好的产品形态。只有方向找准了，流程产生的结果才能有效，才能产生用户价值。所以，**优化研发流程的第一步，就是提高寻找用户价值的效率。**

在这一点上，精益创业（Lean Startup）系统最有效。虽然名字里面有创业二字，也有较多商业模式设计和用户开发方面的内容，但它有两条原则值得作为开发流程的参考。

第一条原则是：衡量每一个时间段成果的标准，应该是价值假设方面的进展。也就是说，你的工作应该让你学习到如何更好地给用户提供价值，而不是开发了多少功能。

举一个极端的例子。比如，你的团队在一月份开发了 5 个功能，用户反馈都一般。这时，你看不出用户更喜欢什么东西，更讨厌什么东西。而在二月份，你们只开发了一个功能，预计很有用，但收到的用户反馈却是负面的，被迫连夜回退了这个功能。

那么，哪个月的成果更好呢？显然是二月份，因为它能明确告诉你，这个假设是错的，让你们在寻找产品符合市场吻合度（Product-Market-Fit）上前进了一大步。

第二条原则是：使用最小可行性产品（Minium Viable Product，MVP）来帮助学习。这里的关键点是，要以探索价值为出发点设计产品，最快地验证你的假设，功能要尽量少，能够使用就可以。具体的方法有数据驱动、A/B 测试、灰度发布等。

在 Facebook 工作的时候，我们开发一个功能时，都会提前计划要验证哪些假设，然后设计怎样收集数据才能够验证这些假设。这样一来，功能上线的时候就开始收集数据了，一旦发现功能不能提供足够的用户价值，就把这个功能下线，从而确保每个功能都是本着提供用户价值的目的去的。

同时，公司也会不断投资 A/B 测试等试验框架，让开发人员能够尽量容易地收集和处理数据。这种开发方式有一个名字，叫作度量驱动开发（Metrics Driven Development）。如果你想深入了解这种开发方式，可以参考 Uber 公司的这篇文章。

目标二：提高用户价值的流动效率

软件研发是一条流水线，里面流动的是一个一个给用户提供价值的功能。那怎么提高这条流水线的效率呢？也就是，如何提高用户价值的流动效率。

这里有 4 条比较直观的基本原则：

第一，让功能尽快地流动；

第二，让节点之间的联动更加顺畅；

第三，节点之间的融合；

第四，发现整个流程中的瓶颈，并解决它们。

接下来，我们分别看看这四条基本原则。

第一，让功能尽快地流动，说白了就是快速开发。问题发现得越晚，修复代价越高，这已经是常识。要做到快速开发，开发人员需要**尽量把功能拆分，同时做好一个提交之后尽快提交**。要做到这一点，关键原则是降低提交的交易成本（Transaction Cost）。

提交的交易成本指的是，每一个提交都需要做的额外工作。只有把这个工作量降下来，才能让功能拆分有价值，否则就会抵消掉拆分带来的好处。

具体工程实践包括提高本地构建速度，提供方便的自测环境、测试自动化、持续集成、定位问题提交自动化等。快速开发这个话题是开发高效的关键因素，我会在下一篇文章中与你详细讲述。

第二，让节点之间的联动更加顺畅，可以通过对关键流程的自动化、工具之间的网状互联，以及节点之间的融合来实现。在具体实践上，个人代码上线后，在和他人的代码集成时容易出现问题，这时就可以使用 CI/CD（持续集成 / 持续交付）流水线来自动化代码集成过程。

尤其是 CI，基本上对所有软件开发公司都很有价值。我们一定要尽量用上 CI。

另外，有些公司有从需求到开发到上线的一条龙流水线，也被叫作开发者桌面。国内的公司，尤其是大公司很喜欢这种方式。它的好处是，可以把很多底层的东西隐藏起来，容易上手。但它的缺点就是灵活性不够，毕竟难以用一条流水线做到高度的定制，来满足各种各样的高效开发。

开发者桌面的全景图如下所示。开发者桌面作为核心节点，连接其他所有工具，各个工具之间也有一些连接。

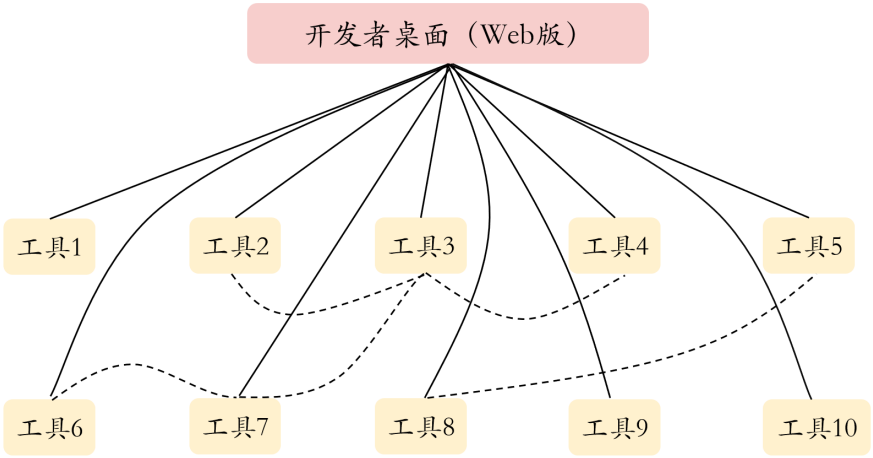

图 3 开发者桌面的全景示意图

在 Facebook 和 Google，并没有一个端到端的流水线，它们采取的方式是：**让这些流程中使用的工具通过开放的 API 进行一个网状连接，从而开发者可以灵活定义小范围的流程，高效工作。**在这个网状结构里面，关键节点会作为重要节点大量连接其他流程中的工具，公司对这些关键的、小范围的流程进行大量优化。

比如下面这张图片，一共有 11 个工具，互相之间连接很多，形成网状结构。其中工具 4 和工具 9 是关键节点，与很多工具有连接。

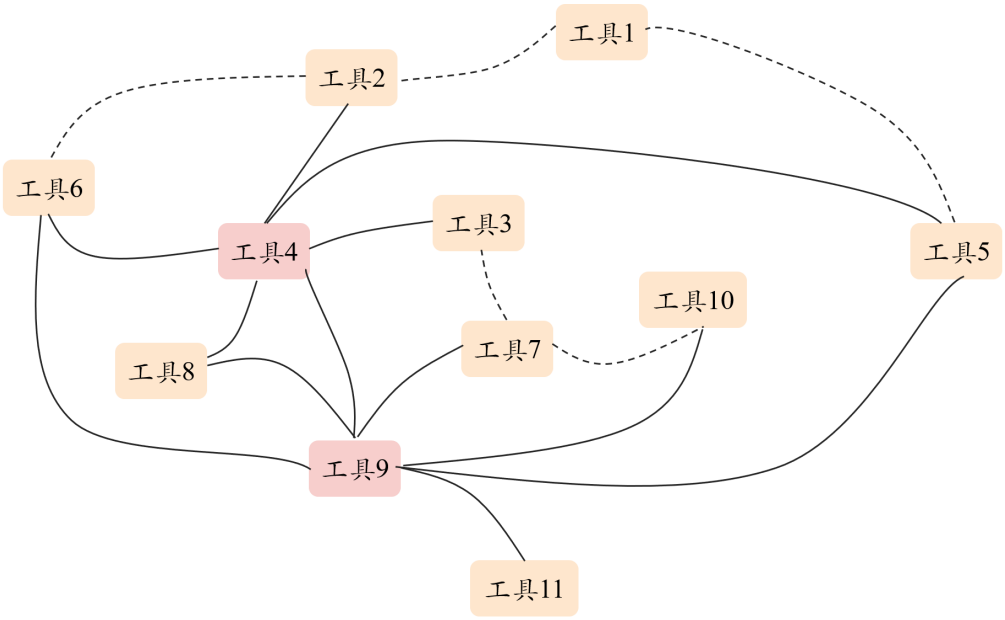

图 4 工具网状连接示意图

比如在 Facebook，代码审查工具 Phabricator 是一个关键节点，可以支持代码审查、代码静态检查、单元测试、集成测试、前后端联调等关键开发工作。这一组工作就是上面提到的一个关键的、小范围的流程。

最近几年由于 IDE 的发展，Nuclide 成为了一个新的重要节点。开发者可以在 Nuclide 里直接进行高效的代码开发，以及代码审查等一系列工作。类似地，Google 的 WebIDE 也是工具链的一个重要节点。

从我个人的经验看，这种网状结构提供的灵活性，对提高开发效率和提升开发体验，都有积极的促进作用。

第三，节点间的融合，也是为了保证节点间的联动顺畅。也就是模糊节点间的边界，让功能在节点之间的流动更顺畅。在这一方面，我见到比较有效的方式是：**职能团队提供平台和工具，让全栈工程师能够自己处理端到端的工作**。比如，测试团队提供测试平台和工具，运维团队提供运维平台和工具，这些平台和工具可以通过服务化自助使用。

比如，在 Facebook 没有专职的测试人员，只有测试工具团队，运维人员也很少，主要是提供大量的工具让开发人员能够自己进行测试、运维的基本工作，实现开发人员在“开发、测试、运维”这几个步骤上的全栈工作。因为开发人员对自己开发的功能最为熟悉，所以这

种端到端的全栈工作方式，可以让开发和调测效率达到最高。在国内，我看到阿里云效也在推荐这种方式。

第四，发现整个流程中的瓶颈，并解决它们。约束理论创始人艾利·高德拉特（Eliyahu Moshe Goldratt）在 20 世纪 90 年代，提出了解决约束的聚焦五步法，至今仍然适用软件研发这个价值流体系。具体步骤如下：

第一步，找到系统中的瓶颈；

第二步，最大限度地利用（Exploit）瓶颈，尽量通过提高效能的办法解决瓶颈；

第三步，让企业的所有其他活动都让步于瓶颈改善工作；

第四步，打破（Elevate）瓶颈，如果第二、三步无效，就通过给瓶颈节点增加资源的方法来解决瓶颈；

第五步，重返（Repeat）第一步，找出新的瓶颈，持续改善。

具体的实践有可视化和复盘。

在可视化方面，粘上便利贴的白板，或是像 Trello 那样的电子看板就是很好的工具。

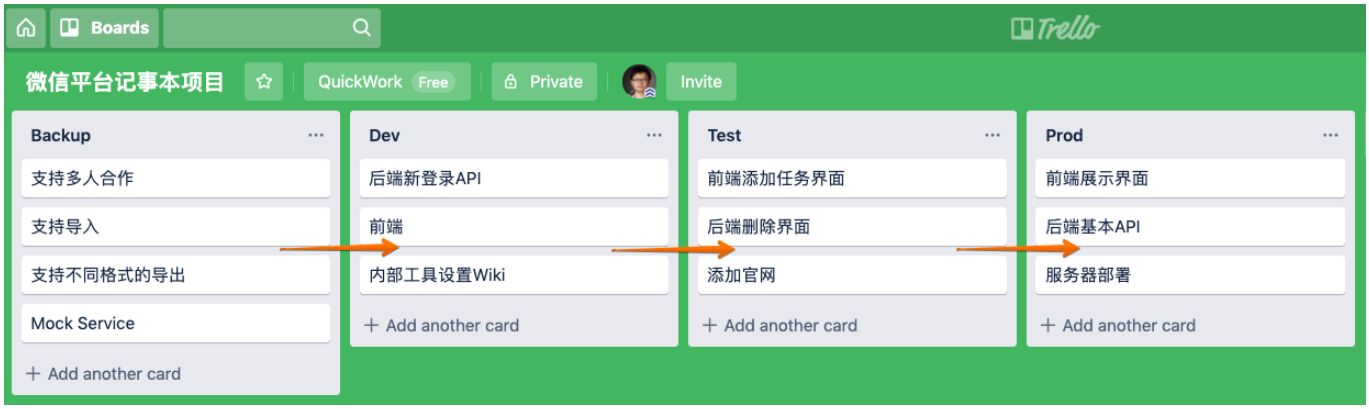

图 5 可视化实践示意图

这里我需要强调一下，我们一般把这种任务卡片化的系统叫做看板，但它并不是真正的“看板”，只是一个任务可视化的面板而已。

真正的“看板”是信号卡片，用来表示流水线系统可以增加新的工作项了。后面我会详细介绍。通过任务可视化，我们可以直观地看到哪一个环节的任务特别多，卡住了，也就是瓶颈在哪儿。另外，前面文章提过的累积流程图，也是发现问题的一个有效工具。

另外，统计图表和仪表板也是很不错的工具。在 Facebook，任务管理系统、部署系统、代码审查系统的 API 是开放的，各个团队可以方便地查询数据制作图表。在仪表板方面，2014 年的时候就有 4 个系统可使用。大部分的统计图表和仪表板，都是非工具团队自己开发出来并最终推广全公司使用，定制功能也很强。

在复盘方面，Facebook 做得也很好。他们有一个 SEV 复盘系统，用来记录公司发生的重要事故进行复盘。每个团队也会不定期地复盘，以定位瓶颈问题并提高。

举一个复盘方法的例子。我在国内某公司遇到一个问题，就是团队在协作上配合不积极，出了问题不是先去解决问题，而是先想方设法甩锅。为解决这个问题，我引入了线上事故回溯讨论会，每两周一次，对发生的事故进行讨论，重在根因分析和以后如何避免，并事前强调目的不是追责。因为，每个故障分析都能暴露出藏在深处的问题，对提高产品质量和团队间的信任效果都很好。

小结

优化流程，是提高研发效能的第一步。我们应该按照目标、原则和具体实践的顺序学习和使用各种方法论，选择和配置最适合自己团队的工程实践。

我基于 Facebook、Google 等高效能公司的实践，并结合对精益、看板、敏捷等方法论的思考，给你介绍了“寻找用户价值”和“提高用户价值的流动效率”两条目标，以及它们对应的 6 条原则和若干具体实践。

从我的经验来看，高效的研发工作流，对产品质量和研发速度至关重要，同时可以提升员工的幸福感，进而保证持续的高效产出，提高研发效能。

Facebook 在流程方面的实践，给我最大的一个感受就是，以实用主义的态度，从原则出发，灵活优化流程。在 Facebook 的几年时间里，我并没有听到很多新方法论的时髦术语，但是公司的很多实践却和这些方法论的原则是一致的，甚至是超前这些方法论的。

比如，在 DevOps 流行前的很多年，Facebook 就开始了打通开发和部署的工作，以及推行全栈工程师的工作方式。

从实际出发、以终为始的实用主义，是我从 Facebook 学到的高效研发的最重要原则，没有之一。

思考题

你理解的精益看板，跟文中提到的任务可视化白板一样吗？你知道它们之间的关系吗？

感谢你的收听，欢迎你在评论区给我留言分享你的观点，也欢迎你把这篇文章分享给更多的朋友一起阅读。我们下期再见！

上一篇　03 | 效能度量：如何选对指标与方法，真正提升效能?

下一篇　05 | 代码入库前：Facebook如何让开发人员聚焦于开发?

精选留言 (10)

写留言

囧囧冰淇淋
2019-08-30

一个问题：大公司流行的方法论，为什么到我们身上就没用了？

1.原封不动的照搬。
2.文中提到的WHY?HOW?WHAT?没有思考

WHY=为什么会发生这个问题？原因在哪？...

展开

作者回复: 关于思考题，有空可以看一下这本书理解看板：
中文版：https://book.douban.com/subject/25788807/
英文版：https://book.douban.com/subject/5350839/

1 3

小树苗
2019-09-01

我会在看板中针对每一道工序都设定两个环节，一个是等待中，一个是处理中，处理完的就到下个工序的等待中或处理中。我们会合理控制处理中的在制品数量，于是等待中就会有任务堆积，当某个工序的等待中的任务数量多起来了，我们就会去分析这中间有什么问题，一般问题分为三类，资源数量(人不够多)，资源能力(人不够强)，资源协同(流程机制方面的的问题)。...

展开

作者回复: 赞！这个是正确使用看板的姿势。工具麻烦一点不要紧，慢慢慢提高，关键是方式正确。

至于具体工具，Trello这样的应该可以实现你们实用的这种工作流呀？

推荐阅读：4 BENEFITS OF WIP LIMITS（https://leankit.com/learn/kanban/benefits-of-wip-limits/）

1

舒偌一
2019-08-31

围绕提高用户价值的目标出发，尽快让用户验证我们现在做的事是正确的，为了提高速度，利用工具让节点通达，使用任务看板和进度工具找到瓶颈。我们使用任务看板和员工日志，系统识别哪个任务超出预期了。我们现在的瓶颈是需求不细、用户验证慢，导致连带返工。

展开

作者回复: > 我们现在的瓶颈是需求不细、用户验证慢，导致连带返工。
你们有什么提高计划吗？

1 1

Polo
2019-08-30

看了下面的评论，有所老师总结的好的，有说有些东西太形式主义了的

大学学的东西，出生社会工作后才发现，几乎全是「形式主义」，那大家怎么把这个社会向前推动的呢?

...

展开

作者回复: 越灵活的东西，越需要掌握本质。

1 1

刘培培
2019-08-30

最近正在把部门所有的移动 App 接入基本的 CI，直接用 gitlab ci 工具最简单，之前试过 Jenkins 可把我烦死了。

展开

作者回复: Jenkins 比较麻烦，但是比较成熟，跟其他工具集成比较完备。 GitLab CI比较新，好用很多，但是有一些场景应该不支持（新的东西的通病）。如果你们的场景能使用GitLab CI，那就最棒了。

2 1

栗芳凯
2019-08-30

无招胜有招

展开

1 1

Dragoonium
2019-08-31

我司的开发瓶颈，在于中间件。每个产品都要用到少则五六个多达十几个的中间件，那那些中间件都是在不同的国家开发的，质量良莠不齐，互相的依赖性又很强，幸好有CI的存在，对于新安装的产品，中间件表现还行，但对于已经安装好的产品在升级时，由于组件

版本繁多，升级路径扑朔迷离，基本上产品的每个版本，要花费六成的资源在中间件的整合与升级上，团队苦不堪言。

展开

作者回复: 这些组件都是在不断更新吗?

安装好的产品升级，总是使用最新的版本，还是有的情况继续使用旧版本?

Robert小七

2019-08-31

云效目前自身的问题还是挺多的，但是对于流程方面的建设确实不错！比如老师提到的复盘系统，云效有与之对应的故障台！云效开发模式中，开发利用云效平台拉取分支，编码后可以一键构建部署，之后进入运维系统查看是否有问题，自行快速回滚！云效商业化后，对很多传统公司都起到了一个很好的效能提升！

展开

作者回复: 云效对推动国内企业对研发效能的重视度，起到了很重要的作用

囧囧冰淇淋

2019-08-30

软件开发会用到钉钉么？如果用到，会要求写日报，周报，月报不？写短了被批评，长了也很少人看，大家都很痛苦的那种？

展开

作者回复: 日报，周报，月报如果用得好，可以达到至少两大作用：1. 沟通信息并电子化；2. 促进自己和团队计划、总结、反思

> 写短了被批评，长了也很少人看

主管要明确这些周报的作用。让花在写报告的时间花得值得。

2

许童童

2019-08-30

老师总结得很好，我也很喜欢Facebook这种实用主义，很讨厌形式主义，我们就冲着目标

去，完成目标，达到目标，遇到问题解决问题，而不是为了敏捷而敏捷，站会要不要开怎么开，都可以根据团队的需要去设定。寻找用户价值、提高用户价值的流动效率，这两个目标是今天的重点，要记在心里。

展开

作者回复: 👍👍👍