01 | 效能模型：如何系统地理解研发效能？

2019-08-23 葛俊

研发效率破局之道

进入课程 >

讲述：葛俊
时长 16:50　大小 15.43M

你好，我是葛俊。今天，我来和你聊聊什么是研发效能，以及研发效能的模型，这些内容是理解整个专栏的基础。

今年的 3 月 26 日，一位昵称为 996icu 的用户，在 GitHub 上创建了 996.ICU 的项目，自此 996 这个话题被推上了风口浪尖。目前，这个项目已经拿到了 24 万多颗星。朋友们也常常问我：硅谷的公司有没有 996？

其实，在硅谷，很少有公司要求 996。不过，在初创公司，因为业务紧张、同事间的竞争，加班也很常见。但是，硅谷和国内的公司有一个很大的区别，就是硅谷的公司一般是任务驱动，只要完成任务就行，不管你花了多少时间。而国内很多实行 996 的公司不仅仅是要求完成任务，更强调工作时长。但其实，专注时长的这种操作在软件开发行业是不合理的，因为长期加班不能保证持续的高效产出。

从我以及身边许多开发者的经验来看，每天能够高效地产出代码五六个小时，已经相当不错了。短期突击加班会有效果，但如果长期加班，通常效率、质量会下降，产生了 Bug 就要花费更多的精力去修复。如果这些 Bug 发布到了用户手上，损失就会更大，得不偿失。

长期加班还会出现无效加班的结果。比如，有个朋友在一家国内一流的互联网公司工作，据他反馈，公司实行 996，很多人加班其实是磨洋工，低效加班非常明显。可想而知，其他推行 996 工作制的公司，大概率也会存在这种问题。

那么，长期加班效果不好，面对激烈竞争，我们到底应该怎么办呢？在我看来，这个问题还是要从软件开发本身的特点来解决。

为什么要关注研发效能？

软件开发是一个创造性很高的过程，开发者之间的效率相差很大。就比如，10x 程序员的生产效率可以达到普通开发者的 10 倍。其实，不仅是个人，团队间的效率相差也很大。**所以，相比工作时长而言，公司更应该关注的是研发效能。**

接下来，我们再回顾一下我在开篇词中给研发效能的定义：

> 研发效能，是团队能够持续为用户产生有效价值的效率，包括有效性（Effectiveness）、效率（Efficiency）和可持续性（Sustainability）三方面。简单来说，就是开发者是否能够长期既快又准地产生用户价值。

硅谷的很多知名公司，比如 Google、Facebook、Netflix 等，在研发效能上做得很好，是研发效能的标杆。这，也是它们业务成功的重要因素。

以我在开篇词中提到的 Facebook 的部署上线流程为例。Facebook 在 2012 年达到 10 亿月活的时候，部署人员只有三个，达到平均每个人支撑 3.3 亿用户的惊人效率。举个形象的例子，如果全中国每一个人都使用 Facebook，那最多只要 5 个部署人员就够了。

Facebook 做到这一点的基础，就是不断提高研发效能。还是以上面的部署流程为例，原来的部署已经非常高效，但是在 2017 年，Facebook 又引入了持续部署，做了进一步的优化，实现了更高效率的部署上线。试想一下，如果 Facebook 选择堆人、堆时间的话，那需要增加多少人、多少加班才能做到呢？

所以在我看来，如果研发效能提不上去，单靠加人、加班根本不可能解决问题。

注重研发效能的另一个巨大好处，是开发者能够聚焦产出价值，更容易精进自己的技术。于是，团队容易建立起好的氛围，进而又能促进生产效率的提高，形成良性循环，支撑持续的高效开发。

所以，国内公司近一两年越来越注重提高研发效能，许多公司甚至专门成立了工程效率部门。但是，在真正开展研发效能提升工作时，它们却常常因为**头绪太多无从下手，或者对方法了解不够，导致画虎不成反类犬的效果**。这，又和软件研发的高度创造性和灵活性紧密相关。

自软件行业诞生起，开发者们就发挥聪明才智，不断创造新的方法、流程和工具来适应和提高生产效率。互联网产业爆发以来，这一趋势更是明显：从最初的瀑布研发流程到敏捷到精益，从持续集成到持续发布到持续部署，从实体机到虚拟机到 Docker，从本地机器到数据中心再到云上部署，从单体应用到微服务再到无服务应用。新的工具、方法，可谓层出不穷。

面对如此多的选择，如果能处理好，则开发体验好，产品发布速度快，研发过程处于一个持续的良性发展情况。但处理不好，就会事倍功半，出现扯皮、重复劳动、产品质量不好、不可维护的情况。

微服务的不合理引入就是一个典型的例子。自从亚马逊（Amazon）成功大规模地应用后，微服务逐渐形成风潮，很多公司在不清楚适用场景的情况下盲目采用，结果是踩了很多坑。

比如，我见过一个初创公司，在业务还没开展起来的时候，一上来就采用微服务，因为没有要求一致的技术栈，也没有限制服务的大小，于是开发人员怎么方便怎么做，只考虑局部优化而忽视了全局优化。半年下来，20 人的开发团队搞出了 30 多个服务和 5 种开发语言。服务之间的调用依赖和部署上线越来越复杂，难以维护，每次上线问题不断，经常搞通宵才能让服务稳定下来。同时，知识的共享非常有限，有好几个服务只有一个人了解的情况，一旦这个人不在的时候这个服务出现问题，解决起来就基本上成了“不可能的任务”。这样的错误使用微服务的公司非常普遍。

那，到底怎样才能有效地提高研发效能呢？硅谷的业界标杆公司又是怎么做到高效能的呢？

只有深入研发活动的本质，才能提高效能

我的建议是，提高研发效能，需要**深入了解研发活动的本质，从纷乱的表象和层出不穷的方法中，看到隐藏的模型，找到根本原则**，然后从这些原则出发，具体问题具体分析，找到合

适的方法。这样做的原因是，软件研发很灵活，在实践的时候总会见到不一样的状况。越是灵活的东西，就越需要理解其本质，这样才能做到随机应变。

在 Facebook 的时候，我们做事时都遵循一些基本原则。比如，有一个原则是“**不要阻塞开发人员**”，贯穿在公司的很多研发和管理实践中。接下来，我给你举两个具体的应用场景，来帮助你理解这个原则。

第一个应用场景是，本地构建脚本的运行速度要足够快。开发人员在自己的开发机器上写完代码之后，都要运行这个脚本进行构建，把新做的改动在自己的开发机器沙盒环境上运行起来，以方便做一些基本检查。

这个操作非常频繁，所以如果它的运行时间太长，就会阻塞开发。因此，确保这个脚本的快速运行就是内部工具团队的一个超高优先级的任务。我们对每次脚本的运行进行埋点跟踪，当运行时长超过 1.5 分钟后，我们就会停下手中的工作，想尽一切办法给这个本地构建加速。

第二个应用场景是，商用软件的采购。对一定数额下的软件购买，开发人员可以自行决定，先斩后奏。而且那个数额还蛮高的，覆盖一般的软件完全没问题。

我个人就经历过两次在晚上加班时，要购买一个商业软件的情况。如果等主管审批，就需要到第二天。但，因为公司信任工程师能够在这样的情况下做出利于公司的决定，所以我可以直接购买并使用。这样一来，除了能提高这几个小时的开发效率外，更重要的是，我觉得自己拥有信任和权力，工作积极性更加高涨。

这两个应用场景差别很大，却都是基于“不要阻塞开发人员”这个原则。Facebook 之所以会有这个原则，正是因为它认识到了，**开发流程的顺畅是生产优质软件的关键因素，只有这样才能最大程度地释放开发者的创造性和积极性。**相比之下，很多公司更注重强管理下的流程和制度，而忽略了开发的顺畅，结果是开发人员工作起来磕磕绊绊，又谈何高效率呢。

看到这里你会说，我已经很清楚地知道，要想提高研发效能，就必须理解软件开发的本质。那么，软件开发的本质到底什么呢？

接下来，**我就和你探讨一下软件开发的本质特点，然后基于这些特点，搭建出一个研发效能的模型，希望你可以灵活运用，找到提高研发效能的主要着力点。这个思路，将是贯穿整个专栏的主线索。**

研发效能的模型是什么？

在我看来，**软件开发本质上就是一条超级灵活的流水线。**这个流水线从产品需求出发，经过开发、测试、发布、运维等环节，每一个环节的产出流动到下一个环节进行处理，最后交付给用户。

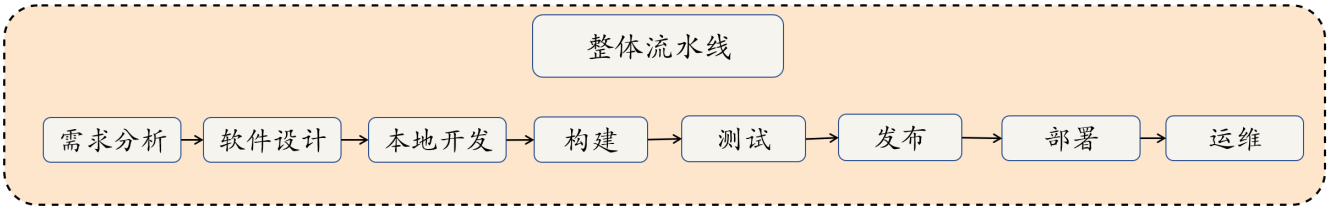

图 1 软件开发的流水线

另外，这条流水线的每个环节都还可以细分。比如，本地开发环节可以细分为下面几个部分：

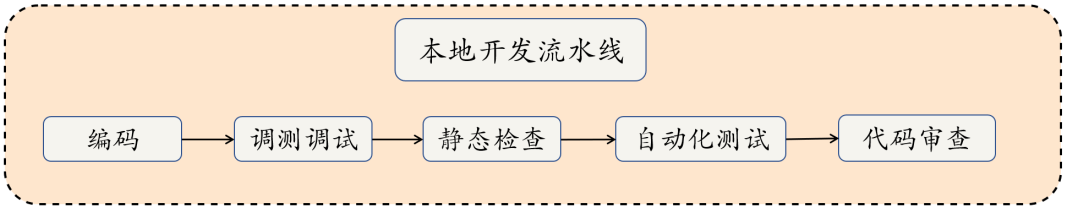

图 2 本地开发流水线

这种流水线工作方式，在传统的制造业中很普遍，也已经有了很多经验和成功实践。最典型的就是汽车生产中使用的丰田生产体系（Toyota Production System，TPS）。所以，**我们可以参考传统制造行业的经验来提高效能。**

事实上，瀑布模式就类似于传统流水线的处理方法：它强调每个环节之间界限分明，尽量清晰地定义每一个环节的输入和输出，保证每一个环节产出的质量。但，**和传统制造业相比，软件开发又具有超强的灵活性，体现在以下四个方面。**

1. 最终产品目标的灵活性。传统流水线的目标确定，而互联网产品的最终形态通常是在不断地迭代中逐步明确，相当于是一个移动的标靶。尤其是最近几年，这一灵活性愈发明

显。比如，在精益开发实践中，常常使用 MVP（最小可行性产品，Minimal Viable Product）来不断验证产品假设，经过不断调整最终形成产品。

2. 节点之间关系的灵活性，比如流水线上的多个节点可以互相融合。DevOps 就是在模糊节点之间的边界，甚至有一些实践会直接去掉某些环节，或者融入到其他环节当中。
3. 每个节点的灵活性。每一个生产环节都会不断涌现出新的生产方式 / 方法。比如测试，最近十多年就产生了测试驱动开发、Dogfood（狗粮测试）、测试前移等方法；最近又出现的测试右移，开始强调在生产环境中进行测试。
4. 每个节点上的开发人员的灵活性。跟传统制造业不同，流水线上的每一个工作人员，也就是开发者，都有很强的灵活性，主要表现在对一个相同的功能，可以选择很多不同的方式、不同的工具来实现。

   比如，之前我在 Facebook 做后端开发的时候，同样一个代码仓，有的同事使用命令行的编辑环境 VIM/Emacs，有的使用图形界面 IDE，有的使用 WebIDE；在实现一个工具的时候，大家可以自己选择使用 Python 还是 Ruby 还是 PHP。这其中的每一个选择，都很可能影响效能。

基于这些特点，我们可以从以下四个方面去提高研发效能。

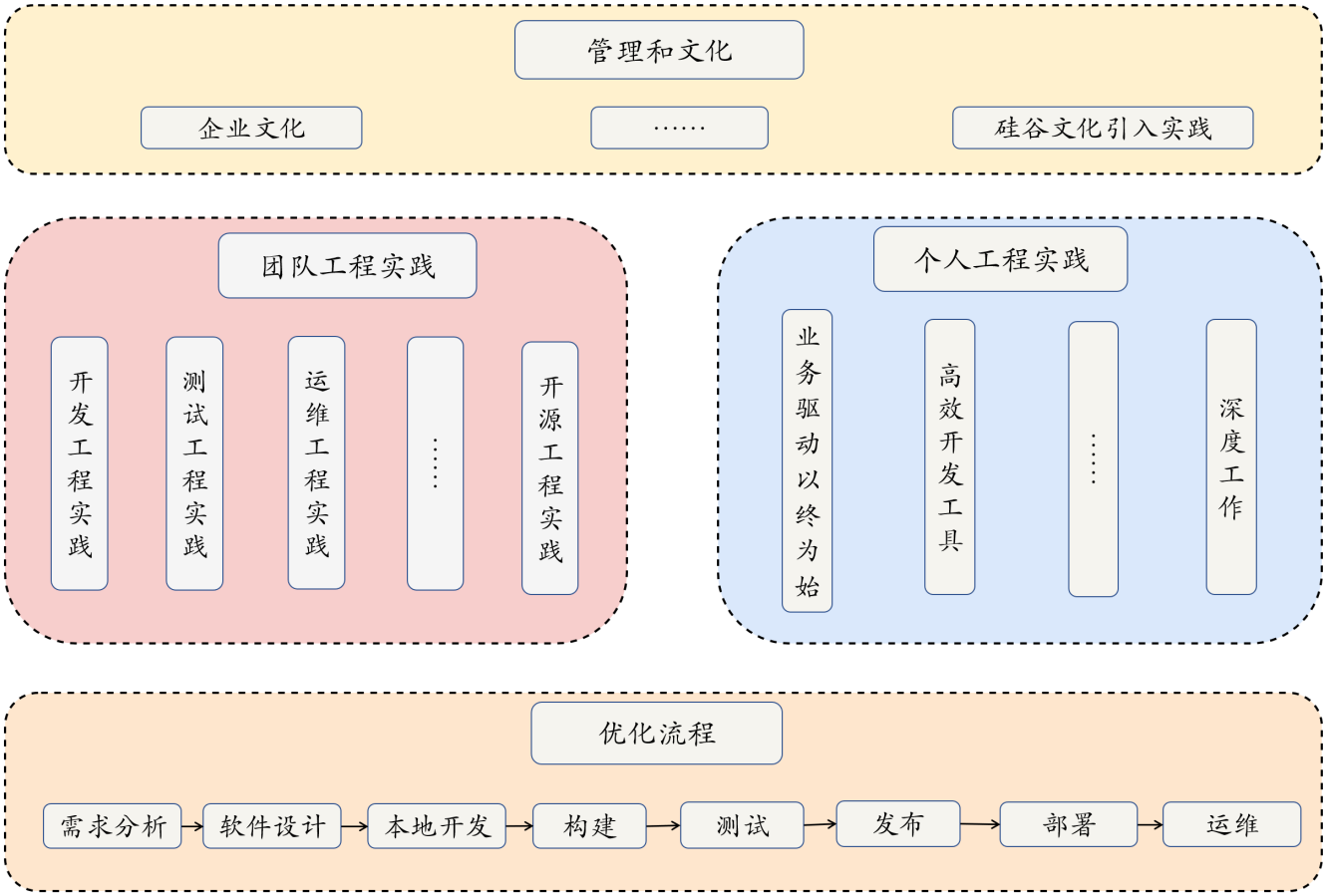

图 3 研发效能模型

1. 优化流流程，主要针对特点 1 和 2，也就是最终产品目标的灵活性和节点间关系的灵活性，进行优化。具体来说，针对最终产品目标的灵活性，主要是提高流程的灵活性，让它能聚焦最终产生的用户价值，以终为始地指导工作，击中移动的标靶。而针对节点之间关系的灵活性，则主要聚焦流水线的顺畅，以保证用户价值的流动受到的阻力最小。
2. 团队工程实实践，主是针对特点 3，也就是每个节点的灵活性进行优化，聚焦每一个生产环节的工程实践进行提高。
3. 个人工程实实践，主是针对特点 4，也就是每个节点上开发人员的灵活性，来提高个人研发效能。争取让每个开发人员都能适当地关注业务、以终为始，同时从方法和工具上提高开发效率，实现 1+1>2 的效果。
4. 文化和管管理。任何流程、实践的引入和推广，都必须有合理的管理方法来支撑。同时，文化是一个团队工作的基本价值观和潜规则。只有建立好文化，才能让团队持续学习，从而应对新的挑战。所以，要提高效能，我们还需要文化和管理这个引擎。

优化流程、团队工程实践、个人工程实践以及文化和管理，就是我们提高研发效能需要关注的四个方面，也就是我们所说的研发效能模型。

在接下来的文章中，我会以这个模型为基础，从以上这四个方向与你介绍硅谷公司，尤其是我最熟悉的 Facebook 的成功实践，并着重向你讲述这些实践背后的原理。因为只有理解了这些原理和原则，我们才有可能在这个超级灵活和高速发展的软件开发行业里见招拆招，立于不败之地。

总结

在今天这篇文章中，我和你再次强调了研发效能是团队能够持续为用户产生有效价值的效率，包括有效性、效率和可持续性三方面。

而关于如何提高效能，我的建议是深入了解研发活动的本质，从纷乱的表象和层出不穷的方法中，看到隐藏的模型，找到根本原则，然后从这些原则出发，具体问题具体分析，找到合适的方法。

最后，我借鉴传统制造业流水线的形式，并结合软件开发的特定灵活性，总结出了研发效能的模型，主要包括优化流程、团队工程实践、个人工程实践、管理与文化。专栏的后续内容，我将分为这四大模块，帮助你提高团队和个人的研发效能。

其实，研发效能对于硅谷的公司和个人来说，已经是常识，而我在美国工作的 10 多年，也已经习惯了这种工作方式。回国之后，我深入了解了国内的开发情况，对效能关注不到位的

现状颇为吃惊。因为这不符合软件开发这个行业的基本特性，也限制了国内软件行业的发展。

同时，我也对国内开发人员的生存环境，感到些许遗憾。本来软件开发是一个可以充分发挥创造性的、有趣的行业，技术的发展有无尽的空间。但是，开发人员却常常被业务拖着跑，技术发展和创造性都被限制住了。另外，因为拼时长的做法，也伤害了开发者的身体健康和家庭关系。

正是因为这些惊讶和遗憾，我将这十几年对研发效能的理解与实践，做了一次系统梳理，于是就有了今天提到的研发效能模型。我希望这种系统化的模型方法，能够为国内软件团队的效能提升提供一些参考和引导，也希望能够提升开发者个人的技能，以节省出时间来体会研发的快乐，提高生活的幸福感。

思考题

你有见过 996 对团队和产品造成伤害的具体事例吗？

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　开篇词 | 为什么你要关注研发效能？

下一篇　02 | 效能度量：效果不好甚至有副作用，怎么回事？

精选留言 (18)

写留言

小树苗
2019-08-23

关于研发效能模型，听了还是很有感触的。
首先关于MVP，就有一个很大的困扰，MVP很依赖产品经理对产品的定义，产品经理往往害怕一些细小功能点的有无会伤害产品和业务，总想第一个版本上去就是完美的，结果就导致新产品第一个版本就是个大家伙，交付周期就会很长，夜长梦多，接着就是需求变更，其他高优先级需求插入，结果就是交付周期极其的长，大家又要加班，恶性循环。...
展开

作者回复: > ...总想第一个版本上去就是完美的...
那就是没有真正理解精益开发，精益创业，没有理解MVP。如果有条件的话，可以想办法提高公司成员对MVP的理解。

> ...流程各个环节却耗时很长...是否舍得花成本去解决这些隐藏在流程背后的问题，又依赖于组织文化...
还有领导的意识。

4

Polo
2019-08-23

1. 用记事本编码和用IDE编码，谁的效率高?

2. 用微信文件传来传去和云端协同办公，比如石墨文档，谁的效率高？

3 . 工作期间，单屏幕工作和分多屏幕，谁的效率高？...
展开

作者回复: 同意同意。我个人一直对工具比较执着。花了很多时间在上面 ：）

不过使用工具也有一个合适度的问题。我曾经就花了太多时间去做不成熟的优化（premature optimization)。一般premature optimization是指架构方面的，但是实际上在使用工具上也一样。

我现在的实用工具原则是：留意平时工作，在重复比较多的工作部分，花一些时间去找工具进行优化。随时注意投入产出的比较。

3

寒光
2019-08-23

研发效能的提升，对团队成员的技能要求应该是什么样的呢？对于国内某些大厂，还是倾向于自上而下，层层分解。这样，最后真正写代码的，可能就是些初级技能的开发者，他们是有非常量化的代码指标的，实现功能已经不错了，还让他们有创造性，要求太苛刻了。不管怎么批判，这就是现实，如何突破？因为我们不可能要求团队的成员都具备硅谷的水平，在这些国内的现状下，我们的突破点在哪里呢？

展开

作者回复: 我建议作为管理者，应该在做业务目标的同时有一些技术目标。提高团队的效能就是技术目标中的一种。当然很可能需要说服更上一层领导，让他了解并支持技术目标。

不一定需要成员有大的创造性。一开始更重要的可能是主动性。可以从绩效等方面鼓励帮团队提高效能的行为。

2 2

yasuoyuhao
2019-08-23

公司目前屬於小團隊，十幾來人，部分人士效率特別差，即使引進了自動化測試，自動部署，在撰寫商業邏輯面依然是非常緩慢。經由分析原因可能如下：

1. 項目需求不明確
2. 開發人員對於流程不熟悉...

展开

作者回复: > 部分人士效率特别差

这些人员背景如何？没有经验的新人吗？还是有经验但是效率不好？

1 2

Jxin
2019-08-23

1.个人无所谓加班，入门晚，学习和勤勉已经养成习惯。但无脑的做业务需求的加班我是拒绝的，因为业务需求干大半年后，再无脑做成长太有限。而重构核心业务逻辑，重构项目分层，梳理业务图谱，写自动化脚本。为这些将来能节省时间的方面去加班去投资时间，我是很乐意的。
2.遗憾的是，你前面加班，把时间投在提高效能上，而后面自己效率上去没加班，引起了...
展开

作者回复: https://github.com/svenstaro/genact 了解一下？😇

1

XUN
2019-08-23

老师好！敏捷开发在软件开发领域有很系统的体系思想和实践方法。但是，源自丰田生产的精益思想，如何用来优化软件开发流程、如何借以提供工程效率一直半知不解，即精益开发，有体系化的参考资料推荐吗？谢谢

作者回复: 这本书不错：
中文版：https://book.douban.com/subject/25788807/
英文版：https://book.douban.com/subject/5350839/

1

李双喆
2019-08-23

是效能低才导致加班，还是加班导致效能低？
展开

作者回复: 这是一个很好的考虑问题的角度。我觉得这是一个恶性循环：
1. 加班之后发现有一些效果
2. 较多地使用加班
3. 加班导致性效能降低。产品质量下降
4. 为了赶上进度，提高产出，又觉得加班是一个办法，回到步骤2

1 1

舒偌一
2019-08-26

期待葛老师研发效能模型落地实践；

期望葛老师多考虑小公司的落地条件！

夜空中最亮的星（华仔...
2019-08-26

打个卡

展开

囧囧冰淇淋
2019-08-25

运营流程
需求分析→产品设计→拍摄→页面设计→客服培训→营销推广→销量提高和维护

目前想到的公式是：产品热销品占比*营销和设计的合作效率*持续性=持续稳定的业绩
...

展开

囧囧冰淇淋
2019-08-25

1.为何国内公司会支持用996工作模式？弊端是什么？
目前国内比较多的三种工作时长：965，966，996。前者大部分是国企，后者则是私营，最后者则是绝少数私营。996暴露出绝大部分公司效率低下，只有通过加长员工的工作时间弥补，但这个弥补绝大部分时候适得其反，员工不理解，磨洋工，最后损失的还是公司。作为员工我们要怎么办？...

展开

chenxqnuaa
2019-08-24

有没哪个公司已经搞了一套能效指标模型？数字化研发管理看样子是趋势，虽然不靠谱。

作者回复: 效能度量比较tricky。我在第二、三篇文章专门讨论效能度量。欢迎听完之后讨论！

牛晋
2019-08-23

最近在参加一些大佬们组织的 Mob 编程活动，感觉这是个对齐团队成员个人效能的好手段，在 Facebook 内部流行么？

作者回复: 我没有看到Facebook有使用Mob编程。我自己也没有参加过。这个看起来的确是一个对其效能的办法（当然了，失望团队中高水平成员对齐😬）

西西弗与卡夫卡
2019-08-23

请教下，Facebook还有哪些研发相关的原则？我觉得这些原则指出了公司研发的基本方向，催生了各种效能工具的诞生。

展开

作者回复: 后面会陆续谈到。这里举3个例子：
1. 重视代码提交的原子性
2. 代码没有严格的ownership。看到别人的代码有可以提高的地方，欢迎去修改。
3. Move Fast and Break Things

yan wang
2019-08-23

葛老师讲得太好啦！不仅在技巧上指明如何效提高效能，终于也在思维上让我们明白了，工作之所以低效，还源于国内老板对员工的习惯性不信任。什么时候员工工作不再认为是给老板打工，什么时候工作效率就会明显提升了？

展开

作者回复: 多谢支持！！

许童童
2019-08-23

老师讲得好，很多干货，读得很过瘾。
这两句重点，一定记在心中。

开发流程的顺畅是生产优质软件的关键因素，只有这样才能最大程度地释放开发者的创造

性和积极性。...

展开

作者回复: 多谢支持！！

研发是灵活的活动，所以一定要从目标和原则出发。理解了原则才好选择具体实践以及应对变化。

Criss Chan

2019-08-23

被工时深深伤害的人听完后感慨颇深！

展开

作者回复: 这个制度对软件研发的确不科学。

杜杰

2019-08-23

期待，后面有更多干货发表

展开

作者回复: 谢谢支持！尽力做好 ：)

1