答疑解惑 | 如何分解一个你不了解的技术任务?

郑晔 2019-02-13

你好，我是郑晔。

在“任务分解”这个模块，我以测试为核心，讲解了任务分解这个原则，同时也给你介绍了一些最佳实践，帮助你更好地理解任务分解的重要性，以及应该怎样分解任务。

同学们对任务分解这个原则大多是表示认同的，但就一些具体应用的场景，还是提出了自己的问题。

在今天的答疑中，我选择了几个非常典型的问题来进行深入讨论。

问题 1：面对不了解的技术，我该如何分解任务？

pyhhou 同学提到

> 很想听听老师的意见，就是在一个自己不熟悉的，充满未知的项目中该怎么更好地进行任务分解？
> ——《11 | 向埃隆·马斯克学习任务分解》

shniu 同学提到

> 想请问一下老师，面对探索型的需求，调研型的需求如何做任务分解呢？
> ——《15 | 一起练习：手把手带你分解任务》

这是一个很好的问题。在这个模块讨论开发中的任务分解时，我说的都是确定了解的某项技术，比如，数据库、REST 服务等等，因为这是开发中最常见的场景，也是最基础的能力，连熟悉的技术都做不好分解，就别说不熟悉的技术了。

那如果不了解这项技术呢？**答案很简单，先把它变成你熟悉的技术。**一旦变成了你熟悉的技术，你就可以应用在这个模块中学到的，面对确定性技术的分解方案。

我知道，这个答案你并不满意。其实，你真正的问题是，怎么把它变成你熟悉的技术。

我的答案是，**做一次技术 Spike。**这里之所以用英文，是因为我没有找到一个特别合适的词来翻译。Spike 这个词的原意是轻轻地刺，有人把它翻译成调研，我觉得是有些重了。

Spike 强调的重点在于快速地试，和调研的意思不太一样。既然是快速地试，就要在一定的时间内完成，比如，五人天，也就是一个人一周的时间，再多就不叫 Spike 了。一些简单的技术，用一天时间做 Spike 就差不多了。

这里强调的重点在于，要做一次技术 Spike。**Spike 的作用就在于消除不确定性，让项目经理知道这里要用到一项全团队没有人懂的技术，需要花时间弄清楚。**

项目经理比你更担心不确定性，你清楚地把问题呈现在他面前，项目经理是可以理解的，他更害怕的是，做到一半你突然告诉他，项目进度要延期。

把事情做在前面，尽早暴露问题，正是我们要在下一个模块要讨论的一个主题。

好，那么接下来的问题变成了：怎么做技术 Spike 呢？

这里，我假设你已经通过各种渠道，无论是新闻网站，还是技术 blog，又或是上级的安排，对要用的技术有了一些感性的认识，至少你已经知道这项技术是干什么的了。

接下来，我们要进入到技术 Spike 的任务分解。

首先，快速地完成教程上的例子。稍微像样点的技术都会有一个教程，跟着教程走一遍，最多也就是半天的时间。之所以要快速地完成教程上的例子，是为了让你有一个直观的认识，这时候，你对这项技术的认识就会超过新闻网站的报道。

其次，我们要确定两件事：**这项技术在项目中应用场景和我们的关注点。**

技术最终是要应用到项目中的，本着“以终为始”的原则，我们就应该奔着结果做，整个的 Spike 都应该围绕着最终的目标做。

很多程序员见到新技术都容易很兴奋，会把所有的文档通读一遍。如果是技术学习，这种做法无可厚非，但我们的目标是做 Spike，快速地试，没有那么多时间，必须一切围绕结果来。

项目中的场景有无数，我们需要选择最重要的一个场景，而针对着这项最重要的场景，我们还要从这项技术无数功能中选取最需要的几个，而不是“满天撒网”。

再有是我们要找准关注点，比如，采用新的缓存中间件是为了提高性能，那关注点就是性能，采用新的消息队列是为了提升吞吐，那关注点就是吞吐。我们选用一项新技术总是有自己的一些假设，但这些假设真的成立吗？这是我们需要验证的。

无论是场景，还是关注点，我们要在前面先想清楚，其目的就是为了防止发散。当时间有限时，我们只能做最重要的事，这也是我在专栏中不断强调的。

确定好场景和关注点，接下来，我们要开发出一个验证我们想法的原型了。这个原型主要目的就是快速地验证我们对这项技术的理解是否能够满足我们的假设。开发一个只有主线能力的原型，对大部分程序员来说并不难，这里就不赘述了。

当你把想法全部验证完毕，这项技术就已经由一项不熟悉的技术变成了熟悉的技术。我们前面的问题也就迎刃而解了。这时候，你就可以决定，对于这项技术，是采纳还是放弃了。

但是，我这里还有一点要提醒，当你确定要使用这项技术时，**请丢弃掉你的原型代码。**

你或许会说，我辛辛苦苦写了几天的代码就这么丢了？是的，因为它是原型，你需要为你的项目重新设计。

如果顺着原型接着做，你可能不会去设计，代码中会存在着大量对这项技术直接依赖的代码，这是值得警惕的，所有第三方技术都是值得隔离的。这是我们会在“自动化”模块讨论的内容。

问题 2：项目时间紧，该怎么办？

在这个模块里，我花了大量的篇幅在讲测试，很多同学虽然认同测试的价值，却提出了开发中普遍存在的一些情况。

玄源 同学提到

> 很多时候，项目时间很紧，经常会提测后，再补测试，或者直接 code review，测试就不写了。
> ——《12 | 测试也是程序员的事吗？》

这是一个非常典型的问题，我在之前做咨询的时候，经常会遇到很多团队说，项目时间紧，所以，他们没有时间做测试。

这里面有一个非常经典误区：**混淆了目标与现状。**目标是应该怎么做，现状是我们正在怎么做。我们都知道现状是什么样的，问题是，你对现状满意吗？如果每个人都对现状是满意的，就不会有人探索更好的做法。

假设现在不忙了，你知道该怎么改进吗？

遗憾的是，很多人根本回答不了这个问题，因为忙是一种借口，一种不去思考改进的借口。

我之所以要开这个专栏，就是为了与大家探讨行业中一些好的做法。

回到这个具体问题上，我们在专栏开始就在讲以终为始，首先要有一个目标，专栏中介绍的各种实践都可以成为你设置目标的参考。有了这个目标再来考虑，如何结合我们工作的现状来谈改进。

接下来，我们以测试为例，讨论一下具体的改进过程。用我们专栏最初讲过的思考框架看一下，假如我们的现状是团队之前没什么自动化测试，而我们的目标是业务代码 100% 测试覆盖。如果要达成这个目标，我们需要做一个任务分解。

这时你会发现，分解的过程主要需要解决两方面的问题，一个是与人的沟通，另一方面是自动化的过程。

与人的沟通，就是要与团队达成共识。关于这点，你可以尝试将专栏里讲到的各种最佳实践以及其背后的逻辑，与团队进行沟通，也可以把专栏文章分享给他们。

再来，我们考虑一下自动化的改进，因为我们的现状是没什么测试，所以，不能强求一步到位，只能逐步改进。下面我给出了一个具体的改进过程：

- 把测试覆盖率检查加入到工程里，得到现有的测试覆盖率。
- 将测试覆盖率加入持续集成，设定当前测试覆盖率为初始值。测试覆盖率不达标，不许提交代码。
- 每周将测试覆盖率调高，比如，5% 或 10%，直到测试覆盖率达到 100%。

这样，我们就找到了一条由现状通往目标的路径，接下来，就是一步一步地具体实施了，由团队成员逐步为已有代码补充测试。

问题 3：多个功能同时开发，怎么办？

妮可 同学提到

> 公司经常存在有两个需求同时开发的情况。请问老师所在的团队如何解决单分支上线不同步的情况呢？
> ——《14 | 大师级程序员的工作秘笈》

在主分支开发模型中，有一些常见的解决多功能并行开发的方法，其中，Feature Toggle 是最常用的一个，也就是通过开关，决定哪个功能是对外可用的。

关于这一点，Y024 同学也补充了一些信息。

> Feature toggle（功能开关）分享两篇文章：
> 1. Feature Toggles (aka Feature Flags)
> 2. 使用功能开关更好地实现持续部署

不过，如果用户故事划分得当，你可以很快完成一个完整的业务需求。实际上，Feature Toggle 只是一个非常临时的存在。但如果你在一个遗留系统上工作，一个功能要跨越很长的周期，Feature Toggle 才显得很有用。

额外补充一个与主分支开发模型相关的常用技术，如果你想对遗留系统做改造，传统的做法是，拉出一个分支。

如果在一个分支上怎么做呢？可以考虑采用 Branch by Abstraction，简言之，再动手改造之前，先提取出来一个抽象，把原先的实现变成这个抽象的一个实现，然后，改造的过程就是提供这个抽象的一个新实现。这种做法对设计能力有一定要求，所以，对很多团队来说，这是一个挑战。

好，今天的答疑就到这里，请你回想一下，你在工作中是否也遇到过类似的问题呢？你又是怎么解决的呢？欢迎大家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。

Geek_73143a

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(11)

Kǎfkā²⁰²⁰

“技术Spike”可以翻译成“技术撩”，就是撩妹的那个撩。试探下，有戏就继续，撩不动就算或者放一段时间再说😃

作者回复: 神来之笔!

2019-02-13 67

zhengfc

老师您好，spring-boot 项目有什么简单易用的测试覆盖率检查工具呢?

作者回复: jacoco，我现在常用的工具。

2019-02-27 4

Xunqf

最近我们团队也用到了不了解的技术做项目，一开始先用不到一周的时间去了解，写demo.又花了一周多写了一个简单一点的新项目，并且成功上线了。但是这个时候我只能说只是会用这个新技术，虽然使用起来很熟练，但是对他的底层原理还是不太了解。这个时候公司有一个老项目，功能比较复杂一些，因为这个老项目历史遗留问题比较多，崩溃率也比较高，这时候老大发话了，大家要不试着用新的技术重构一遍吧！因为新技术还不够成熟，各种功能缺失，或是尚未提供，一些常用的组件都要自己一个个去实现，因为项目涉及到的功能比较多，也不太可能一个一个的去验证，这个时候也只有在做的时候才会暴露问题，所以一再延期。一开始评估的时候也知道新技术对这些功能都提供了支持，但是不清楚能支持到哪一步，到具体去实现的时候才发现很多坑，目前已经陷入这个项目三四个月了，感觉离完成瑶瑶无期啊，不知道老师有什么好的建议没。

作者回复: “怎么把新技术用在自己的项目中”，在这个问题中，很多人有一个严重的误区，他们眼中看到的更多的是“新技术”，而我思考这个的逻辑在于“自己的项目”。

只有理解清楚了自己的问题，才好应用新技术去解决，盲目地采纳新技术，只会让自己不断地纠结，小程序库还好，要是引入一个大框架，无穷的问题就会吞噬你的时间。

2019-02-20 4

🌲树根🌲

spike 我翻译成刺探😁，从medium一篇文章也了解到这个来自于极限编程。里面也有一些建议觉得蛮有用的。翻译整理了一下 http://t.cn/EIjW8su

> 作者回复: 很好的补充!

2019-03-06 3

行与修

对于不了解的技术任务，我会采用写伪代码的方式展开，先肢解成块，后续逐个突破。也就是先完成粗粒度的任务分解，看看哪些是现有的知识储备可以搞定的，把拦路虎限定在小范围内，战略上藐视它，树立解决问题的信心，然后是二次任务分解，着手预研和定型。

> 作者回复: 我不确定写伪代码的效率是否高，你如果很擅长，可以坚持。

2019-02-14 2

陈斯佳

学习任何新知识，最好都先有个框架性的认识，然后抱着最终目的去分析，都能很快入门。这比起系统性学习的想法更加高效有用

2019-05-23 1

helloworld

遇到紧急性的需求并且用到的技术自己不熟悉，首先要对这个技术有一个大致的了解，其最主要的功能是什么，再就是结合需求，看看如何利用这个技术解决这个需求。当需求做完后，并且市场对这个技术有很大的需求的话，可以对这项技术的细节加以学习研究。

> 作者回复: 只要不跑偏，效率都不低。

2019-03-06 1

亥时

对于要使用的新技术 以始为终的角度来看 个人的理解：
1. 了解要做的事： 新技术是否满足功能要求（目前和未来）
2. 业内成功案例背书（大厂、成功案例）
3. 分布式能力是否满足（性能、可伸缩、高可用...）
然后快速写个demo、进而结合实际项目写出功能 这也是学习新技术最快的方式，带着目的去用 而不会太发散 导致没个重点

然后就是结合官方文档，了解原理，猜测底层实现、然后去源码中验证 已经进行性能压测等

> 作者回复: 你说的是一个技术选型的过程。首先，要确定自己要做的问题，再来根据自己的问题确定合适的技术。

2020-05-31

时代先锋

问题分解是不错的方式

2020-03-26

丁丁历险记

1 技术撩。
2 明确关注点，快速对假设进行验证。
30防止发散，防止发散。
3 记得丢失原型代码，隔离第三方。

目标与现状的思考，不忙了如何改进。忙是不思考改进的接口。

这里想到某经济学家的一句话，有时后，勤劳是思考上的懒惰，去躲避那些需要深度思考，反复权衡理解分析调整需要用脑的事

2019-11-11

Twinkle

spike 可以理解为技术探针

> 作者回复: 这是一个正常的翻译 :)

2019-07-10