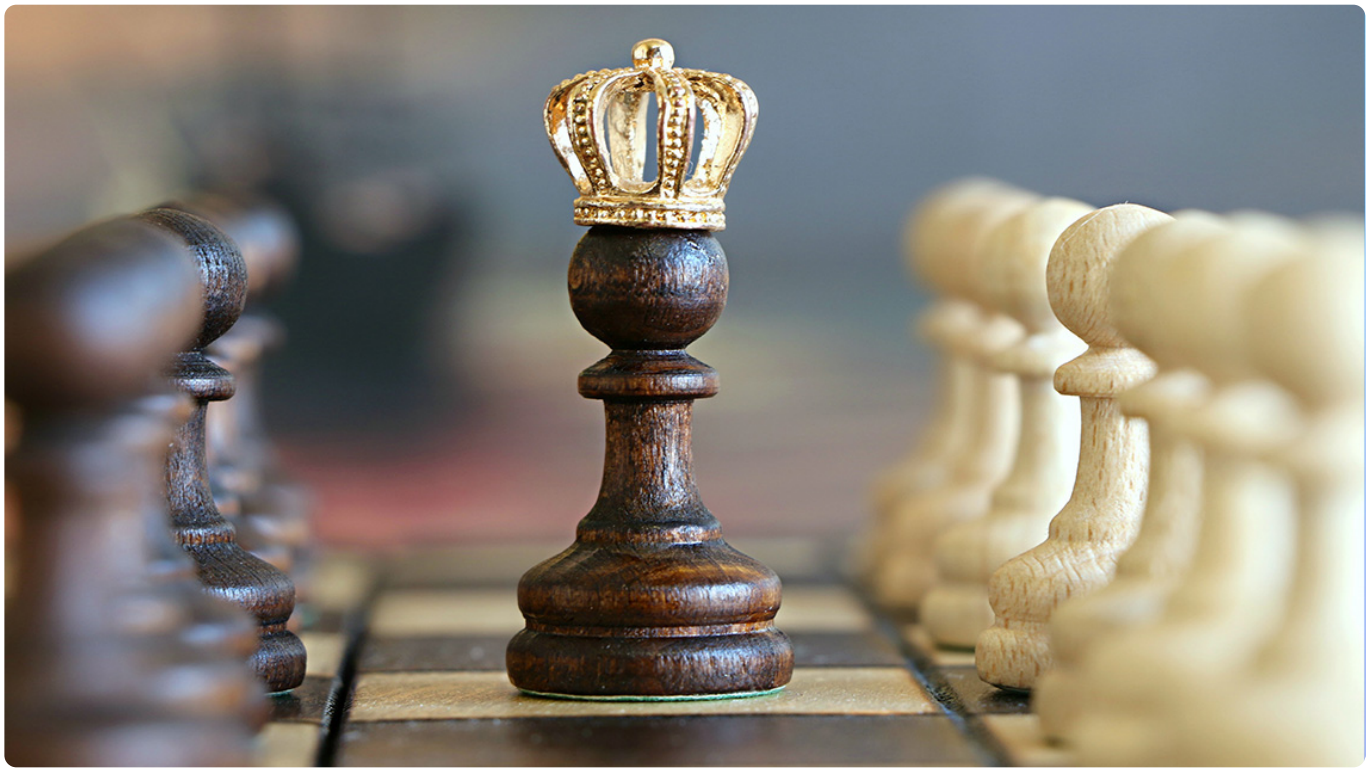
28 | 软件工程师的核心竞争力是什么？（下）

2019-05-02 宝玉



讲述：宝玉
时长 15:47 大小 14.46M

你好，我是宝玉。在上一篇中，我们讨论了什么是软件工程师的核心竞争力，也就是学习能力、解决问题的能力和影响力。

今天我就来跟你谈一谈，如何提升软件工程师的核心竞争力，也就是说，如何分别提升你的学习能力、解决问题能力和影响力。

如何提升学习能力？

学习能力是软件工程师最基础的能力，学习能力直接决定了你掌握技术的速度。现实中，有的程序员掌握新技术就是比别人要快，似乎有什么秘籍，可以让他们快速学习和掌握新技术。真有这样的秘籍吗？如果有，那是什么样的内容呢？

前不久我要写一个家谱的微信小程序，要实现列表、检索等功能。此前我没有任何关于小程序的开发经验，于是我粗略地看了下开发文档，搭建了开发环境，下载了几个示例程序，前后花了一天时间完成了一个不算简单的小程序开发。

为什么我能在一天时间就可以学习掌握小程序的开发呢？其实只是因为我已经构建了自己的开发知识体系，而小程序开发所需要的知识，绝大部分在我的知识体系里面已经有了存储。这些部分我就不需要重新再学习，我只要去学习小程序所独有的知识就好了，而这部分知识只有很小比例，所以很快就可以掌握。

那么你可能会问，怎样可以构建自己的知识体系呢？

首先需要在一个技术领域深耕

每个人精力其实很有限的，一开始专注在一个技术领域容易在短时间取得成绩，同时也相当于建立起了最初的知识体系，在未来的知识森林里种下的第一棵大树，这样当你开始学习新的技术的时候，已有的知识就可以直接借用，相当于这棵大树可以帮助新的知识树的成长提供很好的养分，快速培养出新的大树。

如果一开始就同时涉猎多个领域，每个领域的知识又没有掌握好，这样的知识是没法共享的，就相当于你种的只是一片知识的灌木，最终只能收获像灌木丛的知识体系。

只有一个领域的知识你真正吃透，才能有效地共享到其他领域，构成一个知识领域的森林。

我在毕业之后，整整有 6 年时间，是一直专注于 Asp.Net 技术领域，让我能成为这个领域的专家，也帮我构建了最基础的知识体系。所以在后来我去学 iOS 开发时，发现像面向对象、网络协议、数据存储、MVC 开发模式这些知识，我就不需要再去学习，因为我的知识体系已经有了这部分知识。

换个角度说，如果我当时 Asp.Net 学的不够深入，那么面向对象、数据存储、MVC 开发模式这些我还是要再学一遍，自然也无法快速掌握 iOS 开发。

知识体系建立之初，确实是痛苦的，感觉什么都不懂，有很多要学习的。市面上有新技术出来也会觉得焦虑，觉得应该去学新的技术。但如果初期不能够专注在某个领域深入的话，你学了再多技术，结果也没有一门能深入，这就很难构建出有深度的知识树。

要在某一个领域的技术达到一定深度，通常需要三年以上的时间。当你熬过这个阶段，在一个技术领域取得了一定成就，不仅会收获你的知识树，还能收获技术上的自信，让你有信心在其他技术领域也同样取得成就。

然后往相近的领域逐步横向拓展

当在一个技术领域达到一定深度后，可以开始横向扩展。最好是往相近的领域扩展，因为这样你之前的知识有很多是可以共享的，容易快速取得成绩。

这也是为什么我后来选择了前端，因为对我来说，前端跟我以前掌握的 Asp.Net 的经验是有很多重叠的，我只需要去学习框架和工具那部分知识。

当然横向构建知识体系，也一样不是一个轻松的过程，因为以前你在某个领域取得的成就和经验，反过来也会成为一种阻力。因为以前你熟悉的知识，已经变成了你的舒适区，你会天然地不愿意走出舒适区，不愿意到挑战区或恐慌区去学习新的知识。

这其实就是我当初学 React 时候的感受，我觉得 jQuery 已经用的很好了，为什么要学这个？我老想着用以前 MVC 的经验去套用 React 的编程模式，反而更难理解。

但是，当你迈过去，掌握了新领域的知识，就会感觉整个知识体系一下子扩展了很大一块。相当于让你的知识体系，从一棵树，逐步变成了一个小树林，最终会成为一个森林。

快速掌握新技术的秘籍，就是要构建属于你的知识体系，让你在学习新知识时，能借用已有的知识，加快学习速度。

你可以现在思考一下你的知识体系是什么样子的，是一片灌木丛还是已经有大树了？如果只是一片灌木丛，你打算在哪个技术领域先打造你的知识树呢？

如何提高解决问题的能力？

解决问题的能力是软件工程师进阶的核心竞争力，你看现实中，那些解决问题能力强的程序员，遇到问题总是有办法，都能有条不紊地给解决了。

有一次我们组负责的一个网络服务出现异常，大约 1% 的请求会出现异常导致服务报警，当时值班的同事正好是一个新手程序员，他也试着分析日志，但是没有找到明显的错误日

志，完全不知道如何应对，只好让我帮忙。这个问题我以前也从来没遇到过，但我有一套应对这类问题的思维方式和逻辑。

我第一步就是回滚上一次的部署，看是否恢复正常，结果没效果。于是我再从日志找出所有有问题的请求，寻找规律，从中选取几个有问题的请求，去跟踪它们整个请求过程，结果在某一个路由环节发现指向到了一台错误的服务器，最终定位是某个路由程序出现的问题。

我们将这个问题反馈到相关组，问题马上就得以解决。然后我建议他在这个路由环节增加监控，这样以后再出问题就能第一时间报警通知。

为什么我有这样解决问题的能力呢？是因为我在多年的开发经验中，形成了一套解决问题的方法论，即使遇到没有解决过的问题，也能借助一套方法论去解决。

这其实很像我在专栏开头就提到的工程思维，遇到一个项目，哪怕不是软件项目，我也可以借鉴瀑布模型，用工程方法去解决。所以你要提高解决问题的能力，就要形成自己的一套解决问题的方法论。

在这里分享我解决问题的一套方法论，其实很简单，只有三步。

第一步：明确问题

解决问题，最重要的一步就是要明确问题是什么，这其实就跟做项目需要先需求分析一样，搞清楚目标是什么，才能做到有的放矢。

同时这一步也要透过现象看本质，去明确问题背后是不是还有其他问题。就像我在前一篇文章中举例的抽奖项目，不能光看到功能需求，还需要看到安全上的需求；网络异常的问题，不能光想着应用程序错误，还要看看网络是不是有问题。

第二步：拆分和定位问题

前面我们学习架构思维的时候，也提到了，一个复杂的问题，只有经过拆分，才好找到本质的问题。

就像上面举的解决网络异常的例子，我首先拆分成程序问题和网络问题，通过回滚观察，我就可以基本上断定不是程序问题，那就说明是网络问题。然后对于网络问题，将整个请求过

程拆分，最终就可以定位到有问题的环节。

第三步：提出解决方案并总结

发现并分析完问题后，找到解决方案是容易的，但很有必要总结一下。总结要做的就是两点：

下次有这种问题怎么解决，是不是可以做的更好？

这种问题是不是可以预防？如果可以，应该怎么做？

通过总结，就可以进一步提升解决问题的经验。如果你对于解决技术上的问题还没有总结出来自己的方法论，不妨可以先参考这套方法，一步步去发现问题，分析问题和解决问题。

尤其是在解决完问题后，再想一想如何预防以后类似问题的发生。**如果每次解决完问题，你还能提出一个预防问题发生的方案，一定会让大家印象深刻的。**

如何提升影响力？

刻意地去提升自己的影响力，是很多软件工程师忽略去做的事情。但影响力，却是程序员核心竞争力的最顶层，也是一个软件工程师能力的综合体现。培养影响力也是我职业生涯中受益良多的一件事。

其实当你意识到影响力是有价值有意义的事，怎么去做反而没那么难，比如这些方面可以去考虑。

在某个领域做出了足够牛的成绩

有些程序员能在某一个技术领域做到一定深度，做出了常人难以达到的成绩，比如说 PHP 开发组核心成员的鸟哥惠新宸，写 Vue 框架的尤雨溪，前端的 Winter。做到他们这样，基本上就不用担心影响力的问题了。

要取得这样的成绩，要实力、要机缘、还要坚持。

做事情超出预期

在软件项目中，你作为一个程序员，每个人都会对你有预期，项目经理希望你如期完成项目，产品经理希望你完成需求，其他程序员希望你代码质量好。如果你是初级程序员，则大家期望你代码不要有太多问题就好，如果你是高级程序员，大家不仅期望你要写好程序，还要能带带新人。

如果你做事情的结果能超出预期，就会让人对你刮目相看，进而会形成口碑。就像我前面举的例子，如果你解决完一个问题，还能想到怎么预防问题再次发生的方案，这通常就超出他人对你的预期了。

要让自己的表现超出其他人的预期，除了要付出很多努力外，也要学会管理好其他人的预期，我以前有写过一篇文章《程序员也可以懂一点期望值管理》，谈如何去了解别人对你的期望，降低别人对你的期望，最后做出高于期望的事。

帮助其他人就是在帮助自己

我有遇到过很多程序员不愿意教别人，认为教会徒弟饿死师傅。其实这完全搞反了，帮助别人、教别人收获最大的恰恰是自己。

程序员的经验，很大部分来自于解决问题时积累的经验。你自己在工作中遇到的问题其实是很有限的，但如果帮助其他人解决问题，相当于增加了你解决问题的样本，这些样本能帮助放大你的工作经验。

帮助其他人，还是形成影响力最简单有效的途径。就像我微软最有价值的称号，就是因为我当年在社区热心组织活动，网上帮助其他人解决技术问题而获得的。

分享就是学习和打造影响力

刘未鹏老师写过的一篇博客《为什么你应该（从现在开始就）写博客》影响了很多人，这是一篇值得反复阅读的好文章，讲了写博客的好处，例如交朋友、帮助思考、学习。

我的观点也类似，包括在学习路径中我就建议大家可以通过分享，在教中学。其实，在分享形式上也可以更多样化，除了写博客还有很多其他方式，比如公司内部的讲座就是很好的分享途径。

我在内部也会经常做培训分享，也会写博客分享技术经验，每一次分享都会帮助我学习巩固很多知识点，也是在帮我打造自己的影响力。

写东西这方面我做的不够好，写博客断断续续的，但还是在坚持，也收获很多，包括《软件工程之美》专栏的诞生，其实也是因为极客时间看到我写过的一篇《记在美国的一次校园招聘》，所以才邀请我开的专栏。

希望你也能思考一下影响力这个问题，想想你还可以在哪些方面，用哪些方法去打造你的影响力，和学习能力、解决问题能力一起，形成你的核心竞争力。

总结

最后简单总结一下。软件工程师的核心竞争力，体现在学习能力、解决问题能力和影响力三个方面。

要提升学习能力，要构建好自己的知识体系，首先需要在一个技术领域深耕然后往相近的领域逐步横向拓展。

要提升解决问题的能力，要形成自己的方法论，去发现问题，分析问题和解决问题。

要提升自己的影响力，可以在一个领域深入打造自己独特的有价值的能力，让自己做事情能超出别人的预期，同时乐于分享和帮助他人。

课后思考

你的知识体系是什么样子的，你是怎么打造你的知识体系的？你自己或者你周围哪些擅长解决问题的软件工程师都有哪些独特的方法？你是怎么提升影响力的？欢迎在留言区与我分享讨论。

感谢阅读，如果你觉得这篇文章对你有一些启发，也欢迎把它分享给你的朋友。

上一篇　27 | 软件工程师的核心竞争力是什么？（上）

下一篇　29 | 自动化测试：如何把Bug杀死在摇篮里？

精选留言 (7)

写留言

kirogiyi
2019-05-02

5

提升技能方面的核心竞争力，除了提升思维能力，还要加强基础技能的知识储备，不能去学一些表面能解决问题的知识散点。

拿组件库、框架来说，在面试的时候，一旦问到组件、框架相关的计算机基础知识，怎么实现的，原理是什么，用了哪些设计模式，经常得到些似是而非的答案，更有愠怒的面...

展开

作者回复: 感谢分享，节日愉快

hua168
2019-05-02

4

基础不牢，地动山摇
我发现很多人，基础都没学好，更别谈什么深耕了……
比如基础的基础是自学能力，很多人都不具备自学的能力，所以他们连自学都没学好。
比如很多人想学习运维问我要资料，我就想让他学网络开始，给他一本CCNA学习指南，过几天问我，问他有什么疑问吗？他说没有疑问，根据我的经验，那肯定没学好。我问…
展开

作者回复: 很好的总结。

书看一遍肯定记不住，得经常翻才行，看完得思考，思考完了还得实践，实践完了还得总结思考，最后才能变成自己的知识。

果然如此
2019-05-07

1

1.技术方面
后端.net、扩展到java，前端jquery扩展到vue、小程序等，大数据solr扩展到es、mongodb、redis、rabbitMQ等；
2.产品
3.沟通…
展开

作者回复: 感谢分享

Gao
2019-05-05

1

老师所讲排查生产问题的案例，首先回滚版本，再看日志。这会引发更多的系统功能不可用吧，两个版本之间的功能差异尚不清楚就直接回滚，系统风险是否被进一步扩大?
展开

作者回复: 这个确实要具体情况具体看，因为我日常的系统上线，都会有回滚方案，回滚也是自动化的很方便。有些跟数据库相关的如果数据库结构发生变化又产生了新数据，确实没法直接回滚。

这是我没讲清楚，这里可以作为一个参考即可。

hua168
2019-05-02

1

所说的T型人才，一专多能，先一专后多能?
IT运维和开发基本上都需要学网络、linux、编程、数据库、一些安全知识吧，之前侧重点不同...
开发注重编程和数据库
因为注重服务器管理、编程实现自动化，开发运维工具，平台
展开

作者回复: 这篇主要还是讲的构建知识体系，只是构建知识体系也要先专注才行。

纯洁的憎恶
2019-05-10

提高学习能力，用知识体系对抗复杂多变的问题。提高解决问题能力，明确问题，拆分与定位问题，提出解决方案并总结。塑造影响力，深耕业务能力成为专家，做事超出预期，尽可能多的输出价值，分享是施加影响力的有效途径。

我很早就知道知识体系的重要性，我也比较重视构建知识体系，但并没有什么亲测有效...
展开

作者回复: 方法不是最主要的，最多让你学习提升一点速度。关键还是坚持，多练习多实践。

从知识转变成技能，一定需要通过反复的刻意的练习，才能形成条件反射，最终掌握。没有任何学习方法能替代练习，最多有像催化剂，可以加速练习效果的学习方法。

还有就是对技术的学习，不能太依赖于工作上的输入，工作上如果项目好用户多，那还是很有挑战的，但大多数时候没有那么多挑战，可能就是个增删改查，那么几年的工作经验可能只是简单的重复，不能达到刻意练习的效果。那还是要在工作之外寻找一些练习的途径，比如上次我建议的：自己做一点项目、参与一些开源项目。

要想对知识体系有体感认识，还是建议先在一个领域有深度，有一棵树了才能想像出来森林是什么样子的，不然只能看到一片灌木丛。这过程难免要踩很多的坑，经历很多次的失败和挫折，反复的思考、总结和重试。

Gao

2019-05-05

感谢老师补充

展开