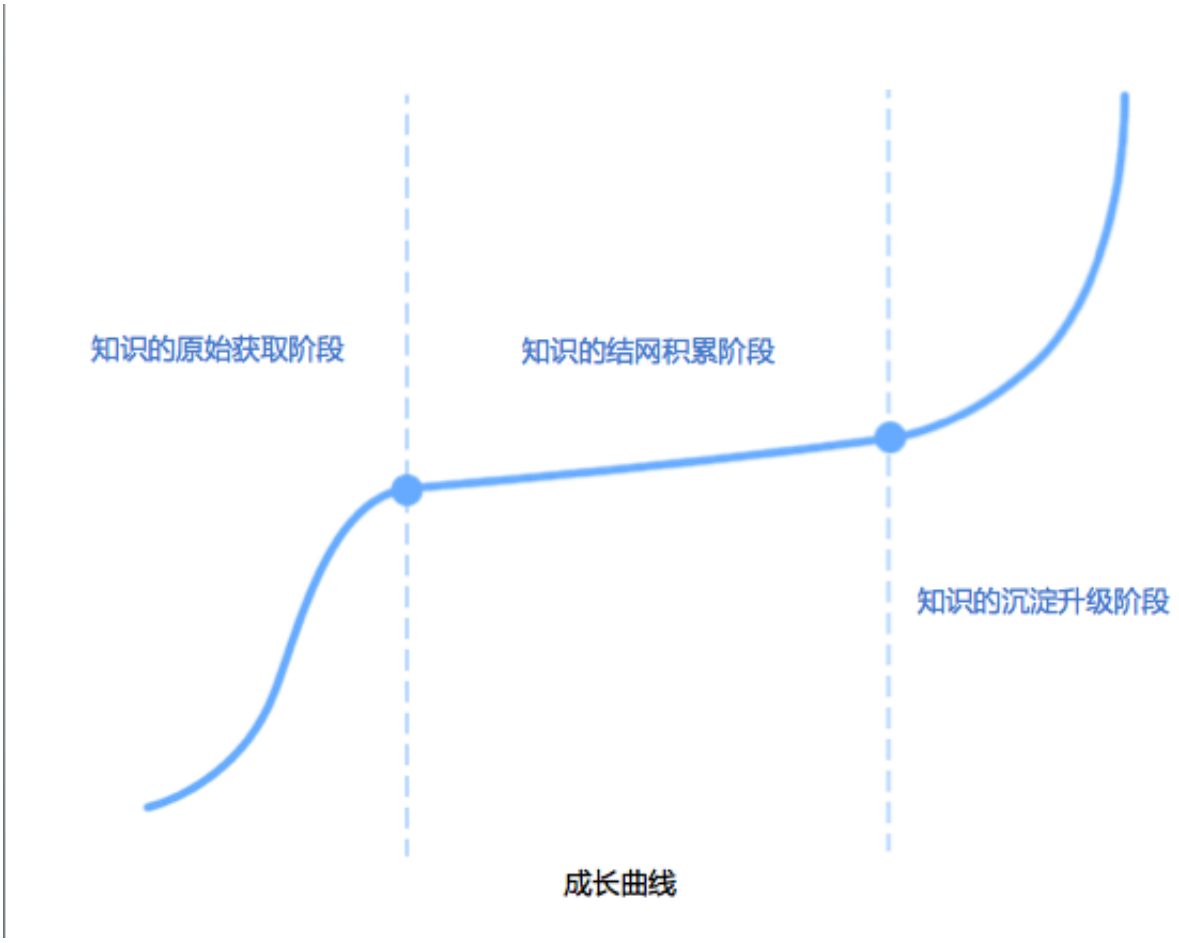
谈谈技术人员的成长

大房 · 8 个月前

> 本文是我在武汉分公司做的一次培训演讲，整理出来希望对大家有所帮助。最先发表于我的公众号：FangTalk

最近两三年，技术之外，花了一些时间带技术团队，也陆陆续续和大家聊了很多关于技术、管理以及成长方面的话题。聊天总归比较零散，今天以文字的形式，把我的所思、所想做一下系统的整理和总结，也许，对正在徘徊的你有所帮助呢？

我眼中的技(手)术(艺)人成长曲线



如果起点是进入一个行业的小白，那么到成为这个行业的专家，至少经历三个阶段：

1. **- 知识的原始获取阶段**：刚进去一个行业，我们都是“学徒”，不懂的东西很多，每天只要稍微动动手、动动脑就可以感觉到不错的成长。毕业工作后开始融入一个团队，开始参与项目的开发，有师傅带、有同事学，很多时候你不想学到东西都很难。所以，在这个阶段，是自身技术的爆棚期，是一个快速成长的阶段。

2. **- 知识的结网积累阶段**：八年抗战，经历过第一个阶段以后，大家都有了自己的看家本领，技术足以应对日常的工作和研究。每天忙忙碌碌，但是感觉都是重复性的工作，“收获很小”。即使继续学习，发现每天研究和接触到的新知识很多都是重复性的内容。渴望伴随着迷惘可能是这个阶段的特点，从“灵魂深入”隐约感觉到自己应该再多学点东西，但是每次付诸行动感觉都收获颇少，所以开始怀疑和迷惘。其实，大家忽略了这个阶段最本质的特点：由于知识的广度加快，知识的深度速度“变慢”，但是广度的知识，往往给人一个“肤浅”、“无用”的幻觉。“广”在某个程度上就是“深”，看似无关的经历经验、看似无关的知识点，其实，在经历一个长期的获取、思考、吸收之后，突然有一天，点成线成网，人有了一种“大彻大悟”的感觉，迅速进入第三个阶段。

3. **- 知识的沉淀升级阶段**：大彻大悟之后的升华，开始一段新的快速成长的阶段。

曲线只有最低点，没有最高点，所谓“心有多大、世界就有多大”。做为一名技术人员，时刻清

楚自己当前的位置。另外，我觉得这个曲线除了对技术人员适用，应该同样适用于其他行业，共勉。

我会在每个阶段，根据我切身的经验，聊一下我的想法。

阶段一：知识的原始获取

这个阶段，就是让自己拥有一技之长，可安身立命。简单点说，就是“我有别人认可、不可被替代的价值”！

做为技术人员，当然首先体现在这几个方面：

- **- 至少熟悉一门编程语言。**不管这个语言是啥，JUST DO IT，把它搞熟，至少做到写代码得心应手，基本语法、高级技巧了然于胸！很多人总喜欢对(纠)比(结)，A语言好还是B语言好，这就像，你看到有人开奔驰，有人开宝马，所以你就开始纠结以后我到底是往奔驰的方向努力，还是往宝马的方向努力？买奔驰还是买宝马，纠结这个，倒不如去纠结：今天我是开奔驰，还是开宝马！

- **- 至少熟悉一个数据库。**不管是MySQL、SQLite还是PostgreSQL，还是其他的RDB。熟悉不仅仅是可以写出“别人看不懂的SQL语句”，更多体现在你对这个数据库的了解和驾驭程度，例如性能调优、索引优化、库表设计、数据库设计的原理以及最佳方案。

- **- 熟悉使用各种项目协作工具。**这是团队合作的基础，例如沟通IM工具、代码管理工具、任务追踪工具、Wiki/Markdown文档编辑工具，等等等等。

- 重要的是，不管是前端、服务端还是客户端，**让自己成为某一个领域的能手**！

如果做到了这一步，只能说明你仅仅胜任一个优秀的软件开发工程师。我觉得还远远不够，最多只能说，这个阶段完成了一半。另外一半更重要的就是自己的修行：

- **- 自己知识结构的横向发展。**如果你只熟悉一门语言，或者你用“熟悉一门语言的眼光”去看“另一门仅仅了解的语言”，就鼓吹说“PHP是世界上最好的语言”，只能说你是无知的井底之蛙。我们需要带着一种对比的眼光，去了解多种语言，知道每种语言的优缺点以及适用场景，做到能够根据产品特点、交付周期、团队特点、性能等多方面考量下的语言和框架选型。同时，关系型数据库之外，开始接触和了解热门的Redis、MongoDB等NoSQL数据库，了解不同的数据库的优缺点和使用场景。另外，读优秀的代码！读优秀的代码！读优秀的代码！至少你得读过两个比较热门的MVC的框架源码吧？“好奇”驱动，我和我的团队常常说的一句话就是，代码里所有你觉得比较“神奇”的地方，都值得你通过阅读源码把神奇

的魔法挖出来。例如，一个请求的URL如何映射到你的action上？映射到你action上的context是如何被创建出来的？有一些工具方法不经过初始化、不需要引用你就能直接用？当你靠“脑补”不能补上这块空白的时候，你需要挖进去，直到下次看到类似的神奇你脑补就能想明白。

- **- 个人能力的全面发展。**开始扛起项目组里的攻关大旗，着手解决一些别人棘手和不能解决的问题，“解决难题”才能拉开距离，这是加分题！（把项目做出来，其实是一件非常简单的事情，持续的维护和运维，才是一件非常困难的事情，个人的能力的深挖，也体现在这里）。另外，全面发展还体现在不断提高自己的沟通能力（沟通的闭环原则），可以参见我另一篇文章《和开发组长聊聊沟通》，可以带团队、以项目负责人的角色制定项目计划、执行项目计划并完成项目或者产品的交付。

这个阶段，持续1年还是5年还是更长，在我看来，决定因素不是智商，而是你到底为了获得这样的一技之长付出多少时间和汗水。如果，IT是你终身的职业，那么，榨干你智商之外的潜力，每个人需要付出的努力是一样的，你唯一需要决定的，是在刚毕业、精力旺盛的头几年就把自己的潜力挖掘出来，还是拉长战线用个十年八年？战线拉的越长，投入产出比一定越低，辛苦程度一定越来越高。如果到成家之前，你都达不到第一个点，那以后工作带给你的只有痛苦，任何新的技术的出现，对你带来的都只是痛苦。

这套理论或者说想法，我刚毕业的时候也不知道，也没人告诉我要多努力多努力。只是我觉得我比别人的起点低，同时也算比较幸运，在我最有精力和时间的时期，在工作、技术成长上付出了几乎全部的努力，来拉短这个差距：

- - 读各种书籍：不管是在项目中用到哪种语言，几乎看完了相关语言的经典书目
- - 看各种代码：项目里相关不相关的代码，通读；用到的开源代码，第一件事就是读完源码再动手使用

- - 写各种代码：尝试的改进通读的代码，工作之外也练手了好多代码

阶段二：知识的结网积累

这个阶段，更像是人到中年，不能靠拼体力、拼“编程代码量”、拼时间来提高自己，而是漫漫长路上，练就内功大法。在我看来，主要包括两个方面的修炼：

- **- 抓住机会拓宽自己的视角（扩大知识面）。**机会可能是你接下来几年参与或者主导的项目以及积累的项目经验，也和你工作之外自己的努力有关。参与不同的项目、接触不同的客户，对一个人的影响和改变，“经验”只是其中一个很小的方面，更大的影响则是你看问题的

视角。不同的项目了解不同的行业，把工作、项目、产品做到自己能力的至高点，成为某个行业的专家（达到力所能及的高度）；接触不同的客户，和不同风格的客户合作、沟通交流，做到从容面对和善的、虚伪的、蛮不讲理等等各色客户，学会站在客户的角度思考问题、讲客户听得懂的语言、上下管理好客户的预期；开始意识到团队的重要性（你真的懂团队吗？），能够架构团队结构、成员的成长，带出产品或者项目的同时，能带出一个能打硬仗的队伍。知识的广度延伸，是一个缓慢的过程，工作中不懂技术万万不能，但是只懂技术也万万不能，而技术之外的领域，对技术人员来说，学好学精，也是最困难的地方。

- **- 个人的持续学习（持续深挖）。**冰冻三尺，非一日之寒。这个阶段，对大部分人来说，是最难的一个阶段，难在很多人会放弃（反正我可以应付日常的工作，满足带来的止步），很容易迷惘和浮躁（我还想学，但是每天看看看的，都是自己会的，那些不知道的对我的工作也没啥用处）。而正确的改变自己的认知（抓住机遇、正确的对待自己的努力），是突破这个阶段的前提。知识的积累，一开始就像在储备知识这个网状结构的节点。每一个知识碎片，都可以是这张网上的一个节点；而且很多节点在一开始的时候，你自己也搞不清楚到底有用还是没用，更搞不清楚节点之间的联系。就像如果你的脑子里只有两个点，没关就是没关，你也连不上，但是当我们把这个结构无限的放大，节点多到一定程度，很多知识相互连接，就是一件很自然的事情。

- **- 社交以及朋(战)友。**找到你的战友，在你孤独的时候、迷惘的时候，有陪你喝酒吹水的人。也所谓三人行必有我师，很感谢这么多年，我身边的这些朋友，亦师亦友，让我学到的远远大于我自己的能力所获取之外。

所以，我个人的经验：

- - 如果你有非常明确的努力方向，那么脚踏实地，坚持不懈。有进步，那是正常的，如果你感觉不到进度，那也是正常的。切勿把自己放到一个结果导向的思维定式里。

- - 如果你没有明确的目标和努力方向，但是你要认识到一点，至少你有看家本领，你要做的就是把自己该做的事情做好就够了，目标只是你不知道，其实“天知道”^_^.

We are always putting ourself into the perceived world, break it!

有时候我也把这种方法，叫做“自我麻痹”。当你发现自己难有突破的时候，你要么找别人“麻痹”你自己，要么你找到一种合理的观念自己“麻痹”自己。很多时候，没人告诉我到底该怎么来做，我需要自己给自己一点信念，不断的用那个还有点激情、还有点梦想的我去打败那个满足、懒散、浮躁的我。

仔细想想，你是不是也有两个我？当你用一个我打败另一个我的时候，要么你进入的第三个阶段，要么你退回到第一个阶段。

阶段三：知识的沉淀升级

胜利即升华。

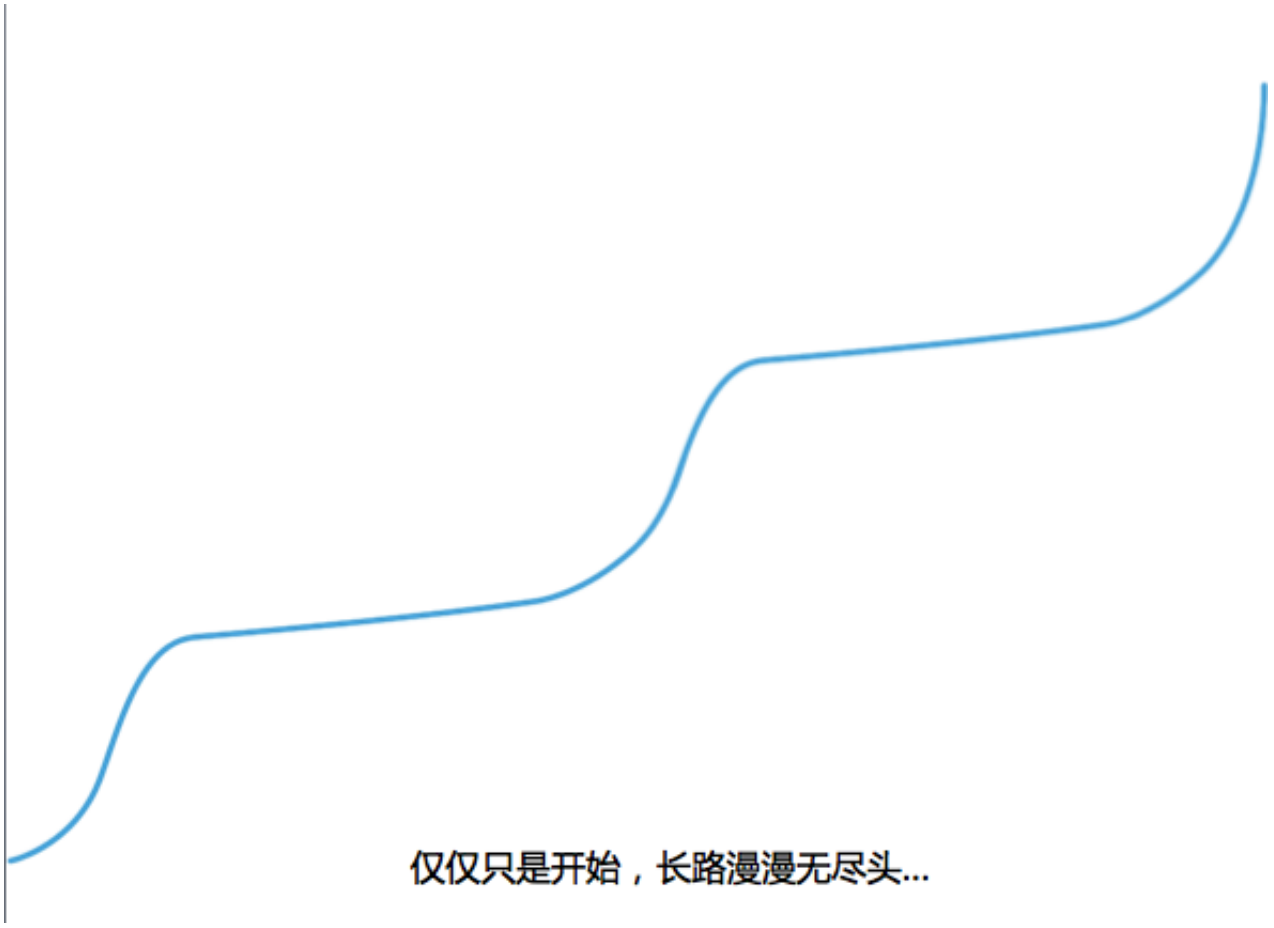
这个阶段，是蜕变成“大神”的开始，是**用知识创造知识**的量变到质变的过程：

- **- 构造模型，用“已知”加速掌握“未知”**。我们经常谈，谁谁谁学习能力比较强。什么叫“学习能力”？就是谁能快速的用自己的旧知识来掌握新知识、未知的知识！怎么做到？思考（对重复性业务的抽象和未来业务拓展的前瞻，强调过去的经验和对行业的预见）-》找到规律规则，也即模型-》应用模型规则-》改善。杂乱无章的东西，不符合人的认知，只有从杂乱无章的事物里找到规律，才能被人的认知体系所接受。找规律的过程，就是建模的过程。几个例子。技术人员经常讲(喷) “这个语法是抄的哪门语言的”。什么意思？因为你的认知体系里已经有一个语法的模型，所以你在用你已有的语法模型去认识新的语言里的语法模型。我们要学一门新的编程语言，学什么东西？“哎呀，我在熟悉的那门语言里是这么写控制语句的，这门新语言是这么写的” “哎呀，我在熟悉的那门语言里是怎么定义类和函数的，这门新语言是这么定义的”。等你语言见多了，你就会有自己对语言的认知模型，然后你就会骂人：“这些创造语言的大牛都该去死；有毛病，用 function 用的好好的，你非要改成 func，或者改成def；我用花括号好好的，你非要改成冒号；我声明变量的时候，把变量类型放到前面，你非要改成写到后面。”... 如果你这么想，而不是“PHP是世界上最好的语言”，恭喜你，你已经开始有了自己认知的模型，然后你会抽象出这样一个模型：编程语言其实是按照编程语言创始人的理(喜)论(好)定义好让你写字的规则而已；所以学习新的语言，就是了解一下别人定义的规则，没有好坏对错，我们都是牛人的“跟屁虫”而已。等你代码写更多之后，你会这样来认知新语言：了解一下语法规则；看一下框架；熟悉一下生态（libs是不是足够过，应用场景都有啥，你熟悉的工具在这个生态下有没有类似的替代）。所有这些放在一起，才是你做技术选型的依据。再举一个例子，开发框架，给你一个新的开发框架，你需要了解啥？这个“需要了解啥”的定义，就是你认识编程框架的模型：路由机制是啥样的；模板机制是啥样的；插件扩展性体现在下；工具集有哪些；性能考量和开发效率考量有哪些。你可以按照同样的方式和方法去认识TCP/IP协议：特定场景(Context)下对0和1的解释。有了这个认知模型，你学习TCP/IP其实就是在学：有多少种context，在每种context下每个位上0和1代表了什么意思。这同时也是性能优化的关键。再比如，按照这个思路应用到编程之外，和客户合作的协议是什么？是基于信任与关系模型下的win-win。
- **- 微创新，对已知的归纳总结、打磨升级**。当你开始寻找规律、构建模型之后，接下来要锻炼的就是，用这个规律和模型去不断的认识新的知识，然后在认识了新的知识之后，返回

去改进或者完善你的认知模型。

- - 最后，你会**掌握了用自己的视角认知这个世界的方法**。别人告诉你的世界都是假的，你从书上学到的知识也都是假的，只有你掌握了自己的认知方法，用自己的方法去认识这个世界，知识才为你所用。我一直用一种简单的“协议模型”来理解和认识工作、社会以及世界。别笑我年纪轻轻，敢谈认识世界，也许明天醒来，我就会鄙视我渺小的看法，但是，在这个时刻，我有自己对世界的认知模型。两年前我给团队做过一场“万物皆协议”的培训，那个时候，更多的关注技术中的协议，现在经过两年的改善，也许下次可以分享一下我脑子里的新的“万物皆协议”的理论了^_^

好不容易经历了三个阶段，以为自己成神了，其实，年轻人，你还是太年轻，醒醒：



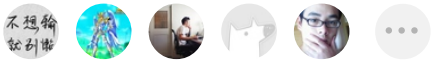
永无止境，靠眼忘不到高度，靠的仅仅是你的想象力！

写在最后

成长即意味着改变，而改变本身又是一件很痛苦的事情，但是改变之后，你能享受到的不仅仅

是一次改变，因为变化会有连锁反应，一次的改变之后，你的心态和你的认知可能会和以前大有不同。

改变只是一个开始！成长之路还很长，共勉！



5 条评论

写下你的评论

蒲良

谢谢你！很有启发！

8 个月前

风满楼

好棒

8 个月前

douglee

有道理

7 个月前