二

00 开篇词 掌握软件开发技术的第一性原理

计算机软件开发是一个日新月异的领域，几乎每天都有新的技术诞生。每隔几年，软件开发领域就会进行一次大的技术潮流变换，所以身处其中的软件开发技术人员也常常疲于奔命，不断学习各种新知识、新技术，生怕被这个快速变革的时代所抛弃。

但是每次从头开始学习一个新的技术，这个过程既痛苦又漫长，好不容易掌握得差不多了，新的技术又出现了，于是不断重复从入门到放弃这一过程。这个过程是如此痛苦、艰难，以至于整个行业形成了一种所谓的“共识”：随着学习能力和体力精力的下降，编程知识和技能逐渐衰退，35岁以后就不能写代码了。

其实很多看起来难以坚持、让人容易放弃的事情，并不是智力、体力或者意志力的问题，更多的是方法问题。很多时候，学习新知识和新技术之所以困难，是因为没有理解这些新技术背后的思想和原理，以及这些新技术诞生的来源。太阳底下没有新鲜事，绝大多数新技术其实都脱胎于一些既有的技术体系。

如果你能建立起这套技术思维体系，掌握这套技术体系背后的原理，那么当你接触一个新技术的时候，就可以快速把握住这个新技术的本质特征和思路方法，然后用你的技术思维体系快速推导出这个新技术是如何实现的。**这个时候你其实不需要去学习这个新技术了，而是去验证这个新技术**，你会去看它的文档和代码，去验证它是不是和你推导、猜测的实现方式一致，而不是去学习它怎么用了。那么，学习一个新技术就变成了一个简单、轻松、快速且充满乐趣的过程了。你不再惧怕学习新技术，而是开始抱怨：为什么技术革新得这么慢，太无聊了。你甚至可以开始自己创造新技术。

第一性原理——建立技术体系的起点

那么如何实现这一美好的愿景，建立自己的技术思维体系呢？

物理学有一个第一性原理， 指的是根据一些最基本的物理学常量，从头进行物理学的推导，进而得到整个物理学体系。有硅谷钢铁侠之称的埃隆·马斯克特别推崇第一性原理，他做电动汽车、做航空火箭，并没有去遵从别人的老路，而是从这个产品最本质的需求和实现原理出发，重新设计了产品最核心的关键以及发展路径，进而开发出自己独特创新的产品。Google的创始人拉里·佩奇说过：“让我自由地从物理规则出发去思考问题，而不是迎合那些

所谓的世俗智慧。”其实也是第一性原理。

第一性原理就是让我们抓住事物最本质的特征原理，依据事物本身的规律，去推导、分析、演绎事物的各种变化规律，进而洞悉事物在各种具体场景下的表现形式，而不是追随事物的表面现象，生搬硬套各种所谓的规矩、经验和技巧，以至于在各种纷繁复杂的冲突和纠结中迷失了方向。

软件开发技术也是非常庞杂的，各种基础技术，各种编程语言，各种工具框架，各种设计模式，各种架构方法，很容易让人觉得无所适从。就算下定决心要从基础学起，上来一本厚厚的《操作系统原理》，好不容易咬牙坚持学完，回头一看，还是各种迷茫，不知道在讲什么。继续学下去，再来一套更厚的《TCP/IP详解》，彻底耗尽了意志力和兴趣，完全放弃。

其实，我们不需要一开始就精通操作系统进程调度的各种算法，也不需要上来就掌握TCP/IP协议里的各种帧格式。我们应该从软件技术的第一性原理出发，了解每个基础技术方向那些最关键的技术原理，明白这些原理是如何和我们日常开发工作发生关系的。

比如我们的程序是如何被操作系统调度执行的？为什么高并发的时候系统会崩溃，原理是什么？在编程时，什么场合下应该使用链表，什么场合下应该使用数组，为什么？当我们使用Hash表的时候，什么情况下它的性能会急剧降低，原理又是什么？我们用Redis这样的分布式缓存的时候，到底要解决什么问题？分布式缓存是如何工作的？还有哪些技术看起来和Redis毫不相干，其实工作原理是一样的？

如果我们能把这些基本问题都回答清楚了，那么这些问题背后的核心技术原理也都理解了，我们就开始建立起自己的技术思维体系了。当有新的问题和技术出现，你就可以思考，这是属于哪个技术领域的？它的核心原理和哪个技术方案本质是一样的？

如果你掌握了软件开发技术的第一性原理，那么当你为了解决某个新问题，去学习和研究一个新技术的时候，就算遇到了知识的盲点，也可以快速定位到自己技术体系的具体位置，进一步阅读相关的书籍资料，这个时候也许你就会深入到操作系统的调度算法实现或者通信协议头信息的具体编码里，但是这时，你不会觉得枯燥无聊，也不会觉得迷茫无措，只会觉得原来如此，太有意思了，甚至觉得这其实可以实现得更好。

专栏如何帮你建立技术体系

我想从软件技术的第一性原理出发，写一写软件技术那些最基本的知识原理和知识体系。在这个专栏中，我对自己过去二十年软件编程生涯和业界的技术发展历史进行回顾总结，将软件知识技术体系分成**软件的基础原理、软件的设计原理、架构的核心原理**三个部分。

软件的基础原理主要是操作系统、数据结构、数据库原理等等，我会从一个常见的问题入

手，直达这些基础技术最本质的原理，并覆盖这些基础技术的主要关键技术点，让你理解这些基础技术原理和你日常开发工作的关联关系，对这些基础技术有一个全新的认知。

在软件的设计原理里，我会讲述如何设计一个强大灵活，易复用，易维护的软件。在这个过程中，应该依赖哪些工具和方法，遵循哪些原则和思想，使用哪些模式和手段。如果软件只是实现功能，那么程序员就没有高下之分，软件也没有好坏之分，技术也就不会有进步。好的软件究竟好在哪里？如何自己也写出一个好的程序？我将在这个模块一一道来。

架构的核心原理围绕目前主要的互联网分布式架构以及大数据物联网架构进行剖析，分析这些架构背后的原理，它们都遵循了怎样的驱动力和设计思想，有哪些看似不同的技术其实原理是一样的，以及如何通过这些技术实现系统的高可用和高性能。

软件开发是一个实践性很强的活动，如果你只是学习技术，那么就是在纸上谈兵。只有将知识技能应用到工作实践中，才能真正体会到技术的关键点在哪里，才能分辨出哪些技术是真正有用的，哪些方法是花拳绣腿。但是公司不是你实践技术的实验室，怎样才能处理好工作中的各种关系，得到充分的授权和信任，在工作中实践自己的技术思想，并为公司创造更多价值，得到更多的晋升和发挥的空间，使自己的技术成长和职业发展进入互相促进的正向通道？我将会在第四模块，技术人的思维修炼和你分享一些这方面的方法和认知。

我在学习几何的时候，开始常常困扰于各种定理、推论，我觉得它们都很相似，以至于进行几何证明的时候，不知道该用哪个。后来我索性不去管这些定理和推论，而是直接从公理开始证明，虽然证明步骤长了一点，但是总归能证明出来。后来做的题多了，发现有些中间推导结果总是重复出现，打开书再学习，发现这些重复出现的中间结果就是各种定理、推论。这个时候我不去记这些定理，也能随心所欲去用它们了。

其实我学几何的这种方式就是第一性原理。第一性原理是一种思维方式，一种学习方式，一种围绕事物核心推动事物正确前进的做事方式。也许这个专栏讲到的很多知识技术你已经掌握，但是这些知识技术和软件技术最基本的原理的关系你也许不甚了解。它们从何而来，又将如何构建出新的技术？如果把这些关系和原理都理解透彻了，你会发现，日常开发用到的各种技术，你不但可以随心所欲地去使用，甚至可以重新创造。

如果说具体的技术是一朵花，那么技术思维体系就是一棵树，希望你跟随我的专栏，种下自己的技术思维体系之树，收获一树繁花。

在学习的路上，你有哪些建议或者心得体会呢？欢迎你分享在评论区，我会和你一起交流这些学习方法，也欢迎把这篇文章分享给你的朋友或者同事，一起交流一下吧！

下一页