第27章、关于写技术书籍/文章的四个指导原则

不知不觉专职写技术文章/书籍的时间已经两年多了（是的，小孩子是离职后才开始写书的），刚开始是摸着石头过河，什么东西都是尝试着来，今天再回头看我最初写的一些东西还是蛮幼稚的，希望再过两年看我今天写的这些东西也能产生这种感觉吧（那才能说明我进步了嘻嘻）。

当我们从事一门工作时，首先要界定清楚我们要达到的目的是什么。对于写技术书籍/文章的我来说，目的很简单，就是：让同学们更快、更舒适的掌握我想让他们掌握的那些知识。达到这个目的比较困难，不过在不断探索中逐渐形成了我自己的一套写作方法论，今天这篇文章就着重于我自己提出的写技术文章的四个指导原则，主要是给下边两类同学看：

- 对于刚刚开始写作的同学们，希望大家可以少走些弯路。
- 对于学不懂某一门专业知识的同学们，虽然我不能让大家立即把专业知识学会，但是至少可以让大家明白自己为什么学不会。

> 小贴士： 我的一个观点：对于大部分工程技术问题来说，理解并掌握它们并不需要很高的智商，但是需要解释这些问题的人下一些功夫，从初学者的角度出发来解释问题。也就是说：学不会东西不怨大家，是我们的文档没写清楚～

好的，废话少说，所谓的写技术文章的四个基本原则就是指：

- 简单
- 简洁
- 有趣
- 系统

我们下边分别来看一下。

简单

计算机是一个非常精密的机器，我们可以用叹为观止来形容，简直比艺术品还艺术品。不过即使是经过成千上万科学家精心研究的这个艺术品，底层也只不过是由一堆晶体管组成的，我们作为程序员平时使用C、Java这些高级语言写出的程序实质上是在操纵这些晶体管。比方说我们用C语言敲了一行`printf`的代码，然后把它编译运行，最后在屏幕上把它显示出来，这样的一个简单的操作，其实是经过层层调用得到的：

- 我们的应用程序（这里是一个C语言的程序）会调用操作系统提供的往屏幕上输出一行字的接口，称之为`系统调用`，不同厂商可以生产不同的操作系统，比方说`Unix`、`Linux`、`Windows`之类的。
- 操作系统接收到需要往屏幕上输出一行字儿的指令和内容之后，会调用相关硬件的机器指

令，这些机器指令其实就是`010101...`这样的二进制代码。不同的厂商生产的不同机器有不同的机器指令体系，比方说`x86`、`MIPS`、`PowerPC`等等等。

- 这些`010101...`二进制代码它们本身没有啥意义，针对同一个指令体系来说，不同的厂商可以针对同样的机器指令画出不同的电路图，比方说对于`x86`机器指令的体系，Intel公司和AMD公司可以真对同一个机器指令开发出不同的电路图，这个电路图就是所谓的微体系结构。
- 电路图其实就是个图，它是由各个具体的逻辑组件组成的，比方说做加法需要加法器，做乘法需要乘法器，存储数据需要触发器吧啦吧啦的。
- 而这些逻辑组件本质上是数字电路，由与、或、非以及其他的一些基本逻辑门构成的。逻辑门还是一个抽象的概念，它假设电路里只有0和1两种状态，但是实际的电路中只能靠电压来代表电路的状态，而电压的变化是一个连续变化的过程，不是单纯的从0V直接跳到5V，所以我们把在某个区间内的电压认为是0，某个区间内的电压认为是1。
- 模拟电路中的电压信号是连续的，它是靠底层的一些硬件来导电的，比方说继电器、真空管、晶体管啥的。
- 继电器、真空管、晶体管这些器件是怎么导电的呢？我们只能认为这是大自然的魔力，或者说大自然的一种规则。

从这个过程可以看出来，这样子的抽象把一个极其复杂的问题分配到了不同层级去处理，就像把军队分成军、师、旅、团、营、连、排、小工兵一样，最高级别的指挥官会下达一个较为抽象的命令，比方说把某个地方给攻占掉，之后这个命令层层落实到最基层，每一层都有自己的任务，但是最具体的任务还是要给小兵去完成。

说了这么多，我们只是想说一个极其复杂的系统背后肯定是有其抽象层级的，作为解释者的我们需要把这样的抽象层级给解构出来，直到解构到足够简单让用户理解，然后再把它们串联起来形成一个大的系统。为了做到`简单`，我们给出一些可以落地的建议：

- 清晰的结构划分。

  这个过程类似于思维导图，我们首先搞清楚自己要讲清楚的主题是什么，然后这个主题是由哪几个部分组成的，每个部分又能继续化为更小的哪些部分。化成的这些部分的相互依赖关系是如何，千万要记住一条硬性规定：不要用一个没有解释过的概念去解释另外一个新概念，这种情况是导致用户学不懂的首要元凶。

- 魔鬼藏在细节里，对一个概念的充分解释

  很多同学以为写的字越少，可能意味着越精华，这纯粹是胡扯。我的经验是我们给出的细节越多，读者更容易理解，读者在越多过程中会产生很多疑惑，细节列出的越多，留给读者产生疑惑的情况就越少，这样他们的思路就更不容易被打断。

- 把某些不适合展开讨论的概念当作黑盒对待，但是要显式的对读者声明

  我们的文章/书籍的篇幅有限，不可能把每个概念都讲述的十分清楚。这时候我们的首要思路是想办法绕过这个概念，在使出浑身解数都绕不过时，需要显式地跟读者说明这个概念我们暂时不讲，只需要知道它有一个什么样的属性就好。

> 小贴士： 就我的写作经验来说，那些当前还不着急介绍的概念十有八九都能绕过。

简洁

简洁直白一点儿的翻译就是废话不要太多。

- 我们在写作之前一定要明确自己介绍的主题是什么，然后与这个主题所有无关的东西都可以算作冗余部分，可以用一句或者两句话介绍带过。因为对于无用概念的过多介绍会让读者降低读者注意力，分散注意力的后果就是他们可能分不清那些是重点，哪些是没用的。
- 避免概念过多出现在一个地方。

  如果你发现自己在某个不长的篇幅里连续引入了大量的概念，而且看上去密密麻麻的介绍性文字，中间也没啥段子的话，那你就要小心了，这样的篇幅容易引起读者不适，很多人会下意识的先扫一眼，发现里边有好多自己不认识的词儿，心里立马就有了抵触，这种抵触情绪会影响到他的阅读体验，从而更难真正理解你所描述的内容。所以如果你需要在某处介绍大量概念的话，这时你就需要考虑如何拆分这些概念，你可以把它们拆成不同的主题，或者在同一主题里拆分到不同的地方去描述。
- 图表让文章立马变得简洁

  语言在图表面前是十分苍白的，真滴～

有趣

首先说一下，有趣和简洁在很多地方都比较冲突，如果你想让文章更有趣，那就得适度牺牲简洁，把握好度比较重要，这个度就像厨师的火候，不可说～

- 能用咱土语描述的就千万别扯过多的专业术语，专业术语用多了，就成了文言文，让人看多了就忘了是个啥意思。有的时候为了表述某个观点你觉得汉语不够使了，你扯点英格蕾丝也是可以的嘛。
- 段子该用就用，甚至可以是我们刻意设计的，笑点掌握在自己手中那是种高级境界，我目前还差着远呢～
- 注意趣味性不能喧宾夺主，沦为儿童读物。

系统

系统可以表述为一种知识的闭环，写书的时候尤其要注意，写单篇文章的时候也要稍微注意一下。

就是说我们把要讲述的那些主题划分成多个模块后，将每一块的依赖关系划分出后，先讲那些简单的，再讲那些复杂的，最后全部都联系到一块就形成了我们要讨论的主题，针对每一块，我们都要考虑清楚：

- 我们在讲什么东西？
- 为什么会有这个东西？
- 这个东西有什么用？

一个概念的产生不是凭空想像出来的，它是有它存在的环境的，我们把它们诞生的情景说清楚，然后把它们和其他模块的依赖说清楚，就可以形成一个浑然天成的体系，用户们也就可以获得一个慢慢的学习成就感。

关于正确性

我们怎么没有唠叨内容的正确性呢？哈哈，这玩意儿还用强调么，多找几本参考书，多对照着源码看一下。虽然我们是人就可能犯错，但是希望可以尽量让错误少一些吧～

结语

希望我们的技术书籍/文章越来越好，曾经帮助过我宣传《MySQL是怎样运行的：从根儿上理解MySQL》的公众号「架构师之路」的博主沈剑老师对我说：“我是技术人，不是商人”，与各位共勉，尽量改变一下国内技术书籍晦涩难懂的现状，一起加油

希望这是各位2019年最爽的一次知识付费，如果各位因为阅读本小册而顺利通过面试，或者解决了工作中的很多技术问题，觉得29.9实在是太物超所值，希望各位能来给点打赏（本人很穷，靠救济生活～ 添加好友可以问关于小册的问题，不过希望不要扯犊子聊八卦了，我其实挺忙的～ 微信号：xiaohaizi4919）。

小贴士： 请允许我鄙视一下那些打着知识付费骗钱的人，除了不生产一点社会价值外，反而生产了数不清的焦虑，让人们连幸福感都丧失掉了。也请各位警惕那些说只要你交几百块钱，就能得到诸如境界上的提升、开阔了眼界、追赶上行业发展趋势之类的课程/知识付费，这类抽象而无法验证的主题都是骗人的。

另外，再次宣传一下我的微信公众号 「我们都是小青蛙」：

写书不赚钱，当然写公众号现在也不赚钱，不过人家说粉丝多了之后就可以赚钱了，希望我可以将写作作为工作，而不用再考虑如何生存下去的问题，谢谢各位。