开发与研发（下）

2011年1月10日
16:10

相对于开发来说，我个人更喜欢研发一点。研发和开发的一个不同之处就是研发有更多的“研究”成分在里面，也就是说研发的时候会有更多“光明正大”的 学习时间，这对于那些对技术本身有追求的工程师来说是很有吸引力的。有一些人做工程师是为了可以创造出好的产品，然后挣大钱或者改变世界；也有一些人做工 程师是因为对技术本身有兴趣，想要好好研究。可以凭借技术名利双收变身成功人士固然很有吸引力，但不关心世事钻研一些自己喜欢的东西也自有它的乐趣在。

研发

如果说开发产品是“输出”，那么学习思考就是“输入”，只有输出没有输入整个人就会废掉，完全沦为一颗螺丝钉。在很多公司尤其是那种经常加班赶项目 的公司，你每天都会处于很忙碌的状态，脑子里想的都是赶紧把指定的任务完成上线。因为时间紧，所以你在开发过程中遇到什么问题都是只求解决，没有心思和时 间去搞明白为什么会出现那种问题，在这样的工作状态下完全没有办法积累工作经验，看上去好像工作了五年，其实是工作了一年，然后重复了四年。

做研发一般不会直接为产品贡献代码，更多做的是一些基础架构或者实验性的产品，所以它有几个很明显的好处。首先，很少开会。其次，没有产品经理。第 三，一般都会把质量放在第一位，时间不会特别紧。这是三个非常巨大的优势，这意味着你绝大部分时间都可以安心学习、思考、设计、编程，幸福指数会飙升。如 果你是做基础架构，那么代码质量就会有硬性要求，你不得不写得健壮、易用、松耦合并且易于调试，要花心思和时间细细打磨，对个人的能力提高、习惯养成和经 验积累都非常有帮助；如果你是做实验性的产品，那么你就有大量的机会和时间去调研最新的技术，而且最棒的是你可以在产品当中使用它们——这对于开发线上产 品的工程师来说是不太可能的，因为不成熟的新技术存在太多未知的风险。

此外，做研发对工程师的素质要求很高，需要很好的技术基础、学习能力和研究能力——我把它看作是一个优点。从个人角度来说，我宁愿一家公司招聘非常严格需 要竭尽全力才可以进去，因为严格的招聘可以保证团队所有成员的质量，不用担心进去之后会“和臭棋篓子下棋”。既然选择去做研发，那么基本可以说明你是一个 对技术有追求的人，也肯定希望周围是一群和你一样的人，而不是连基础知识都不够熟悉的家伙。只有这样一群“互相看得起”的人在一块研究、学习、思考、切磋 才会其乐无穷，才能够产生更多创意，做出好玩的东西。

当然，做研发也有不好的地方。只有大公司才有研发部门，这些公司一般都已经上市或者员工已经很多，你不太可能有机会一夜暴富。当你埋头做了几年研发 之后，某一天去参加同学会，发现大学时候那个数据结构不及格总是求你让他拷贝编程作业的张三衣着光鲜四处敬酒。他所在的公司刚刚上市，因为进去得早，现在 他变成了百万富翁而且荣升高层。于是你忽然开始怀疑自己当初的选择，连学习和编程的乐趣都变得很不真实。所以，如果你渴望建功立业，那么就不要选择做研 发，或者做几年研发之后就出来闯荡。成功需要的条件很多，而编程只是你的优势之一，只有这一个优势你需要太多的运气才可以得到你想要的。

不过，我们也可以换个角度看。“乱世放不下一张安静的书桌”，现在到处都无比浮躁，有个地方可以让你安安心心做一些自己喜欢的事情已经非常难得，多 少人拼命挣钱就是为了可以和你一样做自己喜欢的事情。尽管那么多人在叫嚷“搞原子弹的不如卖茶叶蛋的”，但总有一些人愿意去追求人类最高财富——知识和艺术家般的技艺。

本来做研发成就感会少一点，作为一个 Twitter 的开发工程师看到那么多人在用 Twitter 肯定会特别开心，相比之下某个在 Google 做基础研究的工程师的成就感可能没那么强烈。不过在国内环境比较神奇，开发工程师非但成就感不多，反而会不少挨骂，还经常会有负罪感，相信做过邮件推广和 广告弹窗的工程师都深有体会。这样一来，研发工程师的“清苦”反而变成了一个优点，可以远离很多“不得不做”的违背良心的事情。

相信很多工程师在入行之前是喜欢技术的，但是工作之后发现完全不是自己当初想象的那个样

子，然后就变得失望麻木，不再对技术有热情。其实你可以把热 情延续下去，只不过要去做研发，而不是做开发。大部分由于兴趣而不是生计学习编程的人，内心真正渴望的都是去做研发，只不过没有人告诉他们开发和研发的巨 大差别。现在不少大公司都有自己的研发部门，有一些还成立了自己的研究院，想要一直做技术的同学不妨尝试一下。

如何选择

很多人在大学里之所以会选择计算机为自己的专业，并不是因为自己对计算机和编程有兴趣，而是因为计算机是“热门专业”，在毕业之后也浑浑噩噩地找了 一份工作进入了这个行业，做着自己并不喜欢的事情；还有一些人则是毕业之后找不到工作，然后看到一些培训机构的广告就去报名学习编程，希望广告上描绘的 “月薪过万”不只是一场梦。于是就有了越来越多的“代码民工”，在形形色色的大小公司做着又脏又累的工作，只为了“混口饭吃”。

我并不想批评这些人，毕竟在这个大环境下有着太多无奈，逼得我们无从选择。对于这样一些只想找一份好工作的人，是被骗到这个行业中来的。仔细回忆一 下，这些年来我们看到的业界新闻，了解到的互联网公司文化，大部分都是有关诸如 Google, Facebook 等国外公司的；我们平时学习和使用的技术，几乎都是国外发明的。这让我们深信互联网就是那样美好，那些激动人心的东西触手可及，但请你关上电脑出门好好看 一下周围：这是在中国。互联网没有国界，但互联网公司有。Google 和 Facebook 这样的公司看上去离我们很近，我们每天也使用它们的产品，但国内的互联网公司可能要几百年之后才会有那样的气质和文化。所以如果你不幸误入了这个行业，还 是及早打算改行或者转型做管理比较好，这样就不需要再学习自己并不喜欢的“枯燥”技术了。

对于那些“真的”对技术有兴趣的人，要么去做一个同时具备软件设计能力的开发人员，也就是富有创造力的 Hacker；要么去做一个自得其乐的研发工程师。虽然环境恶劣，但是任何东西都挡不住真正的热爱。在这个几乎人人都把金钱作为衡量标准的社会里，你真是得到了上天的眷顾，不仅能够以自己喜欢的事情谋生，而且收入还过得去。

Hacker 是适合创业的，因为他拥有创造一个产品的全部能力。电影《社交网络》让很多以写代码为生的人产生了幻觉，Facebook 创始人传奇般的经历好像在向全世界宣布：世界是程序员的。很多人只是激动地看到扎克伯格的技术能力，但是却忽视了他的软件设计能力和对产品细节的重视程 度，好像只要埋头编程就可以做出 Facebook。除了优秀的技术能力之外，扎克伯格的思考能力和创造力同样出类拔萃，可以感受得到他眼里的世界是不一样的。我们的工程师又有多少人对生 活中的事物有独特而深刻的理解呢？独立思考也应该是 Hacker 的必备技能。

很多工程师都觉得自己会编程，只是缺少一个“好的 idea”；很多非技术人员则觉得自己有一个“好的 idea”，但是缺少编程能力来实现。要做一个产品，好的 idea 和实现它的能力缺一不可。然而，我们可以看到最后成功的往往是那些非技术人员，因为他们可以清楚地看到编程是一件可以学习的事情；而工程师们则往往天真地 认为好的 idea 靠的是“灵机一动”，不会有意识地培养自己的观察能力和想象力。很多好的 idea 都是来自于平日对生活的敏锐观察和思考，然后这些点在某个时候忽然连成了一条线，把它简单地归结为“天才”是懒惰的做法。

“成为一个 Hacker”和“做研发”，很难说二者哪一个更困难。Hacker 在技术上可以不是一流，但他运用技术创造产品的综合能力肯定是一流的；而研发更注重技术上的造诣和理解程度，关注的是深度而不是广度。如果想要做研发，那 么就要好好把基础知识研究透彻，比如数据结构、算法和网络协议等，不然很容易就会遇到瓶颈。我遇到过的每一位研发工程师都是技术上的大牛，在很多技术问题 上都有非常深刻的见解；他们会从本质上分析问题，而不只是纠结于语言细节。

如果你想要通过自己的作品改变世界，那么就好好提高一下编程之外的能力，做一个好的 Hacker；如果只想埋头技术，就应该选择去做研发。不过，无论是想要做一个 Hacker 还是一个研发工程师，都需要长年累月地不断学习和思考。听上去好像非常辛苦，不过每一个热爱技术的人应该都会把学习和思考当作一种乐趣，而不是一种苦役。 如果你无法享受学习和思考的乐趣，那么还是不要在技术这条路上走下去了，你会活得特别累，并且毫无幸福可言。

在这个充斥着“代码民工”并且缺乏“技术文化”的国度，我们只是关心怎么样可以活得更舒

服，似乎忘记了编程本身所具有的迷人色彩。Joel Spolsky 说过，许许多多的人选择编程，首要的原因就是，他们宁愿将自己的时间花在一个公平有序的地方，一个严格的能者上庸者下的地方，一个只要你是对的就能赢得任 何争论的地方。此外，我觉得选择编程还可以获得最大限度的自由和独立。因为找工作的时候只需要凭借自己的编程能力，所以不需要见人说人话见鬼说鬼话，不需 要去结交权贵达人，不需要去为了所谓人脉去混圈子，也不需要看到邮件列表里有领导的邮件就去“顶”。平日里写写代码，其它时间喝酒吃肉，只交性情相投的朋 友，武侠小说里的畅快适意也不过如此。这种独立和自由是极为宝贵的，你可知道有多少人在醉酒之后哭喊“安能摧眉折腰事权贵，使我不得开心颜”？

所以说，编程这件事情关乎公平，关乎自由，关乎美。而作为一个拥有编程能力的人，你可以亲手创造美。只有艺术家才可以创造美。希望有越来越多的人可以真正领会到编程的魅力所在，喜欢上这种艺术。正如 Raymond 所说，软件设计和实现应该是一门充满快乐的艺术，一种高水平的游戏。你需要用心。你需要去游戏。你需要乐于探索。

黑客事业之未来，全依赖我们今日之创造。

最后推荐一些文章和书，这些文章和书大部分都与技术细节无关，它们讨论的是基于编程的令人心醉的文化，也适合非技术人员阅读。

1. 如何成为一名黑客。所有学习编程的都应该多看几遍这篇文章，至少把 Hacker 和 Cracker 的区别弄清楚。

2. 大教堂和市集。这是一篇关于 Linux 的经典文章。这里需要声明一下，我对那些 Windows 程序员没有偏见，只是我觉得作为一个以编程为职业的人，如果不参观一下 Linux/Unix 的深邃世界，未免太过狭隘。

3. UNIX编程艺术。 这本书虽然名字叫做“编程艺术”，但里面并不讲授如何编程，而是全面展示了迷人的 Unix 哲学和文化。看完之后你会发现，那些看上去不修边幅、整日对着电脑屏幕编写代码的邋遢程序员，对于美竟然会有那么高的追求。“美在计算机科学中的地位，要 比在其他任何技术中的地位都重要，因为软件太复杂了。美是抵御复杂的终极武器。” 这本书的作者 Raymond 同样是《如何成为一名黑客》和 《大教堂和市集》的作者。

4. 黑客与画家。这篇文章是 Paul Graham 写的，文中详细描述了黑客与画家的相似之处。这里所说的“黑客”和《如何成为一名黑客》中所说的“黑客”略有不同，但你可以看到他们很多共同点。本文也已 经被收录到 《Hackers and Painters》一书，该书的中文版《黑客和画家——Paul Graham文集》由阮一峰翻译，应该很快就会面世，我十分期待。

5.创造者的品味。作者同样是 Paul Graham，文章观点独到，见解深刻，每读一次都有新的收获。

6. 软件随想录:程序员部落酋长Joel谈软件。这本书是 Joel Spolsky 的精华文章结集，作者写文章写得非常有趣，擅长讲故事，前几天我翻译的那篇《程序员阿士顿的故事》就是他的手笔。本书由阮一峰翻译，翻译质量非常高，有兴趣的可以先去试读几篇。

7. About Face3交互设计精髓。本书是交互设计领域的经典著作，作者之一 Alan Cooper 原来也是知名程序员，被称为 “Visual Basic 之父”，所以这本书里面对程序员的批评还是很中肯的。另外，书中“设计体贴的软件”的核心思想非常棒，值得程序员好好阅读和思考。

原文地址