孟岩 一个技术文化人的片段感悟

05 March 2015

2003年我加入CSDN，6年之后离开。在2003年之后，我的技术身份就很难界定了。曾经有个朋友称我为“技术文化人”——不以软件开发为生，但整天都在拿软件开发来说事，与这个行业的整体关系可能比任何一个具体的程序员或者架构师都更密切。听上去像是一种恭维，又好像是暗讽，似乎我是站在戏台下面带头起哄的票友。其实在我看来，我与一线技术人的根本区别，在于关注的问题不同：他们关心的如何做好软件，我关心的是如何做好软件人。更确切的说，我关心的问题是，对于一个普通的程序员来说，如何能够通过软件开发这一职业，实现精神上的自由，获得专业上的成就，生活上的安全感，以及对未来的信心。当然，这是很高的目标，在任何社会、任何时代、任何行业，最终都只有一部分人能够到达这个境界。但是，我确实是曾经把这当成工作的中心，毕竟人的问题才是最根本的问题。所以，我在这篇文章里想讲述的，并不是个人的那一单流水账，而是我自己所见所闻所经历的一些点滴片段，其目的也是希望能够就“做好软件人”这一命题，对自己的感悟做一点概括。

时势可用而不可恃

与很多同龄人相比，我接触电脑的时间比较晚，一直到1996年，我才开始学习电脑操作。最初我的目标很简单，就是打打字，画画图，打打游戏，看看影碟。如果说想写程序的话，主要也就是用来应付一些专业课，确实没有在程序设计上深究的想法。但初步学会了一点编程基础之后，意识到编程并不复杂，我就不能自禁地在编程学习上投入越来越多的时间，并最终决定放弃本专业，转行软件开发。后来我了解到，很多同龄人都跟我有相似的“转行”经历。而之所以我们会放弃自己的专业优势，串行到软件开发领域，“兴趣”固然是一方面原因，而更重要的原因，恐怕还是当时的时代潮流。

在九十年代中后期，随着PC的普及和互联网的出现，国内高校中计算机和软件开发成为显赫一时的潮流，“电子化”、“信息化”势不可挡地涌入各个专业领域，给几乎所有成熟行业带来了巨大的冲击。社会上对于软件开发人才的需求突然增加，待遇也有明显优势。再加上美国克林顿时期的“高科技浪潮”、“新经济繁荣”的光环，以及微软、Borland、金山、江民等等成功传奇，使得人们普遍对于计算机和软件行业的未来产生了过于乐观的预期。很多人都觉得，只要搞了软件，成功是唾手可得的。

在这种大潮流之下，学校里能摆弄电脑，特别是会写程序的学生，特别受老师的器重，在就业市场上也有特别的优势，一个个器宇轩昂，盛气凌人。今天听起来可以当笑话，但当时我认识的本校和外校的同学当中，至少有三个人信心满满地宣称自己要做中国的比尔盖茨，其他的人呢？很多只是没有说出来，心里的梦想是一样的。在这样的一种狂热氛围之下，一旦你学会写程序，就不可避免地开始在脑海里编制各式各样的美梦，在这些美梦的诱惑下，“串行”就成了理所当然。

如今，我们这一批“串行生”中，有不少已经成为中国IT产业的骨干，从这个意义上讲，似乎实现了当年的愿望。但IT产业在中国却缩水为一个竞争激烈、外部估值严重下调的普通行业。曾经的美梦，对绝大部分人来说并没有成真，反而是当年大家并不热衷的公务员、国企等去处，靠着似乎取之不尽的公帑和用之不竭的公权成为高通胀时代的幸福特区，这不得不说又是当初没有想到的。于是很多人在失意之余，经常“悔不当初”地设想，如果自己当年不转行，或者如今已经在某机关爬到处长的位置，如何如何。

我不以为然。

不可否认，当年我们这么一批人转行IT，有很大的跟风投机的成分。对于这个行业以及其发展趋势，缺乏基本的了解和积累，对于自己的发展也缺少基本的定位和构想，而是看到这个行业的火爆，就迫不及待地想冲进去分一杯羹。进入这个行业之后，很多人也继续保持投机的心态，今天看到这个火了，就过去捞一把，明天看到那个有上升趋势，就冲过去占个位置。然而事实已经证明，我们所处的这个时代，是一个外部环境变动不居，复杂性不断加剧的时代。就拿IT产业内部的这点事情来说，我在CSDN的六年，说长不长，说短不短，而风潮的变化何等激烈！最初，企业应用市场中.NET与Java之争是所有人关注的中心，谁知Google的崛起掀开Web 2.0大幕，一时之间人人都想着“做网站、赚大钱”。Web 2.0泡沫还未散去，云计算和移动又成为显学，引得无数人趋之若鹜。这还是相对较大的趋势变化，更具体地看，软件工程领域里从CMM到敏捷，J2EE领域里从EJB到SSH，编程语言领域中从Java、C#到动态语言再到Erlang、Scala等新生代语言，还有无数飘起来又迸碎的肥皂泡，如果是追风潮，只会无所适从。因此，在这个时代，我们已经不能基于对外部环境的简单预期来制订自己的规划，只有打好基础，积累优势，守时待势。换句话说，时势可以“用”，而不可以“恃”。我认识的成功的技术人，或多或少都经历过一段咬牙坚持的低谷。冲过低谷，就能够获得别人无法企及的积累，时机一到，便能势如破竹。

从这个意义上说，今天去羡慕公务员，就跟当年投机IT一样。当年不假思索地认为搞软件就能发大财，今天则坚信公权力的挥霍可以长期持续下去，一样的盲目，一样的笃信。但时势的变化，孰能逆料？

抬头看路，埋头赶路

我最初迷上编程，也就是用Turbo C 2.0开发一些DOS下的结构计算和简单的有限元程序，然后用Visual Basic去写一些例子水平的Windows程序，照着书上的例子用汇编语言调用DOS的INT 21h中断。但很快，我就感到不满，我意识到自己对于编程这个领域知之甚少，完全是盲人瞎马，无法确定自己是在朝什么方向前进。今天的年轻学生很难理解当时我所有的这种不安全感，但处于我当时的环境，走错路的风险是现实存在的。身边没有什么高手可以请教，更没有互联网来开拓视野，我甚至从老师那里得到诸如“DOS将会永远是主流”、“Visual Basic将取代C语言”、“C++将被Visual C++淘汰”之类的“专业建议”。这些糟糕的经验让我强烈的不安，所以我决定，在进入这个行业之前，要先对它有一个基本的认识。我的方式，就是大量的、广泛的阅读。

那时候在我出没的范围之内，有三四家上规模的计算机专业书店。没课的下午和周末的大块时间，就成了我的阅读时间。我几乎每周都要去几次，站在书店的角落里，一读就是半天。书店里一排排的书柜，在我看来就是了解软件开发这个行业的一幅幅地图，不管是有关的、无关的，看得懂的、看不懂的，听说过的、没听说过的，我都不放过。通过如饥似渴的阅读，我了解到，除了DOS和Windows 95之外，世界上还有Windows NT和UNIX，了解到Win32不是Windows 3.2，COM跟.com不是一回事，VBA也不是Visual Basic Advanced，了解到Delphi正在跟Visual Basic激烈竞争，了解到C/S体系结构的含义，也明白了当时仍然走红的FoxPro将很快被SQL数据库所取代。逐渐的，我的大脑里出现了一副软件开发领域的全景地图，尽管今天看来，这幅地图非常不精确，也并不全面，但是对当时的我来说，已经可以用它来为自己的学习导航了。

事实上，这段时间的经历对我正反两方面的影响都非常深远，一方面，我由此形成了对软件开发领域的全局性的理解，多年之后，这种理解成为我在CSDN工作的主要优势，也使我对于行业发展的趋势形成了自己的观点；另一方面，过于关注大格局，使我少了埋头钻研的恒心，对关键领域深入不够，这又成为我的遗憾。

我后来在CSDN工作的时候，曾经用“抬头看路”和“埋头赶路”这两个状态来描述一个程序员理想的学习周期。“抬头看路”，就是专门拿出一个时间段，把所关注行业的大趋势看清楚，并结合自己的情况，设定目标和计划。“埋头赶路”，就是在目标和计划设定清晰之后的一段时间里，把自己封闭起来，“两耳不闻窗外事”，不再关心行业的风云纷扰，而是踏踏实实实现自己的目标，形成特长。

拿我自己的例子来说，我那时拒绝了计算机专业课老师主攻Visual Basic的建议，果断地选择C语言作为自己的主攻方向，应该说是基于“抬头看路”所得出的正确决策。而之后过早的从C过渡到C++，则应该说犯了一个错误。C语言的小巧、明快、圆满和强大，迄今无出其右。由于其语言简捷，没什么可学的，学习者的旺盛精力将很快“被迫”转向真正有价值的东西——算法、数据结构、编译、图形、系统编程，等等。我后来认识的很多高手，就是因为早走了几步，“没听说C++”，就在C上下了苦功夫，“埋头赶路”，反而“因祸得福”练成了很强

的动手能力，而能有一方成就。而我过早进入C++之后，在C++的语言里打了几年的滚，反而对于算法、编译、汇编语言等基本领域投入不够，基础没有打牢，离开学校之后不得已花了很多倍的精力来弥补。现在回想起来，这就是专注不够的教训。

到后来我在CSDN工作的时候，这方面的体会就更深。那几年里，为了能够与各路高手平等交流，我几乎涉猎了所有重要的技术领域，研究了大多数热门的技术概念，阅读之广，尝试之杂，远远超过一般软件开发者的需要。正因为这种“博”，使我对于各技术派别以及各主要企业之间的关系和沿革能够了然于心，从而对于行业发展形成自己的见解。这对于我在CSDN的工作来说，固然是一种必要，但是其实身处其中，甘苦自知。俗话说“样样皆通，必然样样稀松”，广泛涉猎的代价就是深入不够，我对此可谓有切身之痛。

反而是到了现在，我可以在业余时间以平和的心态深入研究自己喜欢和擅长的领域，便又可以享受“埋头赶路”、不闻世间纷扰的充实与快乐了。

遇高人不可交臂而失之

我在初步掌握C语言之后不久，就一步踏进C++。C++复杂的语法、强大多样的抽象机制、奇妙的各种语言现象，极大地满足了我的好奇心和求知欲。我买了好几本书，几乎是手不释卷地每日畅游于其中。我不想过多渲染当年学习C++的艰辛，其实对于语言本身，我并没有花多长时间就形成了一个大概的认识。但是C++的真正挑战在于从“知”到“行”。我最突出的印象是在当年学习计算机图形学时，老师布置了一个大作业，我信心满满地希望用最新掌握的Visual C++在Windows下来完成。那时候我已经熟悉Win32 SDK开发，在窗口过程（wndproc）之中援引定义好的一组类，就可以完成手上的工作。构思起来似乎很容易，但是一下手就发现脑子非常乱，要设计哪些类，在类与类之间建立什么样的关系，是不是使用模板，选择似乎非常多，而似乎每一条路都有优势，也有问题。以我当时的经验，完全没有选择的依据。勉强下手之后，很快遇到了一系列的问题，产生了一连串新想法，进一步动手写出一堆新的类，就这样，我在问题的外围架床叠屋地打转，几乎写了一个小小的Windows图形类库，但好像代码越多，距离要做的具体事情反而越远，心里越发虚。到了即将交付任务的日期，我已经积累了一大堆没有测试过的类代码，结果可想而知，当最后我终于将任务代码写完之后，编译运行的结果就是程序崩溃。在进行了几个小时的调试之后，我丧失了耐心和信心，于是推翻全部代码，转而用C风格重写了整个程序。这一次效果非常明显，仅仅熬了一个通宵就完成了一份高分作业。

当时这件事情对我的刺激很大，在一种羞辱的感觉中，我强烈地意识到我的C++水平其实非常低下，于是我就到处寻找能够提高C++水平的书。很自然的，MFC就被我纳入视线之中。

当时正值MFC处于其顶点，书店里介绍MFC的书不但数量多、篇幅大，而且那行文的派头也最足，给我的印象，好像MFC就是软件开发皇冠上的明珠，掌握了它就可以俯瞰众生。所以我买了好几本大部头的MFC书来啃。老实说，如果抱着知其然而不知其所以然的心态去学习MFC，其实也并不是那么困难的事情。在微软强大的开发环境支持之下，照着书上的例子多做多练，上手并不难。但对于我来说，越是使用MFC，我就越是不满。第一，我不明白MFC为什么要这么设计，特别是那个Document/View，到底有何奥妙，第二，我看不懂MFC里那许许多多奇怪的宏，第三，我不知道MFC是怎么跟我熟悉的Windows API环境结合起来的，或者更具体的说，MFC是怎么能够把Windows巨大的switch消息处理结构拆接到一个个类的消息处理成员函数的。简单的说，我完全无法把MFC与我熟悉的C++连接起来，两者之间似乎存在一个巨大的断层，让我觉得MFC完全是另一门语言。我花了不少心思去猜测、分析，并试图阅读MFC的源代码，但是那时候的能力还非常浅薄，完全无法理解MFC的奥秘，也无法弥合那个断层。很多次，我都不禁灰心地想，也许自己并不是写程序的料，或者不是搞C++的料，或许应该放弃。

就在我为这些问题感到困惑和郁闷的时候，在一个阴天的下午，我在一家书店的柜台上发现了一本装帧普通的新书，书名是《深入浅出Windows MFC程序设计》，作者侯俊杰。我只站在那里翻了五分钟，就被其中的内容“雷”得头皮发炸——这本书不但正中靶心地直接打到我的兴趣点上，而且语言之优美，内容布局之巧妙，都是前所未见。记得那本书的价格不菲，我当时着实负担不起，于是找同学借了钱也要把它买下来。在接下来的几个星期里，我每天捧着这本书反复琢磨到深夜，感觉所获得的长进，比前面几年都要多。通过这本书我了解了一个完全超过我之前层面的C++的世界，也把作者的名字牢牢记住。当时我就想，如果有朝一日，能够结识这位侯先生就好了。

几年之后，这本书的第二版以《深入浅出MFC》为名发行，畅销海内，终于有更多的人得以见识这本技术图书的典范之作，也认识了这位侯捷先生。不过这个时候我对于C++的关注点，已经从面向对象的应用框架转向泛型和STL，相反，对于MFC我有了更多批判的看法。在2000年底，我动手翻译了STL之父Alex Stepanov的一个长篇访谈，大约两万多字，翻译之后发表在CSDN网站上。后来又转载到《程序员》杂志上。因为这篇文章，我得以认识了CSDN的掌门人蒋涛，并且经他介绍，终于认识了心仪已久的侯先生。

与侯先生的相识，是我C++学习生涯的一个重要的转折点。当时侯先生也已经将注意力转向泛型和STL技术，并且在《程序员》杂志上发表了著名的《C++大系》系列文章，为国内的C++学习者打开一片全新天地。与侯先生刚刚认识不久，他就通过当时《深入浅出MFC》的编辑周筠女士，向我赠送了好几本“C++大系”中的重要作品。我至今还能回忆起打开那个大包裹时兴奋得几乎要晕厥过去的感觉，也还能清楚得记得那其中的内容：Scott Meyers的Effective C++和More Effective C++，Bjarne Stroustrup的The C++ Programming Language，Matt Austern的Generic Programming and the STL，Nicolai Josuttis的The C++ Standard Library和那时刚刚出版的Andrei Alexandrescu成名作Modern C++ Design。一下子有了这么多经典，我几乎废寝忘食的阅读、试验，遇到困难，就用邮件向侯先生请教，在侯先生的指导之下，仅仅短短的几个月，我对于C++的理解便上了一个大台阶。特别是Scott Meyers的两本书，对我的作用可以说是醍醐灌顶、脱胎换骨。在那之后，侯先生又邀请我与他合译The C++ Standard Library，以共同合作的方式给我另一个层面的教益。在那之后几年里，侯先生总会时不时的给我帮助和关怀，帮我购买珍贵的资料，关心我的事业和生活。与侯先生的交往，套用一句俗话，“千言万语说不尽”，但如果可以概括的话，可以归为八个字：知遇之恩，师生之情。至今，这段回忆对我来说，是深藏心底里的一份温暖，也是一份歉疚。侯先生曾寄希望我走入技术写译和培训的行业，并几次为我创造这样的机会，但是我却最终没有从命。虽然多为国内现实所限制，却也少了报答先生的机会。

实际上侯先生不光帮助了我一个人，在七八年前，我们这群在内地的C++爱好者中，先后直接受到侯先生帮助的有数十人，其中有不少与我保持联系。据我的观察，这些人中的大部分都有着还算不错的发展。而如果算上侯先生图书的读者，侯先生帮助的人何止十万众。一个人的贡献，可至于斯！

所以每当我总结这段历史的时候，就会不禁感叹，对于一个学习者来说，高人的点拨确实可以令人“顿悟”，走到一个全新境界。因此，遇高人不可交臂而失之。我也知道，像侯先生这样品质和才情的高人毕竟是极少，但我认识的大部分技术高手，其实都是可师之人。如果能够放下身段，认认真真向高人请教，那么对于自己的成长和发展，都将有莫大的好处。

文武之道

在技术这个行业观察思考了多年之后，我认为我发现了程序员实现事业发展的一个关键原则，那就是在编程技术上保守一点，而在专业及行业领域进取一点。我称之为“技术组合的文武之道”。有趣的是，这个“文武之道”，恰好与一般程序员的实践相反——大部分程序员在编程技术上比较激进，却疏于在行业领域下功夫。

我最早注意到这个问题，是在研究生实习期间。在研究生的第二学年，我被导师派往清华同方参与一个大型专业软件项目的开发，体验到了真实环境下的团队软件项目开发。这个项目由于脱离具体的条件，目标过高，与同时期很多雄心勃勃的科幻项目一样，最终都以失败告终，但是这段经历却使我受益良多。正是在这个专业项目的开发当中，我重新认识了领域知识与编程技术之间的主次关系。

项目本身的目标是开发建筑工程企业的ERP系统，第一阶段的任务是形成概预算自动计算功能，其中涉及大量的图形绘制、对象建模、数据存储等我非常感兴趣的技术内容。在参与这个项目的初期，我非常兴奋地设想过一个雄心勃勃的技术方案：用VC/MFC为开发工具，自主开发图形库和建筑构件类库，支持3D可视化建筑建模，并且用类似对象数据库的方式进行持久化，等等。然而真正进入这个项目，我发现项目主管并没有带领我们冲向这些令人兴奋不已的技术高峰，而是一个会接着一个会的分析需求，研究当时的概预算规范，了解有关领域知识。这个冗长繁琐的过程，让我大失所望。然而更令人郁闷的还在后面，当项目进入到编码阶段时，项目主管竟然选择Visual Basic作为开发工具，而数据库服务器更是非常阳春的Jet DB，也就是Microsoft Access。我对此强烈不满，几次酝酿之后，终于找到项目经理，反应我的想法。这位项目经理是清华的毕业生，已经通过Visual C++的MCSD认证，是当时极为罕见的技术人才，在听了我的想法之后，他陈述了自己的看法。他说，这个项目是一个专业软件，主要的困难和障碍都集中在专业领域，而在编程技术本

身，无论是VB还是Jet DB，都足以在现阶段满足要求。在这样的情况下，如果选择MFC或者其他更酷的技术，无疑会大大增加技术实现的难度，纯属是毫无意义的自找麻烦。他的原则，就是尽可能用可实现项目需求的最简单的技术来完成任务，因为技术越简单，项目风险就越低，团队开发的管理成本就越小。

事实上，当年我骄气很盛，并没有被说服，反而觉得他缺少冒险精神。但是他对于编程技术与领域知识之间轻重关系的阐述，确实也引起了我的思考。而且在项目实践中我也确实发现，相对于变化多端的概预算规则，在屏幕上画个3D的房子并不是这个项目的关键。在一起工作的专业程序员们，似乎很快就可以完成类似这样的任务，但是面对复杂的业务规则，他们就莫名其妙，非常无助了。

毕业之后，我的第一份工作是在联想做掌上设备的软件开发，在那里我也观察到相同的问题。在联想，我搞了一些编程技术上的创新，把SGI STL和 Boost的几个组件移植到Windows CE平台上，并自己写了一个远程设备的软件调试库，还为几个自主软件设计了非常符合设计模式精神的架构。但是所有这一切都没有得到大家的认可，大家关注的中心，是新设备的产品特色，软硬件配置，营销重点，成本控制，项目实施等等。领导和同事们对于我的这些努力，既没有鼓励，也没有反对，而是任我自便，我当时对此确实是大有“怀才不遇”的抱怨。但几年之后，当我有了更多的职业积累、更广阔的视野和更成熟的心态之后，我才能更客观地反思自己在联想期间的经历。其实，无论是手机、掌上电脑还是平板电脑，消费电子类产品的开发当然是以产品设计、营销规划和工程控制为中心，这才是这个行业的关键领域知识。相比之下，什么STL，设计模式，完全不是重点。如果我当时的心态更成熟一些，能够主动学习和掌握消费电子产品的产品设计专业知识，让自己的编程技术能够为组织的整体目标服务，那么我今天的职业发展可能会是完全不同的另一番模样。

进入CSDN之后，我的所见所闻就更多了。大量的技术高手，把主要精力放在“改善程序员开发体验”的事情上，比如新潮酷派的语言，漂亮的开发环境，方便的代码生成和魔术般的设计抽象，倒是能赢得程序员圈子里的一片叫好声，但是却得不到客户的认可。反而是很多团队，由于对应用理解到位，用普通的技术为客户创造了巨大的价值。比如我认识的一家企业，经过深入交流和分析之后，发现移动财务审批是客户没有提出来，但却非常需要的功能，于是用简单可靠的技术手段，实现了这个功能。客户非常兴奋，大加赞赏，甚至破天荒地将其在全国各地的中层干部召集到北京，专门针对这个功能进行培训。

这些经历和见闻，逐渐使我意识到了领域知识与编程技术之间的主次关系。事实上，从客户和管理者的角度来看，你使用什么先进的编程技术并不重要，重要的是你的软件能够正确地做好该做的事情，你是否能对所解决的现实问题有深刻的认识，有所洞见，有所创新。很多时候，在程序员圈子里被追捧的新概念、新思想和前卫技术，实际上并没有经过时间和实践的检验，存在很多问题，过于积极地引入它们，对于客户的利益，反而是一种危害。真正能够意识到这个问题的开发者，在编程技术本身上会趋向于稳妥保守，不急于追新赶潮，宁可做一个后知后觉分子。但在行业领域，他们则完全是是另一个姿态，积极进取，敢于创新和尝试，总是殚精竭虑地思考如何为客户提供更合用的功能，更高的价值。在我所见范围之内，凡是这样做的，不论是企业还是个人，也无论是在企业应用市场还是在消费类市场中，都能够更快的走向成功。

正是基于这个认识，我在很多场合都提出程序员要懂行业，或者重视领域知识。我相信，如果能够掌握好编程技术和行业知识之间的文武火候，作为程序员个体，无论是在企业中服务，还是自己创业，成功会相对容易些。

结语

2009年，我开始了一段新的人生尝试，究竟将走向何方，尚未可知。但对于我来说，作为“软件文化人”的这段经历，已成为我生命中的一段充满精彩与遗憾的篇章。当然，现在还远远没有到正式总结时候，这段旅途对于我的意义，可能也尚未真正揭示出来。我在这里用了这样一种松散的手法，随意收集了几个片段和感悟，由于漏掉了许许多多帮助过我的人，这篇东西是不可以称为“成长故事”的。但写下来的这几个散片，倒也确实是我有感而忆，有感而发。不知道过几年以后，再读到这些，是否又会有不同的看法？那就不得而知了，到那时再说吧。

作者简介

孟岩，现就职于IBM中国公司企划传播部。曾任CSDN和《程序员》杂志技术总编。

story [15] (/categories.html#story-ref)

insight [6] (/tags.html#insight-ref)

← Previous (/blog/02/19/2015/summary-of-2014) Archive (/archive.html)
Next → (/blog/06/19/2015/github-star)