现代软件工程个人体会

龙秋娴

我在本科开始学习计算机科学与技术,通过软件工程课程让我对软件工程有了初步的认识。研究生阶段在章老师的带领下又学习了现代软件工程的课程,让我对软件工程有了更近一步的学习。

软件工程牵涉的范围很广,软件工程的内容对于投身IT产业的我们来说是非常重要的。假期期间在企业的软件测试部做过4个月的实习生,参与过一些项目.从产品平台上看是Web类型的项目；从团队类型来看，是互联网产品团队。每个项目都有不同的挑战，工作也一直在改进，虽然没有达到最理想的状态，但是我一直觉得软件工程是一件很有意思的事情。

老师上课的方式很独特，为避免乏味的课堂,老师让学生成为了主角,带动我们的积极性,通过自我学习然后分享,老师就会了解到我们接受的程度,也会通过学生曾经作过的项目运用软件工程的知识进行分析，让我们印象深刻，生动形象的理解软件工程。

通过对现代软件工程《构建之法》这本书17章的学习，让我认识到软件工程对软件的重要性，因为软件工程的目标就是创造“足够好”的软件。软件工程是软件的一部分，我们在编写软件的过程中会经历不同的阶段，但是了解能够透彻的了解软件工程，成为一名合格的软件工程师却需要更长的路，更多的努力。软件有它的特殊性，软件工程的“工程”二字所代表的是大量的计算，计划，各类计划的协作，以及人，机械的劳作，软件工程对软件的意义就不言而喻了。

单元测试和效能分析工具是我们必须掌握的基本技术，一个程序员要写好一个功能，就要对单元测试负责，单元测试应该覆盖所有的代码路径，以及要保证代码的独立性和效率。效能是每个程序员都梦寐以求的东西，效能分析工具可以帮助找到程序的效能瓶颈，从而改进程序，对于没有分析就盲目优化程序，可能会事倍功半。一个团队需要一定的流程来管理开发活动，每个工程师在软件生命周期的工作也应该有一个流程。

在软件开发流程中，不管有团队的流程，还包括个人开发流程，每个人的工作质量直接影响最终软件的质量。要想成长为一名初级软件工程师，要积累软件开发相关知识，提升技术技能。积累问题领域的知识和经验，一个工程师可以掌握更广泛、更深入的技术和问题领域的知识，对通用的软件设计思想和软件工程思想的理解，提升职业技能，包括对自我管理的能力，表达和交流的能力，与人合作的能力，按质按量完成任务的执行力，这些能力在IT行业和其他行业都很重要，最后，实际成果是最重要的评价标准。PSP认为有四个因素可以衡量软件开发的工作量和质量，分别是项目/任务的大小，时间，质量，是否按时交付。并且对软件工程师的职业发展有了大概的了解。

现代软件产业现在发展很迅速，规模也越来越大，软件编写的团队也不可能由一个人完成，在合作中完成的软件，需要一个好的代码规范和设计规范，这样才可以让团队中的程序员可以互相看懂团队中每个人的代码。代码的风格规范，代码设计规范可以统一团队中程序员开发的代码都统一，且有自己的规范。代码复审可以更正明显的错误，对于无法很快更正的错误，要在项目管理软件中创建Bug把它们记录下来，把所有的错误记录下来，作为常犯错误的汇总，可以作为复查的第一步。结队编程是随着敏捷思潮的兴起而广为人知，结队编程在部分编程情况适合，并不是适用于所有的开发情况。两个人的合作也会经历不同的阶段，合作是有技巧的，正确的给予反馈都会影响两个人的合作。

软件团队的模式和开发模式针对不同的情景各有优劣，正确的选择模式不仅可以使软件开发更加顺利，而且会事半功倍。对于敏捷开发可能是大家慢慢总结出来的，敏捷不是万能的，敏捷的方法可能会更早的知道能否如期完成任务，敏捷开发可以看成是软件开发方法的思潮产物。MSF没有敏捷的宣言，有9条基本原则：推动信息共享与沟通，为共同的远景而工作，充分授权和信任，各司其职，对项目共同负责，交付增量的价值，保持敏捷，预期和适应变化，投资质量，学习所有的经验，与顾客合作。MSF团队模型中，任何技术项目都必须达到特定的关键质量目标，才能够被认为是成功的项目。MSF过程模型时从传统的软件开发瀑布模型和螺旋模型发展而来的，它把瀑布模型中基于里程碑的规划优势与螺旋模型中增量迭代的长处结合了起来。MSF敏捷开发模式吸收了近几年来在软件业流行的各种“敏捷”开发模式的优点，认识到目前大部分软件是以网络应用相联系的，强调和用户更紧密地交流，快速碟带，避免不必要的过程。

软件需求是一个软件最开始的基准，准确全面地找到这些需求可以从不同的角度进行划分，以及要考虑软件产品的利益相关者，通过用户调查获取用户需求。还要考虑典型用户和场景，用户体验的标准。在软件团队中了解到项目经理这个角色的职责和任务，以及这个角色的能力要求。

目前，计算机技术正日益成为推动经济、工业、行政、医疗、教育、娱乐和整个社会发展的核心技术。正如软件工程专家戴维所说的最有用的技术不在将来,而是已经出现好些年了,只不过我们没好好用。我要不断总结不断重新学习，深刻理解软件工程的知识，运用到软件产品的产生的过程中。