# 软件工程管理

## 1. 开源软件开发

* 高质量：可用性不理想：开发者缺乏在可用性，HCI方面的设计经验，常常会模仿商用软件的设计，缺乏自主创新，导致可用性和用户满意度不高；发布的频率过高造成缺乏成熟的设计：大部分开源的项目都没有得到详细并有计划性的设计；参与的人员太多使得设计的质量不高；文档质量不高：开源软件的文档质量并没有商业软件那么好，商业软件有完善的技术文档和用户手册，但很多时候看文档比看源码更有效率；软件越来越复杂而难以控制：参与开发的人员众多，并且开发者们都往往按照自己的需求以及意愿来进行开发；集成和版本控制的成本增加；过多冗余的功能。
* 高开发效率：需求获取及分析：不存在需求不明或与用户难以沟通的问题；设计：不一定要经历从概要设计到详细设计的过程；实现：开发者分散开发，并行工作；可进行快速的bug修复。
* 高强度测试：可以有效地利用开源社区来查找和改进软件的缺陷，测试团队庞大，如果有一个足够大的beta测试人员和协作开发人员的基础，几乎所有的问题都可以被快速的找出并被一些人纠正\*\*（如果有足够多的眼睛，所有的错误都是浅显的）

## 2. 敏捷

* **敏捷的本质：**敏捷的本质，是承认软件开发的复杂性。而且承认，这种复杂性，达到了这样的一种程度：“无法通过足够充分的前期准备，而消除后续的风险。甚至于，前期准备得越是充分，后续的风险越大”；软件开发具有复杂性和可变性；软件开发需求可变性“实际上具有其合理性”。
* 高效率：敏捷开发是轻量型的开发方法，它反对传统、庞大、重型的过程；强调与人交流的重要性，提倡用高质量的软件代替文档，具有能够适应需求变化，进行快速开发的能力。这类方法以快捷、轻便的思维方式，迅速解决了一些传统软件开发中存的问题，提高了软件企业的生产效率，得到了迅速的推广
* 高适应性：产品角度：适用于需求不明确并且快速改变的情况，如系统有比较高的关键性、可靠性、安全性方面的要求，不完全适合；组织结构的角度：组织结构的文化必须支持谈判、人员少而精并且彼此信任、开发人员所作决定得到认可，环境设施满足成员间快速沟通之需要；适用于30人及以下较小队伍、较小规模。
* **敏捷VS计划驱动：**适应性,短计划VS可预期(严格遵循预先计划的需求、分析、设计、编码、测试的步骤顺序进行，变更代价高昂导致难以应对变更)；短周期VS长周期；可运行软件进行度量VS可交付产物或步骤成果进行度量；敏捷仍有严格的规范,与最终产物有关。
* TDD：在开发功能代码之前，先编写测试代码，然后只编写使测试通过的功能代码，迭代该过程，直到所有的测试通过哟，然后重构代码优化。（不可运行-可运行-重构）。测试用例用来表达需求。TDD**实践问题**：测试范围的确定；TDD导致大量的Mock和Stub；Test Case并没有想像中的那么简单：测试用例的完备与否，测试代码本身逻辑的正确与否都依赖于程序员。
* SCRUM：是一个框架，包括Scrum团队及其相关的角色、事件、工件和规则。框架中的每个模块都有一个特定的目的，对Srum的成功和使用都至关重要。Scrum框架包括三个角色（产品负责人、Scrum Master、团队）四个仪式（Sprint计划会议、每日站会、Sprint评审会议、Sprint回顾会议）三个物件（产品BackLog，Sprint BackLog，燃尽图）。具有可操作的过程管理，而xp具有较强的操作性。
* **中国实施敏捷可能会遇到的困难：**1.文化冲击，敏捷导致原来项目管理模式的改变，在企业里推广敏捷很大程度上会触及到企业文化的变革，企业可能无法快速适应2.当遇到困难的时候，他们会选择性的回到其习惯的开发方式，或者根据自己的情况进行裁减，选择性地实施，可能就会变成半敏捷的方式3.学习成本

## 3. 测试

* **软件测试**的经典定义是：在规定的条件下对[程序](https://zh.wikipedia.org/wiki/%E7%A8%8B%E5%BA%8F)进行操作，以发现[程序错误](https://zh.wikipedia.org/wiki/%E7%A8%8B%E5%BA%8F%E9%94%99%E8%AF%AF)，衡量[软件质量](https://zh.wikipedia.org/wiki/%E8%BB%9F%E9%AB%94%E5%93%81%E8%B3%AA)，并对其是否能满足设计要求进行[评估](https://zh.wikipedia.org/w/index.php?title=%E8%AF%84%E4%BC%B0&action=edit&redlink=1)的过程。
* 自动化测试工具：
* 随机测试
* 众测