软件工程—软件生命周期与过程模型

（一）软件生命周期

软件定义

问题定义

可行性研究

需求分析

软件开发

总体设计

详细设计

编码

单元测试

综合测试

运行维护—持续满足用户需要

（二）开发模型

1.瀑布模型

模型优缺点

瀑布模型**适合用户需求明确、完整、无重大变化**的软件项目开发。瀑布模型的成功很大程度上是由于它基本上是一种文档驱动的模型。

“瀑布模型是由文档驱动的”这个事实也是它的一个主观缺点。实际项目很少按照该模型给出的顺序进行，用户常常难以清楚的给出所有需求，用户必须有耐心，等到系统开发完成

2.原型模型—快速原型模型

在**用户不能给出完整，准确的需求说明**，或者开发者不能确定算法的有效性、操作系统的适应性或人机交互的形式等许多情况下，可以根据用户的一组基本需求，快速的建造一个原型（可运行的软件），然后进行评估，进一步精化、调整原型，使其满足用户的需求，也使开发者对将要做的事情有更好的理解

可能存在的问题

1. 为了使原型尽快的工作，没有考虑软件的总体质量与长期的可维护性
2. 为了演示，选择的不合适的系统，语言，算法，并成为了之后系统的组成部分
3. 开发过程不便于管理

正确的使用方式

建造原型的目标仅仅是为了定义需求，之后就可被抛弃（或部分抛弃），实际的软件在充分考虑了质量和可维护性之后才被开发

3.增量模型

是一种逐渐开发并逐步完善的软件版本的模型

优点

1. 在较短的时间内可以向用户提交部分工作的产品，并分批逐步的向用户提交产品，从第一个构建交互后，用户就能做一些有用的工作
2. 整个软件产品被分解为很多个增量构建，开发人员可以一个构件一个构件的逐步开发
3. 给予用户充足的时间来适应新的产品
4. 采用此模型需要更加精心的设计，但是在设计阶段多付出的劳动将在维护阶段得到回报

困难

1. 每个新增量构件集成到软件体系结构中时，必须不能破坏原来已经开发的出的产品，向软件体系结构中增加新构件的过程也要简单，方便，也就是说，**软件体系结构必须是开放的**
2. 开发人员既要把软件系统看作整体，又要看成可独立的构件，相互矛盾
3. 多个构件并行开发，具有无法集成的风险

4.螺旋模型

螺旋模型将瀑布模型和增量模型结合起来，**加入了风险分析。**在该模型中，软件开发是一系列的增量发布，早期的迭代中，发布的增量可能是一个纸上的模型或原型，在以后的迭代中，逐步产生系统更加完善的版本

螺旋模型的基本思想是降低风险

优点

1. 对可选方案和约束条件的强调有利于已有软件的重用，也有助于把软件质量作为软件开发的一个重要目标
2. 减少了过多测试或测试不足
3. 维护与开发之间没有什么本质区别

特点

1. 风险驱动，需要经验相当丰富的风险专家
2. 主要适用于内部开发的大规模软件项目，随着过程的进展演化，开发者与用户能更好的识别与对待每一个演化级别上的风险
3. 随着迭代次数的增加，工作量加大，软件开发的成本增加

5.喷泉模型

特点

主要支持面向对象的开发过程体现软件创建所固有的迭代与无间隙的特征

具体特点

1. 开发过程有分析、系统设计、软件设计、实现4阶段
2. 各个阶段相互重叠，它反映了软件过程并行性的特点
3. 以分析为基础，资源消耗成塔型
4. 反映了软件过程迭代性的自然特征，从高层返回底层无资源消耗
5. 强调增量开发，整个过程是一个迭代逐步提炼的过程

6.RUP—软件统一开发过程

此过程重复一系列周期，每个周期由一个交付给用户的产品结束

每个周期划分为初始，细化，构造，与移交四个阶段，每个阶段围绕着5个核心工作流

（需求，分析，设计，实现，测试）分别迭代

传统计划驱动模型的局限性

1.传统计划驱动的开发方法强调过分过程控制 ，**很多的时间用于文档与维护**，开发的时间对应减少，**主要依赖过程控制**，而不是程序员的自我管理，开发过程的管理复杂且低效

2.在计划驱动方法中，过程与工具不是为程序员服务的，而是**为管理者服务**的，程序员成为了工具与过程的奴隶，极大的阻碍的生产率的提高。

敏捷开发与极限编程

概念

1.敏捷过程是容易适应变化并迅速做出调整，在保证质量的前提下做到文档适量适度。

2.敏捷开发并不是一个具体的过程，而是一个概念性的术语，具有**以人为核心，循环迭代，响应变化**的特点，着眼于高质量的快速交付令客户满意的工作软件

3.开发过程以代码为核心，而不是以文档为核心。

4.以人为本。程序员在软件开发中不再是单纯被管理的对象，而是开发的主体

如何保证开发进度，效率，质量

1.这个问题的答案就是靠人自我的管理，团队自我的管理

2.敏捷方法抛弃了机械、严格的过程控制，就必须依赖于程序员和开发团队的高标准自我要求：严格的自律，团队合作精神，个人高度自觉的主动性，责任感。

极限编程

极限编程是敏捷过程中最有名的一个, 适于小型项目

适用范围：具有有限资源及有限时间的小项目

优点

1. 极限编程加强开发者与用户的沟通需求，让客户全面参与软件的开发设计，保证变化的需求及时得到修正
2. 极限编程注重用户反馈与让客户加入开发是一致的，让客户参与就是随时反馈软件是否符合客户的要求

传统的方法学

传统的软件工程采用瀑布模型作为软件工程的基本模型，把**软件开发和运行过程划分为六个阶段：软件计划、需求分析、软件设计、程序编码、软件测试、运行和维护**等，强调各阶段的完整性和先后顺序，根据不同阶段的工作特点，运用不同的手段完成各阶段的任务

面向对象方法学

优点

1. 符合人们通常的思维方式
2. 高度连续性
3. 重用性好
4. 可维护性好

小结

软件 =程序+数据+文档

软件危机: 原因,现象,办法 (软件工程学)

软件工程(学):开发、运行和维护软件的系统方法

软件工程3个要素：方法、工具和过程。

软件生命周期:定义,开发,运行维护

软件过程:瀑布模型+RUP