2、软件工程

软件工程是将系统化的、严格约束的、可量化的方法应用于软件的开发、运行和维护，即将工程化应用于软件。

软件工程方法学包括三个要素：方法、工具和过程。

二、软件生命周期

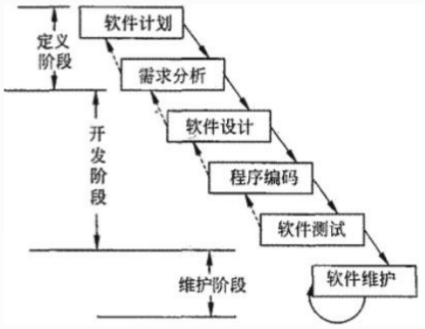
是指软件的产生直到报废的生命周期，包括：问题定义、可行性分析、需求分析、总体设计、详细设计、编码、测试、运行维护等阶段。

三、软件开发模型

常见的开发模型有：瀑布模型、增量模型、螺旋模型、喷泉模型、智能模型、V模型、快速应用开发模型、构件组装模型、敏捷方法和统一过程等。

1、瀑布模型

瀑布模型也称为生命周期法，是结构化方法中最常用的开发模型，它把软件开发的过程分为软件计划、需求分析、软件设计、程序编码、软件测试和运行维护6个阶段。



瀑布模型的优点：

(1)为项目提供了按阶段划分的检查点。

(2)当前一阶段完成后，只需要去关注后续阶段。

(3)它提供了一个模板，这个模板使得分析、设计、编码、测试和支持的方法可以在该模板下有一个共同的指导。

瀑布模型的缺点：

(1)各个阶段之间产生大量的文档，极大地增加了工作量。

(2)由于开发模型是线性的，用户只有等到整个过程的末期才能见到开发成果，从而增加了开发风险。

(3)不适应用户需求的变化，并且在需求分析阶段不可能完全获取。

(4)在软件开发前期未发现的错误传到后面的开发活动中时，可能会扩散，进而可能会导致整个软件项目开发失败。

所以，瀑布模型适用于需求明确或很少变更的项目

2、快速原型模型

快速原型是利用原型辅助软件开发的一种新思想。

经过简单快速分析，快速建造一个可以运行的软件原型 ，以便理解和澄清问题，使开发人员与用户达成共识，最终在确定的客户需求基础上开发客户满意的软件产品。

3、演化模型

也称为变换模型，根据用户的基本需求，通过快速分析构造出一个初始可运行版本(原型)，然后根据用户在使用原型的过程中提出的意见和建议对原型进行改进，获得原型的新版本。重复这一过程，最终可得到令用户满意的软件产品。

快速原型模型是“抛弃式”的，演化模型是“渐进式”原型方法。演化模型特别适用于对软件需求缺乏准确认识的情况。

演化模型的优点：

(1)很早就可以验证是否符合产品需求。

(2)风险管理可以在早期就获得项目进程数据，可据此对后续的开发进度作出比较切实的估算。增加项目成功的机率。

(3)经验教训能反馈于本产品的下一个循环过程，提高质量效率。

(4)心理上，开发人员早日见到产品的雏型，是一种鼓舞。

(5)使用户可以在新的一批功能开发测试后，立即参加验证，以便提供非常有价值的反馈。

演化模型的缺点

(1)产品需求在一开始并不完全弄清楚的话，会给总体设计带来困难及削弱产品设计的完整性，并影响产品性能的优化。

(2)如果缺乏严格的过程管理，这个生命周期模型可能退化为一种原始的无计划的“试 - 错 - 改”模式。

(3)用户接触开发中的尚未测试稳定的功能，可能对用户都产生负面的影响。

4、增量模型

融合了瀑布模型的基本成分和原型实现的迭代特征，是第三种原型化开发方法，但它不是“抛弃式”的，也不是“渐进式”的。

增量模型把软件产品划分为一系列的增量构件，第一个增量往往是核心的产品，即第一个增量实现了基本的需求。客户对每一个增量的使用和评估都作为下一个增量发布的新特征和功能，这个过程在每一个增量发布后不断重复，直到产生了最终的完善产品。

增量模型与原型实现模型和其他演化方法一样，本质上是迭代的，但与原型实现不一样的是其强调每一个增量均发布一个可操作产品。

增量模型的特点是引进了增量包的概念，无须等到所有需求都出来，只要某个需求的增量包出来即可进行开发。

增量模型优点

(1) 人员分配灵活，初期不用太大投入。

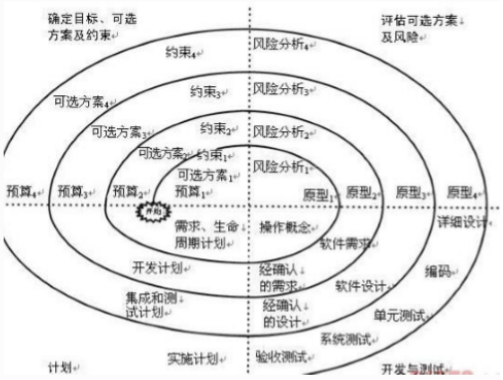
(2) 每隔一小段时间就提交用户部分功能，用户可以直观感受项目进展，及时试用产品功能。

(3)有利于风险的把控。

增量模型将功能细化、分别开发的方法适应于需求经常改变的软件开发过程。

5、螺旋模型

将瀑布模型和演化模型相结合，综合了两者的优点，并增加了风险分析。它以原型为基础，沿着螺线自内向外旋转，每旋转一圈都要经过制订计划、风险分析、实施工程及客户评价等活动，并开发原型的一个新版本。经过若干次螺旋上升的过程，得到最终的系统。



螺旋模型的优点：

(1)设计上灵活，可以在项目的各个阶段进行变更；

(2)以小的分段来构建大型系统，使成本计算变得简单容易；

(3)客户始终参与每个阶段的开发，保证了项目不偏离正确方向；

(4) 随着项目推进，客户始终掌握项目的最新信息，从而能够和管理层有效地交互。

螺旋模型的缺点：

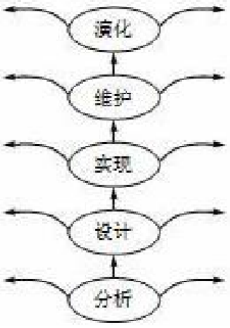
(1) 需要具有相当丰富的风险评估经验和专门知识，如果未能够及时标识风险，势必造成重大损失；

(2)过多的迭代次数会增加开发成本，延迟提交时间。

6、喷泉模型

是一种以用户需求为动力，以对象为驱动的模型，主要用于描述面向对象的软件开发过程，该模型认为软件开发过程自下而上的，各阶段是相互迭代和无间隙的。

无间隙是指在开发活动中，分析、设计和编码之间不存在明显的边界。



7、基于构件的开发模型

将整个系统模块化，并在一定构件模型的支持下复用构件库中的一个或多个软件构件，通过组合手段高效率、高质量地构造应用软件系统的过程。

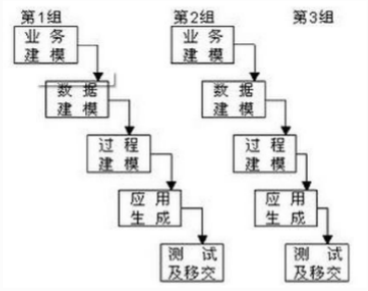
基于构件的开发模型由软件的需求分析和定义、体系结构设计、构件库建立、应用软件构建以及测试和发布5个阶段组成。

优点：构件复用，提高了软件开发的效率。构件可由一方定义其规格说明，被另一方实现。然后供给第三方使用，构件组装模型允许多个项目同时开发，降低了费用，提高了可维护性，可实现分步提交软件产品。

缺点：缺乏通用的组装结构标准，因而引入了较大的风险。可重用性和软件高效性不易协调，需要精干的有经验的分析和开发人员。客户的满意度低，并且由于过分依赖于构件，所以构件库的质量影响着产品质量。

8、快速应用开发模型（RAD）

是一个增量型的软件开发过程模型。强调极短的开发周期。RAD模型是瀑布模型的一个“高速”变种，通过大量使用可复用构件，采用基于构件的建造方法赢得快速开发。如果需求理解得好且约束了项目的范围，随后是数据建模、过程建模、应用生成、测试及反复。



原型可以分为三类：

- 探索型原型

主要用于需求分析阶段，目的是要弄清用户的需求，并探索各种方案的可行性。它主要针对开发目标模糊，用户与开发人员对项目都缺乏经验的情况，通过对原型的开发来明确用户的需求。

- 实验型原型

主要用于设计阶段，考核实现方案是否合适，能否实现。对于大型系统，若对设计方案心中没有把握时，可通过这种原型来证实设计方案的正确性。

- 演化型原型

主要用于及早向用户提交一个原型系统，该原型系统或者包含系统的框架，或者包含系统的主要功能，在得到用户的认可后，将原型系统不断扩充演变为最终的软件系统。

它将原型的思想扩展到软件开发的全过程。