# 软件工程相关概念

## 软件工程的基本概念

软件工程是指导计算机软件开发和维护的工程学科。采用工程的概念、原理、技术和方法来开发与维护软件，把经过**时间**考验而证明正确的管理技术和当前能够得到的最好的技术方法结合起来，这就是软件工程。

软件工程多数采用的是**生存周期**方法，就是从时间角度对软件开发和维护的复杂问题进行分解，把软件生存的漫长周期依次划分为若干个阶段，每个阶段有相对独立的任务，然后逐步完成每个阶段的任务。

## 二、软件工程的基本原理

著名软件工程专家B.Boehm综合有关专家和学者的意见并总结了多年来开发软件的经验，于1983年在一篇论文中提出了软件工程的七条基本原理。

（1）用分阶段的**生存周期**计划进行严格的管理。

（2）坚持进行**阶段评审**。

（3）实行严格的**产品控制**。

（4）采用**现代**程序设计**技术**。

（5）软件工程结果应能**清楚**地**审查**。

（6）开发小组的人员应该**少而精**。

（7）承认**不断改进**软件工程实践的必要性。

B.Boehm指出，遵循前六条基本原理，能够实现软件的工程化生产；按照第七条原理，不仅要积极主动地采纳新的软件技术，而且要注意不断总结经验。

## 三、软件工程遵循的原则

围绕工程设计、工程支持以及工程管理已提出了以下四条基本原则：

（1）选取**适宜**的开发**模型**

该原则与系统设计有关。在系统设计中，软件需求、硬件需求以及其它因素间是相互制约和影响的，经常需要权衡。因此，必需认识需求定义的易变性，采用适当的开发模型，保证软件产品满足用户的要求。

（2）采用**合适**的设计**方法**

在软件设计中，通常需要考虑软件的模块化、抽象与信息隐蔽、局部化、一致性以及适应性等特征。合适的设计方法有助于这些特征的实现，以达到软件工程的目标。

（3）提供高质量的工程**支撑**

在软件工程中，软件工具与环境对软件过程的支持颇为重要。软件工程项目的质量与开销直接取决于对软件工程所提供的支撑质量和效用。

（4）重视软件工程的**管理**

软件工程的管理直接影响可用资源的有效利用，生产满足目标的软件产品以及提高软件组织的生产能力等问题。因此，仅当软件过程予以有效管理时，才能实现有效的软件工程。

# 按照生存周期的顺序来介绍每个阶段的情况

## 问题定义

问题定义阶段必须回答的关键问题：“**要解决的问题是什么？**”

通过问题定义阶段的工作，系统分析员应该提出关于问题性质、工程目标和规模的书面报告。通过对系统的实际用户和使用部门负责人的访问调查，分析员扼要地写出他对问题的理解，并在用户和使用部门负责人的会议上认真讨论这份书面报告，澄清含糊不精的地方，改正理解不正确的地方，最后得出一份双方都满意的文档。

## 二、可行性的研究

可行性的研究要回答的关键问题：“**对于上一个阶段所确定的问题有行得通的解决办法吗？**”

可行性研究的结果是使用部门负责人做出是否继续进行这项工程的决定的重要依据，一般说来，只有投资可能取得较大效益的那些工程项目才值得继续进行下去。可行性研究以后的阶段将需要投入要多的人力物力。及时中止不值得投资的工程项目，可以避免更大的浪费。

在问题定义阶段提出的对工程目标和规模的报告通常比较含糊。可行性研究阶段应该导出系统的高层逻辑模型（通常用数据流图表示），并且在此基础上更准确、更具体地确定工程规模和目标。然后分析员更准确地估计系统的成本和效益，对建议的系统进行仔细的成本／效益分析是这个阶段的主要任务之一。**简单来说，就是将前一阶段的文件进一步的研究和细化**。

## 三、需求分析

该阶段要回答的问题：“**为了解决这个问题，目标系统必须做什么**”，主要是**确定目标系统必须具备哪些功能**。

用户了解他们所面对的问题，知道必须做什么，但是通常不能完整准确地表达出他们的要求，更不知道怎样利用计算机解决他们的问题；软件开发人员知道怎样使用软件实现人们的要求，但是对特定用户的具体要求并不完全清楚。因此系统分析员在需求分析阶段必须和用户密切配合，充分交流信息，以得出经过用户确认的系统逻辑模型。通常用数据流图、数据字典和简要的算法描述表示系统的逻辑模型。

然而，系统分析员通常都是计算机软件专家，技术专家一般都喜欢很快着手进行具体设计，然而，一旦分析员开始谈论程序设计的细节，就会脱离用户，使他们不能继续提出他们的要求和建议。较件工程使用的结构分析设计的方法为每个阶段都规定了特定的结束标准，需求分析阶段必须提供完整准确的系统逻辑模型，经过用户确认之后才能进入下一个阶段，这就可以有效地防止和克服急于着手进行具体设计的倾向。

## 四、总体设计

这个阶段必须回答的关键问题是：“**概括地说，应该如何解决这个问题？**”

首先，应该考虑几种可能的解决方案，例如：

（1）低成本的解决方案。系统只能完成最必要的工作，不能多做一点额外的工作。

（2）中等成本的解决方案。这样的系统不仅能够很好地完成预定的任务，使用起来很方便，而且可能还具有用户没有具体指定的某些功能和特点。虽然用户没有提出这些具体要求，但是系统分析员根据自己的知识和经验断定，这些附加的能力在实践中将证明是很有价值的。

（3）高成本的“十全十美”的系统。这样的系统具有用户可能希望有的所有功能和特点。

系统分析员应该使用系统流程图或其他工具描述每种可能的系统，估计每种方案的成本和效益，还应该在充分权衡各种方案的利弊的基础上,推荐一个较好的系统 (最佳方案),并且制定实现所推荐的系统的详细计划。如果用户接受分析员推荐的系统，则可以着手完成本阶段的其他工作。

## 五、详细设计

详细设计阶段的任务就是把解法具体化，也就是回答下面这个关键问题：“**应该怎样具体地实现这个系统呢？**”

这个阶段的任务还不是编写程序，而是设计出程序的详细规格说明。这种规格说明的作用很类似于其他工程领域中工程师经常使用的**工程蓝图**，它们应该包含必要的细节，程序员可以根据它们写出实际的程序代码。

## 六、编码和单元测试

这个阶段的关键任务是**写出正确的容易理解、容易维护的程序模块**。

程序员应该根据目标系统的性质和实际环境，选取一种适当的高级程序设计语言（必要时用汇编语言），把详细设计的结果翻译成用选定的语言书写的程序，并且仔细测试编写出的每一个模块。

## 七、综合测试

这个阶段的关键任务是**通过各种类型的测试（及相应的调试）使软件达到预定的要求**。

最基本的测试是集成测试和验收测试。必要时还可以再通过现场测试或平行运行等方法对目标系统进一步测试检验。

为了使用户能够积极参加验收测试，并且在系统投入生产性运行以后能够正确有效地使用这个系统，通常需要以正式的或非正式的方式对用户进行培训。

通过对软件测试结果的分析可以预测软件的可靠性；反之，根据对软件可靠性的要求也可以决定测试和调试过程什么时候可以结束。

应该用正式的文档资料把测试计划、详细测试方案以及实际测试结果保存下来，作为软件配置的一个组成成分。

## 八、软件维护

维护阶段的关键任务是，**通过各种必要的维护活动使系统持久地满足用户的需要。**

通常有四类维护活动：

1. 改正性维护，也就是诊断和改正在使用过程中发现的软件错误；
2. 适应性维护，即修改软件以适应环境的变化；
3. 完善性维护，即根据用户的要求改进或扩充软件使它更完善；
4. 预防性维护，即修改软件为将来的维护活动预先做准备。

每一项维护活动都应该经过提出维护要求（或报告问题），分析维护要求，提出维护要求，提出维护方案，审批维护方案，确定维护计划，修改软件设计，修改程序，测试程序，复查验收等一系列步骤，因此实质上是经历了一次压缩和简化了的软件定义和开发的全过程。

**参考资料**

<http://blog.csdn.net/u012555132/article/details/46430517>

<http://blog.csdn.net/fengzi_shen/article/details/2155349>