第二讲 软件开发过程模型

北京邮电大学网络空间安全学院 芦效峰

1

1

1. 软件生存周期

•软件产品从提出开始，到该软件产品被淘汰的全过程。

一个软件从定义、开发、使用、维护，直到最终被废弃，要经历一个漫长的时期，这就如同一个人要经过胎儿、儿童、青年、中年、老年，直到最终死亡的漫长时期一样，通常把软件经历的这个漫长时期称为**生存周期**。

2

2

软件生存期 life cycle

- 软件生存期的六个步骤，即**可行性研究**、**需求分析**、**设计**、**程序编码**、**测试**及**运行维护**
- 软件工程学的一个重要目标就是提高软件的可维护性，减少软件维护的代价。
- 在软件开发的不同阶段进行修改需要付出的代价很不相同.

3

3

软件生存周期

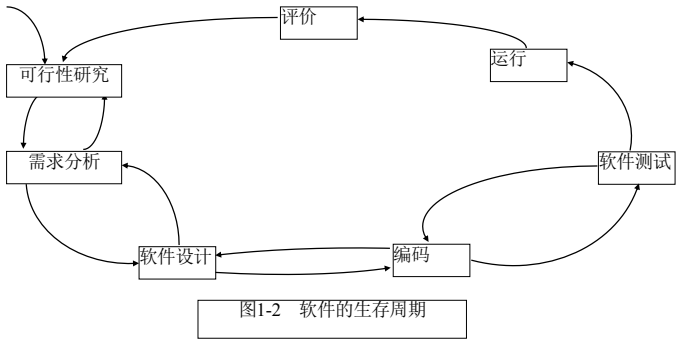


图1-2 软件的生存周期

4

4

可行性研究

- 确定要开发软件系统的总目标及可行性
- 给出功能、性能、可靠性以及接口等方面的要求
- 估计可利用的资源 (硬件，软件，人力等)、成本、效益、开发进度
- 制定出完成开发任务的实施计划，连同可行性研究报告，提交管理部门审查

5

5

需求分析和定义

- 对用户提出的要求进行分析并给出详细的定义
- 编写软件需求说明书或系统功能说明书及初步的系统用户手册
- 提交管理机构评审

6

6

软件设计

- 概要设计 – 把各项需求转换成软件的体系结构。结构中每一组成部分都是意义明确的模块，每个模块都和某些需求相对应
- 详细设计 – 对每个模块要完成的工作进行具体的描述，为源程序编写打下基础
- 编写设计说明书，提交评审。

7

7

程序编写

- 把软件设计转换成计算机可以接受的程序代码，即写成以某一种特定程序设计语言表示的“源程序清单”
- 写出的程序应当是结构良好、清晰易读的，且与设计相一致的
- 大部分计算机课程在这一层

8

8

软件测试

- **单元测试**，查找各模块内部在功能和结构上存在的问题并加以纠正
- **组装测试**，将已测试过的模块按一定顺序组装起来
- 按规定的各项需求，逐项进行**有效性测试**，决定已开发的软件是否合格，能否交付用户使用

9

9

运行 / 维护

- **改正性维护** 运行中发现了软件中的错误需要修正
- **适应性维护** 为了适应变化了的软件工作环境，需做适当变更
- **完善性维护** 为了增强软件的功能需做变更

10

10

阶段	关键问题	结束标准
问题定义	问题是什么	关于规模和目标的报告书
可行性研究	有可行的解吗	系统高层逻辑模型：数据流图，成本/效益分析
需求分析	系统必须做什么	系统的逻辑模型：数据流图，数据字典，算法描述
总体设计	概括的说，系统应该如何解决这个问题	可能的解法：系统流程图，成本/效益分析，推荐的系统结构：层次图或结构图

11

11

阶段	关键问题	结束标准
详细设计	怎样具体地实现这个系统	编写规格说明：HIPO图或PDL
编码和单元测试	正确的程序模块	源程序清单，单元测试方案和结果
综合测试	符合要求的软件	综合测试方案和结果，完整一致的软件配置
维护	持久地满足用户需要的软件	完整准确的维护记录

12

12

2 常用软件开发过程模型

2.1瀑布模型：将各项活动规定为依固定顺序连接的若干阶段工作，从上到下，不可逆转

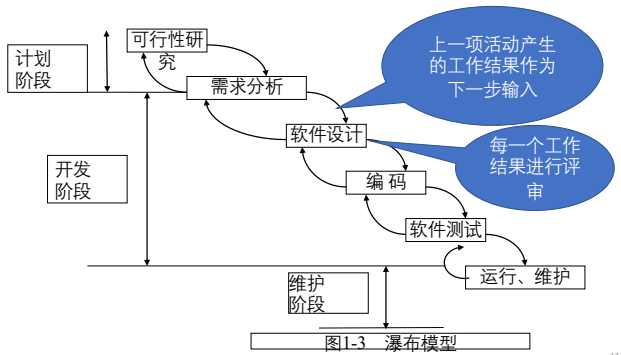


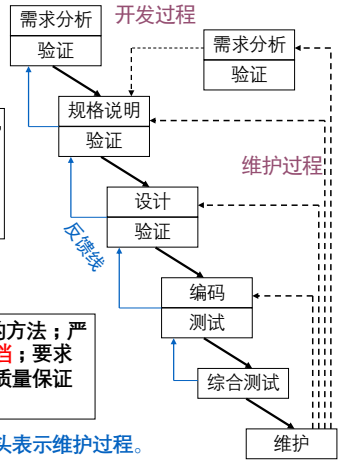
图1-3 瀑布模型

13

13

瀑布模型特点

按照传统的瀑布模型来开发软件，有以下特点：
1、阶段间有顺序性和依赖性；
2、推迟实现的观点；
3、质量保证的观点；



优点：可强迫开发人员采用规范的方法；严格规定了每个阶段必须提交的文档；要求每个阶段交出的产品都必须经过质量保证小组的仔细验证。

实线箭头表示开发过程；虚线箭头表示维护过程。

14

14

瀑布模型缺点

- 项目开始阶段，开发人员和用户对需求的描述常常不全面。如果需求阶段没发现问题，则影响后面各阶段的工作
- 各阶段所做的工作都是文档说明，一般用户不易理解文字所叙述的软件。用户的修改意见加大了修改难度
- 改变会影响整个软件开发。事先选择的技术或需求迅速发生变化，需要返回到前面，对前面一系列内容进行修改

15

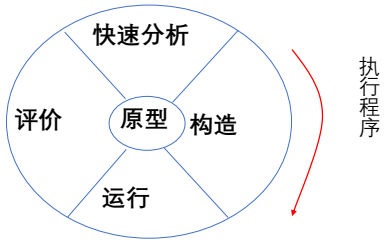
15

2.2 快速原型模型

- **由于在项目开发的初始阶段人们对软件的需求认识常常不够清晰，因而使得开发项目难于做到一次开发成功，出现返工再开发在所难免。**
- **快速原型模型的第一步是建造一个快速原型，实现客户或未来的用户与系统的交互，用户或客户对原型进行评价，进一步细化待开发软件的需求。通过逐步调整原型使其满足客户的要求，开发人员可以确定客户的真正需求是什么；第二步则**在第一步的基础上开发客户满意的软件产品。

16

• 模型原理图



17

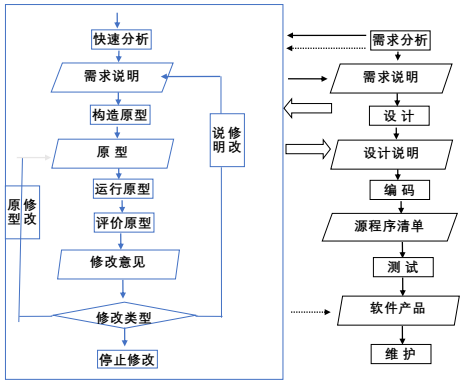
17

- **做两次或多次**
- **第一次只是试验开发，其目标只是在于探索可行性，弄清软件需求**
- **第二次则在此基础上获得较为满意的软件产品**
- **显然，快速原型方法可以克服瀑布模型的缺点，减少由于软件需求不明确带来的开发风险，具有显著的效果。**

18

18

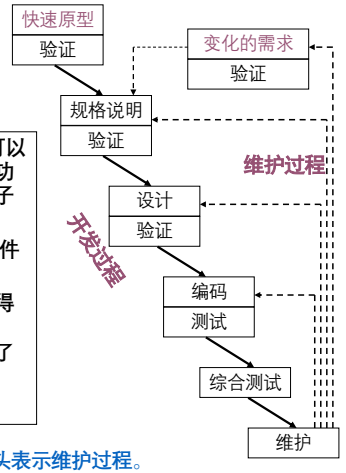
• 过程图



19

19

快速原型模型特点



快速原型是快速建立起来的可以在计算机上运行的程序，完成的功能是最终产品能完成功能的一个子集。

1. 快速原型模型不带反馈环，软件开发基本上是线形顺序进行的。
2. 原型系统已经通过与用户交互得到验证；
3 。 开发人员在建立原型中学到了许多东西；

实线箭头表示开发过程；虚线箭头表示维护过程。

20

20

特点

- 原型模型的最大特点是：利用原型法技术能够快速实现系统的初步模型，供开发人员和用户进行交流，以便较准确获得用户的需求，采用逐步求精方法使原型逐步完善，是一种在新的高层次上不断反复推进的过程。
- 它可以大大避免在瀑布模型冗长的开发过程中看不见产品雏形的现象。

21

模型的关键

- 快速原型的关键在于尽可能快速地建造出软件原型，一旦确定了客户的真正需求，所建造的原型将被丢弃。因此，原型系统的内部结构并不重要，重要的是必须迅速建立原型，随之迅速修改原型，以反映客户的需求。

22

优点

- 克服瀑布模型的缺点，减少由于软件需求不明确带来的开发风险。
- 一是开发工具先进，开发效率高，使总的开发费用降低，时间缩短；
- 二是开发人员与用户交流直观，可以澄清模糊需求，调动用户的积极参与，能及早暴露系统实施后潜在的一些问题；
- 三是原型系统可作为培训环境，有利于用户培训和开发同步，开发过程也是学习过程。

23

原型模型的缺陷

- 1） 建立原型模型的软件工具与环境与实际模型的存在脱节的现象。
  2） 以目前通用的开发工具，开发原型本身就不是件容易的事情。
  3） 原型模型对用户深层次的需求并不能深入分析。
- 4）快速建立起来的系统结构加上连续的修改可能会导致产品质量低下；

24

缺点

- 产品原型在一定程度上限制了开发人员的创新，没有考虑软件的整体质量和长期的可维护性。
- 由于达不到质量要求产品可能被抛弃，而采用新的模型重新设计，因此原型实现模型不适合嵌入式、实时控制及科学数值计算等大型软件系统的开发；

25

2. 3螺旋模型

1988年，Barry提出螺旋模型，将瀑布模型和快速原型结合，注重风险分析，适合于大型复杂系统。

软件开发的风险：

1、 产品交付用户之后用户可能不满意；
2、 到交付日期后软件可能还未开发出来；
3、 实际开发成本可能超过预算；
4、 一些关键的开发人员可能“跳槽”；
5、 聘请不到需要的专业人员；

基本思想：使用原型及其他方法来尽量降低风险。可看作在每个阶段之前都增加了风险分析过程的快速原型模型。螺旋线每个周期对应一个开发阶段。

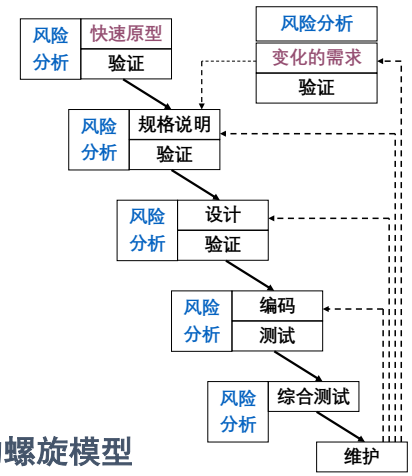
26

26

螺旋模型的基本思想：使用原型及其他方法来尽量降低风险。

简化的螺旋模型可看作在每个阶段都增加了风险分析过程的快速原型模型。

风险分析是对风险进行识别，分析，采取对策，进而消除或减少风险的损害。



简化的螺旋模型

27

27

螺旋模型

- 螺旋模型沿着螺线旋转，在四个象限上分别表达四个方面的活动，即：
- 制定计划—确定软件目标，选定实施方案，弄清项目开发的限制
- 风险分析—分析所选方案，考虑如何识别和消除风险
- 实施工程—实施软件开发
- 客户评估—评价开发工作，提出修正建议

28

28

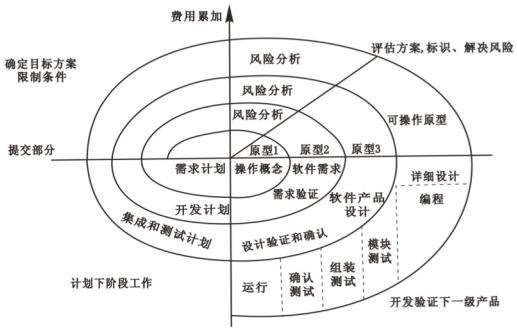
每个阶段开始时（左上象限）的任务是：确定该项目的目标、为完成这些目标选择方案及设定这些方案的约束条件。接下来的任务是：从风险角度分析上步的工作结果，努力排除各种潜在的风险，如果风险不能排除，则停止开发工作或大幅度削减项目规模。如果成功排除了所有风险，则启动下一个开发步骤（右下象限），这个步骤相当于纯粹的瀑布模型。最后是评价该阶段工作成果并计划下一阶段的工作。

29

29

螺旋模型

沿着螺旋线每转一圈，表示开发一个更完善的版本。



30

30

- “螺旋模型”的核心就在于不需要在刚开始的时候就把所有事情都定义的清清楚楚。轻松上阵，定义最重要的功能，实现它，然后听取客户的意见，之后再进入到下一个阶段。如此不断轮回重复，直到得到满意的最终产品。

31

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，也有利于把软件质量作为软件开发的一个重要目标；

2、减少了过多的测试或测试不足所带来的风险；

3、维护只是模型的另一个周期，在维护和开发之间没有本质的区别。

32

32

螺旋模型的缺陷

- 1）采用螺旋模型需要具有相当丰富的风险评估经验和专门知识，在风险较大的项目开发中，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。

33

螺旋模型限制条件

- 螺旋模型强调风险分析，但要求客户接受这种风险，并做出相关反映是不容易的。
- 如果执行风险分析会大大影响项目的利润，那么风险分析就没有意义了。

只适合大规模软件开发

- 软件开发人员应该擅长寻找可能的风险，准确地分析风险，否则将会带来更大的风险。

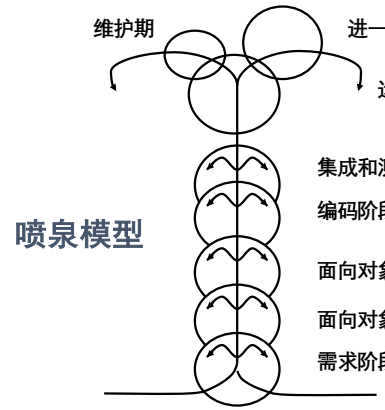
34

34

2.4 喷泉模型

- 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周期的各阶段是相互重叠和多次反复的，就像水喷上去又可以落下来，类似一个喷泉。
- 各个开发阶段没有特定的次序要求，并且可以交互进行，可以在某个开发阶段中随时补充其他任何开发阶段中的遗漏。

35



喷泉模型

各个开发阶段没有特定的次序要求，可以随时补充其他任何开发阶段中的遗漏。

36

- 喷泉模型主要用于面向对象的软件项目，软件的某个部分通常被重复多次，相关对象在每次迭代中随之加入渐进的软件成分。各活动之间无明显边界，例如设计和实现之间没有明显的边界，这也称为“喷泉模型的无间隙性”。
- 由于对象概念的引入，表达分析、设计及实现等活动只用对象类和关系，从而可以较容易地实现活动的迭代和无间隙。

37

- 喷泉模型不像瀑布模型那样，需要分析活动结束后才开始设计活动，设计活动结束后才开始编码活动。该模型的各个阶段没有明显的界限，开发人员可以同步进行开发。

38

优点

- 优点是可以提高软件项目开发效率，节省开发时间，适应于面向对象的软件开发过程。
- 可以从任何一个开发阶段（泡泡）转到其它任一个开发阶段，各个阶段之间没有明显的界限。也就是，在整个过程中补漏、拾遗、纠错的切入点大大增多，不受开发阶段的限制。

39

缺点

- 由于喷泉模型在各个开发阶段是重叠的，因此在开发过程中需要大量的开发人员，因此不利于项目的管理。
- 此外这种模型要求严格管理文档，使得审核的难度加大，尤其是面对可能随时加入各种信息、需求与资料的情况。
-

40

2.5 组件集成模型

- 我们将一个或多个相关类的组合称为一个组件或构件。
- 组件集成模型融合了螺旋模型的许多特征，本质上是演化型的，开发过程是迭代的
- 以前项目中创建的组件被存储在一个组件库中。一旦开发人员经过与用户交谈，确定软件目标和方案，就可以标识出所需组件。首先搜索已有的组件库，如果需要的组件已存在，就从库中提取出来复用。如果不存在，就采用面向对象方法开发它。

41

41

2.6 敏捷开发过程

- 敏捷开发(Agile Development)是一种以人为核心、迭代、循序渐进的开发方法。
- 敏捷不是一个过程，是一类过程的统称，它们有一个共性，就是符合敏捷价值观，遵循敏捷的原则。
- 一种软件开发的流程，它会指导我们用规定的环节去一步一步完成项目的开发；而这种开发方式的主要驱动核心是人；它采用的是迭代式开发；

42

42

为什么说是以人为核心？

- 我们大部分人都学过瀑布开发模型，它是以文档为驱动的。因为在瀑布的整个开发过程中，要写大量的文档，把需求文档写出来后，开发人员都是根据文档进行开发的，一切以文档为依据；
- 而敏捷开发它只写有必要的文档，或尽量少写文档，敏捷开发注重的是人与人之间，面对面的交流，所以它强调以人为核心。

43

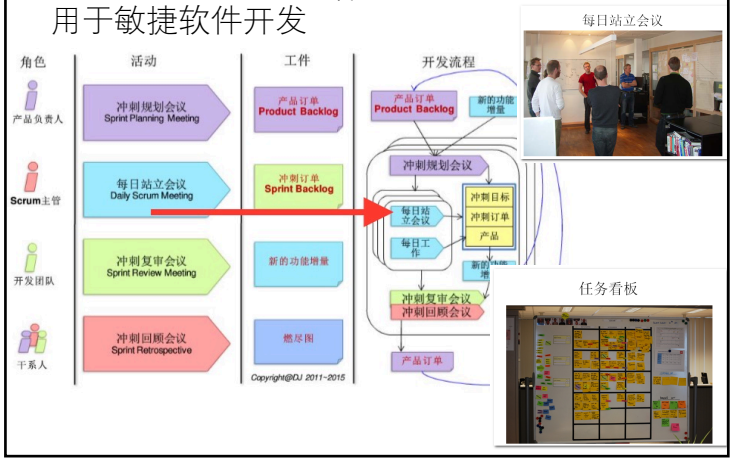
43

- 迭代是指把一个复杂且开发周期很长的开发任务，分解为很多小周期可完成的任务，这样的一个周期就是一次迭代的过程；同时每一次迭代都可以生产或开发出一个可以交付的软件产品。

44

44

Scrum是一种迭代式增量软件开发过程，通常用于敏捷软件开发



45

2.7 微软开发方法

- 《微软的秘密》对微软公司软件产品开发过程进行了介绍。
- 三个阶段：计划阶段、开发阶段、稳定化阶段

（1）计划阶段

- 产生：想象性描述、产品说明，接口标准、最初的测试计划、文档策划、市场营销
- 想象性描述主要定义产品开发目标，不涉及具体细节。

46

46

- 想象性描述包括对竞争对手的产品分析、未来版本的规划、必须解决的问题，新功能
- 产品说明：定义出新的产品特性，并赋予不同优先级，及特性之间的关系。随着项目深入，不断添加细节
- 对重要产品的分析和设计要高层领导复审
- 基于产品说明，管理部门指定进度表，安排开发组。

47

47

（2）开发阶段

- 首先做计划，定里程碑版本，产品分成3、4个子项目，测试人员与开发人员配对，不断测试
- 基于产品说明开发。
- 任务细化到4小时~3天
- 预留1/3的缓冲时间

（3）稳定化阶段

- 着重对产品测试和调试，不再增加新的功能
- 内部测试和外部测试

48

48

3 软件开发方法

- 结构化方法自1968年被提出以来经过了多年的发展，形成了一套完整的体系。构成结构化开发方法。
- 面向对象开发方法

49

49

结构化开发基本原则

- 自顶向下

  程序设计时，应先考虑总体，后考虑细节；先考虑全局目标，后考虑局部目标。不要一开始就过多追求众多的细节，先从最上层总目标开始设计，逐步使问题具体化。
- 逐步细化

  对复杂问题，应设计一些子目标作为过渡，逐步细化。
- 模块化设计
- 限制使用goto语句，3种程序结构

50

50

结构化的不足

- 随着软件产品剧模的不断增大，结构化范型有时不能应付。也就是说，结构化技术处理5000行 或50000行代码是有效的。然而.当今的软件产品，有50000行或5000000甚至更多行代码的产品非常普遍。维护阶段是结构化范型不能满足人们期望的第二个方面。

51

51

面向对象开发方法

- 面向对象技术自20世纪90年代提出以来得到快速发展，并被运用到各种各样的软件应用开发中。
- 面向对象技术将数据和数据上的操作封装在一起。对外封闭这些细节。从而实现了信息隐藏的目的。
- 类、对象、封装、

52

52

二者区别

- 面向对象程序设计可以看作一种在程序中包含各种独立而又互相调用的对象的思想。这与传统的思想刚好相反：传统的程序设计主张将程序看作一系列函数的集合，或者直接就是一系列对电脑下达的指令。面向对象程序设计中的每一个对象都应该能够接受数据、处理数据并将数据传达给其它对象，因此它们都可以被看作一个小型的“机器”，即对象。

53

53

1.5 软件文档

1.5.1软件文档在软件开发中的地位和作用

1．软件文档在软件开发中的地位

- 软件开发是一个系统工程，从软件的生存周期角度出发，科学的编制软件文档很有必要。
- 软件文档也称文件，通常指的是一些记录的数据和数据媒体，可被人和计算机阅读。
- 在软件工程中，文档常常用来表示对活动、需求、过程或结果进行描述、定义、规定、报告或认证的任何书面或图示的信息，他们描述和规定了软件设计和实现的细节，说明使用软件的操作命令。

54

54

4、软件文档

4.1 文档的作用

(1) 提高软件开发过程的能见度。检查软件开发进度和开发质量的依据，实现对软件开发的工程管理。

(2) 提高开发效率。使得开发人员对各个阶段的工作都进行周密思考、并且可及早发现错误，便于及时加以纠正。

(3) 作为开发人员在一定阶段的工作成果和结束标志。

(4) 记录开发过程中有关信息，便于协调以后的软件开发、使用和维护。

(5) 提供对软件的运行、维护和培训的有关信息，便于管理人员、开发人员、操作人员、用户之间协作、交流和了解，使软件开发活动更科学有效。

(6) 便于潜在用户了解软件的功能、性能等各项指标，为选购符合自己需要的软件提供依据。

55

55

4.2 软件文档的种类及写作要求

1．软件文档的种类

(1) 根据形式，软件文档可以分为两类：

1) 工作表格，包括开发过程中填写的各种图表；
2) 文档或文件，包括应编制的技术资料或技术管理资料。

56

56

(2) 按照文档产生和使用的范围

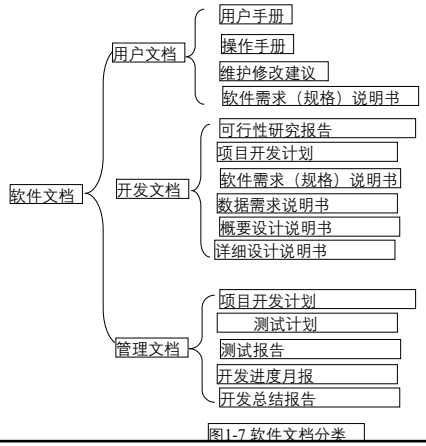


图1-7 软件文档分类

57

57

4.3 软件文档的种类及写作要求

1. 软件文档的写作要求

(1) 针对性，分清读者。
(2) 精确性，不出现多义性。
(3) 清晰性，力求确切，配以图表。
(4) 完整性，自成体系。
(5) 灵活性，因项目不同而有差别。
(6) 可追溯性。

58

58

公共邮箱

- buptpowerpoint@126.com
- powerpoint

59

59