**软件工程**

第一章 软件工程学概述

1.软件危机的主要表现

(1)对软件开发成本和进度的估计不准确

(2)用户对已完成的软件不满意现象京城发生

(3)软件产品的质量往往不可靠

(4)软件常常不可维护

(5)软件同城没有适当的文档资料

(6)软件成本在计算机系统总成本中的比例逐年上升

(7)软件生产率的加速度跟不上计算机应用迅速普及和深入的趋势

2.产生软件危机的原因

(1)用户需求不明确(

在1软件开发出来之前，用户自己也不清楚软件开发的具体需求；

用2户对软件开发需求的描述不精确，可能有遗漏、有二义性、甚至有错误；

在软件开发过程中，用户还提出修改软件开发功能、界面、支撑环境等方面的要求；

软件开发人员对用户需求的理解与用户本来愿望有差异。

)

(2)缺乏正确的理论指导

(3)软件开发规模越来越大

(4)软件开发复杂度越来越高

3.什么是软件工程

(1)软件工程是知道计算机软件开发和维护的一门工程科学

采用工程学的概念、原理、技术和方法开发和维护软件。

把经验的管理技术和当前能得到的最好的技术方法结合起来。

经济的开发高质量软件并维护他，之就是软件工程

4.软件工程的基本原理

(1)用分阶段的生命周期计划严格管理

(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)结果应能清楚的审查

(6)开发小组的人员少而精

(7)承认不断改进软件工程实践的必要性

5.软件工程方法学

(1)传统方法学：生命周期方法学/结构化方法学：采用结构化技术

(2)面向对象方法学：1.将对象作为融合了数据以及对这些数据操作的统一的软件结构

2.对象划分为类3

3.父类子类构成结构层次系统

4.对象之间只能通过发送消息联系

基本原则：尽量模拟人类习惯的思维方式

6.软件生命周期(三个时期八个阶段)

三个时期：软件定义，软件开发，运行维护

八个阶段：

(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，编码和单元测试，综合测试

(运行维护)软件维护

7.瀑布模型

(1)阶段间顺序性和依赖性

(2)推迟实现的观点点

(3)质量保证观点：每个阶段有文档，对文档评审

文档驱动模型

第二章 可行性研究

1.目的：最小的代价在尽可能短的的时间内确定问题能否解决，不是解决问题，而是是否值得去解决

3个方面：技术、经济、操作可行性

2.可行性研究过程

(1)复查系统规模和目标

(2)研究目前正在使用的系统

(3)导出新系统的高层逻辑模型

(4)进一步定义问题

(5)导出和评价供选择的解法

(6)推荐行动方针

(7)草拟开发计划

(8)书写文档提交审查

3.数据流图

4.数据字典

组成：数据流、数据流分量（数据元素）、数据存储、处理

5.数据字典的实现

6.成本效益分析方法

(1)货币的时间价值

(2)投资回收期

(3)纯收入

(4)投资回收率

第三章 需求分析

1.需求分析任务

需求分析是软件定义时期的最后一个阶段，基本任务回答”系统必须做什么“

对目标系统提出完整准确清晰具体的要求

（1）.确定对系统的综合要求

-1功能需求

-2性能需求

-3可靠性和可用性需求

-4出错处理需求

-5接口需求

-6约束

-7逆向需求

-8将来可能提出的需求

（2）.分析数据要求

（3）.导出逻辑模型

（4）.修正开发计划

2.实体联系图（ER）

(1)数据对象

(2)属性

(3)联系

第四章

第五章 总体设计

1.模块化定义：把程序划分成独立命名、独立访问的模块

每个模块完成一个子功能。模块集合起来可以构成一个完成指定功能的整体

2.模块独立：

(1)耦合：不同模块之间互联程度的度量，要小

尽量少使用数据耦合，少用控制偶合和特征耦合，限制公共环境耦合的范围，完全不用内容耦合

数据耦合：

控制耦合：

特征耦合:

环境耦合：

内容耦合：

(2)内聚：同意模块各元素结合的紧密程度

功能内聚

顺序内聚

通信

过程

时间

逻辑

偶然

第六章 详细设计

1.人机界面设计指南

一般交互指南

(1)保持一致性

(2)提供有意义的反馈

(3)执行破坏性大的动作前征求用户确认

(4)允许取消多数操作

(5)减少两次操作间要记忆的信息量

(6)提高对话移动和思考的效率

(7)允许犯错误

(8)按功能动作分类

(9)提供对用户工作内容敏感的帮助设施

(10)用简单动词或动词短语做命令名

信息显示指南

(1)只显示与当前工作有关的信息

(2)以便于理解的方式表示数据

(3)用一致的标记、标准缩写、可预知的颜色

(4)允许用户保持可视化的语境

(5)产生有意义的出错信息

(6)使用大小写，缩进，文本分组帮助理解

(7)用窗口分隔不同类型的信息

(8)使用模拟显示的方式表示信息

(9)高效使用显示屏

数据输入指南

(1)减少用户输入动作

(2)保持信息显示和数据输入之间的一致性

(3)允许用户自定义输入

(4)交互应该是灵活的，可调成用户最喜欢的输入方式

(5)是在当前语境中不适用的命令不起作用

(6)让用户控制交互流

(7)对所有的输入动作都提供帮助

(8)消除冗余输入

第七章 实现

1.选择程序设计语言：

2.编码风格

（1）程序内部的文档

（2）数据说明

（3）语句构造

（4）输入输出

（5）效率

3。软件测试

3.1目标：

(1)测试是为了发现程序中的错误而执行的过程

(2)好的此时极有可能发现迄今为止微发现的错误

(3)成功的测试是发现至今为止尚未发现的错误的测试

3.2准则

(1)所有测试应该能追溯到客户需求

(2)在测试开始之前制定测试计划

(3)把Pareto原理（80%的错误由20%模块造成）

(4)测试规模由小到大

(5)穷举测试不可行

(6)独立的第三方从事测试工作能达到最佳测试效果

3.3方法

(1)黑盒测试

知道功能，测试功能能否正常使用，功能测试

(2)白盒测试

知道内部结构与处理算法，结构测试

3.4步骤

(1)模块测试

(2)子系统测试

(3)系统测试

(4)验收测试

(5)平行运行

3.5单元测试

(1)概念、特征

单元测试机中检测软件最小单元——模块

方法：人工测试，计算机测试：相互补充，各有所长

主要使用白盒测试技术，可多模块并行测试

(2)测试重点：

-1 模块接口

-2 局部数据结构

-3 重要的执行通路

-4 出错处理通路

-5 边界条件

3.6集成测试:用来测试和组装软件的系统化技术

(1)自顶向下集成

优点：不需要测试驱动程序，能在早期实现并验证系统的主要功能，能在早期发现上层接口的错误

缺点：要存根程序，低层模块错误发现较晚，早期不能充分展开人力

(2)自底向上集成

优缺点与上面相反

3.7确认测试

Alpha测试，在开发者指导的受控环境中的测试

Beta测试，用户在客户场所进行的测试

3.8白盒测试技术（结构）

(1)逻辑覆盖

-1语句覆盖

-2判定覆盖

-3条件覆盖

-4判定/条件覆盖

-5条件罪和覆盖

-6点覆盖

-7边覆盖

-8路径覆盖

(2)控制结构

-1基本路径测试

-2条件测试

-3循环测试

3.9黑盒测试技术（功能）

(1)等价划分

(2)边界值分析

(3)错误推测：靠直觉、经验

白盒测试在测试的早期阶段进行，黑盒测试在测试的后期阶段测试。两者互补。

第八章 维护

1.软件维护定义

软件维护是在软件已经交付使用后，未改正错误或满足新的需要而修改软件的过程。分为4项：改正性维护、适用性维护、完善性维护（占大部分）、预防性维护

2.软件维护特点

-1结构化维护与非结构化维护差别巨大（软件的配置完不完整）

-2维护代价高昂

-3维护的问题很多

3软件可维护性

(1)决定可维护性的因素

-1可理解性

-2可测试性

-3可修改性

-4可以执行

-5可重用性

(2)文档：影响软件可维护性的决定性因素，比还代码重要

-1用户文档

-2系统文档

(3)可维护性复审

从容易修改、模块化、功能独立的目标出发，评价软件的结构和过程，为可能修改的部分做准备

第十三章 软件项目管理

1.人员组织

(1)民主制程序员组

-1主要特点：小组成员完全平等，充分民主，协商决策

-2成员：

-3关系:相互平等

-4优点：程序员对发现错误持积极态度，快速发现错误，生成高质量代码。凝聚力高，学术空气浓厚，有利于攻克技术难关

-5缺点：如果程序员水平不高，容易缺乏必要合作，导致工程失败

(2)主程序员组

-1主要特点：专业化、层次化

-2成员：编程秘书、主程序员、后备程序员、程序员。

-3关系：主程序员指导程序员，编程秘书负责事务性工作，后备程序员协助工作

-4缺点：不利于找错误，代码质量不高

(3)现代程序员组

-1主要特点：专业化、层次化

-2成员：技术组长，行政组长，程序员。

-3关系：技术组长负责技术，行政组长负责行政

-4优点：充分发挥每个程序员的积极性和主动性，集思广益攻克技术难题。于集中指导下发扬民主

2.质量保证

软件质量因素的定义：

-1正确性

-2健壮性

-3效率

-4完整性：安全性

-5可用性

-6风险

-7可理解性

-8可维修性

-9灵活性：适应性

-10可测试性

-11可移植性

-12可再用性

-13互运行性