Chapter 1

1. 什么是软件：

指令：使计算机提供功能和性能

数据结构：使程序能够操纵信息

文档：记录程序的产生，功能与使用

1. 软件的特征：
2. 最根本的：软件是一种逻辑元素而不是物理元素
3. 软件是开发出来的，而不是用传统的方法制造出来的
4. 软件不会被用坏
5. 工业界已经走向了标准化装配时代，然而绝大多数软件还是定制出来的。

3、项目是什么

没有例行的任务

需要计划

特定的目标需要满足或者特定的产品需要生成

项目有一个预定义的时间范围

工作不仅仅是为自己，也是为他人

工作中有些特性

工作分为若干阶段

项目完成需要资源

项目是大型的或者复杂的

* 什么是项目管理，项目管理的特征，涉及的知识域

项目管理是在项目活动中应用知识，技能，工具和技术来满足项目需求的过程，它通过初始化，计划，执行，控制和结束等活动来完成。

特征：普遍性／目的性／独特性／集成性／创新性

涉及的知识域：项目整体管理；项目范围管理；质量管理；时间管理；成本管理；人力资源管理；沟通管理；项目采购管理；项目风险管理。

* 项目阶段，项目生命周期

每一个项目阶段的标记是一个或几个可交付的物件（deliverable）。Deliverable是一个具体的可验证的工作产品，如可行性研究，详细设计或者一个工作原型。

初始化，计划，执行，控制，结束。

项目生命周期：

项目生命周期定义了一个项目的开始与结束。

项目生命周期定义的阶段顺序通常包括某些技术转移或“握手”(hand off)，如从需求到设计，从构造到运行，但是在风险允许下，也可以下一阶段提前进行，这种重叠的阶段被称为快速跟踪(fast tracking)。

项目生命周期通常定义：

* + 各个阶段需要完成的技术工作；
  + 每个阶段需要涉及的人。

绝大多数项目生命周期有一些共同的特点

项目生命周期与产品生命周期是不同的。

* 软件项目及其特点，软件项目的阶段

软件项目特征：

不可见

复杂性（以每一单位货币来看）

灵活性：软件去适应人或组织而不是相反

软件项目的阶段

需求分析

描述

设计

编码

校验

安装

维护

支持

* 软件项目的目标与人员

软件项目的目标：

目标：项目的目标必须清晰定义，以确定成功的关键。每一个项目必须有一个项目权威（authority）,该权威经常被称为项目指导委员会(steering committee),项目的目标必须由指导委员会确定。

效率的测量：有效的目标必须易于测量，从而更能够衡量是否成功。。

子目标

软件项目的人员：项目影响者->项目小组内部；项目小组外部，但是在同一组织内；项目小组和组织外部；如客户

可能遗忘的人员是：当地政府相关人员，因为如果得到他们的帮助可能有助于新系统的开发，甚至有可能直接招聘他

* 常见错误

产品相关的错误

* + 需求镀金：项目具有比实际需求多得多的性能
  + 功能蔓延：项目平均会有25%的需求变更（Jones 1994）
  + 开发人员的镀金：开发人员着迷于新技术
  + 又推又拉的交易：经理在批准项目进度顺延时又加入了新的功能
  + 研究导向的开发
* 过程
  + 缺乏计划
  + 过于乐观的计划
  + 在压力下放弃计划
  + 缺乏足够的风险管理
  + 承包人导致的失败
  + 在模糊的项目前期（fuzzy frond end）浪费时间
  + 前期活动不合要求
  + 设计低劣
  + 缺少质量保证措施
  + 缺少管理控制
  + 太早和过于频繁的集成
  + 项目估算时遗漏必要的任务
  + 追赶计划
  + 鲁莽编码
* 技术相关的错误
  + 银弹综合症: 过于相信以前没有采用过的技术的宣传
  + 过高估计了新技术或方法带来的节省量
  + 项目中间切换工具
  + 缺少自动的源代码控制手段
* 人员相关的错误
  + 挫伤积极性
  + 人员素质低
  + 对有问题的员工失控
  + 英雄主义
  + 项目后期加入人员：“火上加油”
  + 办公环境差
  + 开发人员与客户之间发生摩擦
  + 不现实的预期
  + 缺乏有效的高层对项目的支持
  + 缺乏各种角色的齐心协力
  + 缺乏用户介入
  + 政治高于物质
  + 充满想像：“项目组没人真正相信他们能够按给定的计划进度完成项目，但他们认为如果每个人能够努力工作，并且不出现问题，他们可能会很幸运地按时完成任务。

2/项目计划总览

* 项目计划的作用
  + 建立项目目标
  + 分析项目特点
  + 建立包含组织、标准、方法和工具在内的基础架构
  + 确定项目的产品和所需的活动
  + 分配资源
  + 建立质量控制手段
* 项目计划模型
  + Step Wise Project Planning
  + PRINCE 2 【 Central Computing and Telecommunications Agency(CCTA) for use on British government IT projects.】（荷兰和澳大利亚也采用）
* PBS,PFD及活动图

PBS：Project Breakdown Structure 产品分解结构

* PBS的描述方法
  + - 产品的名字/标识
    - 产品的功能
    - 产品的派生物（从中派生的其它产品）
    - 产品的分解结构
    - 产品的形式
    - 相关标准
    - 适用的质量标准

PFD：Product Flow Diagram 产品流程图

3/项目评估

* 项目评估原则，内容，与可行性分析的差别
* 原则
* 投入与产出相匹配的原则
* 面向需求的原则
* 资金时间价值原则
* 公正性原则
* 科学性原则
* 客观性原则
* 内容
* 策略评估
* 操作性评估
* 计划评估
* 技术评估
* 风险评估
* 社会可行性（法律，合同，政治……）
* 经济性评估－重点
* 项目评估与可行性分析的差别
* 承担两项工作的主体不同：一般需要由不同的机构分担
* 视角和着重点不同：可行性研究注重从企业角度评价项目，项目评估注重于项目的宏观效益
* 为项目决策服务的任务和目的不同
* 可行性研究为复杂的技术经济性论证工作，需规划多种方案
* 项目评估一般借助于可行性研究的成果进行系统的审查和核实
* 可行性研究与项目评估在项目决策过程中的时序和地位也有差别：可行性研究是项目评估的前提，项目评估为决策提供直接的、最终的依据
* 评估内容中经济评估是重点

经济分析在ppt中，内容有点多。。。

* 综合评估

4/需求管理

* 目标管理的概念和特点
  + 概念（概念中感觉已经有了特点了）
    - *目标管理是参与管理的一种形式：*上下级目标形成“目标－手段”链
    - *强调“自我控制”：*人们应“控制”的是行为的动机而不是行为本身
    - *促进下放权力：*有助于协调集权与分权的矛盾
    - *注重成果第一的方针：*目标考核体系
* 项目目标的特点

多目标性，优化性，层次性

* 范围管理

项目范围是指为了成功达到项目的目标，项目所规定要做的工作；

在项目环境中，范围包括：

* + - 产品范围，即一个产品或一项服务应该包含哪些特征和功能
    - 产品规格，即产品所包含的特征和功能具体是怎样的
    - 项目范围，即为了交付具有所指特征和功能的产品所必须要做的工作

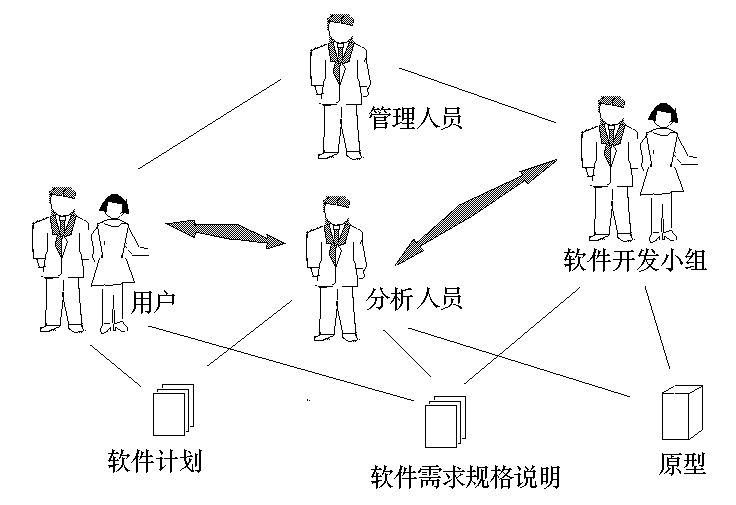
范围管理的过程：启动－范围计划－范围定义－范围核实－范围变更控制

* 需求管理
  + 需求工程

需求工程的概念



需求工程涉及的人员



* + 如何获取需求

需求的来源

* 访问并与有潜力的用户探讨
* 把对目前的或竞争产品的描述写成文档
* 系统需求规格说明
* 对当前系统的问题报告和增强要求
* 市场调查和用户问卷调查
* 观察正在工作的用户
* 用户任务内容分析
  + 需求规格说明
    - 作用
      * 客户和营销部门依赖它了解他们所能提供的产品
      * 项目经理根据包含在软件需求规格说明重描述的产品来制定规划并预测进度安排、工作量和资源
      * 软件开发小组依赖它来理解他们将要开发的产品
      * 测试小组利用它来制定测试计划，测试案例
      * 软件维护人员和支持人员依据它了解系统的功能
      * 产品发布组根据它编写客户文档，包括用户手册和帮助
      * 培训人员根据它编写培训教材
    - 注意点
      * 文档可读性
      * 需求的标识
      * 不完整需求（TBD to be determined）
      * 用户界面与需求说明

1. 项目方法选择
2. 识别项目中的高风险

* 产品不确定性
* 过程不确定性
* 资源不确定性

1. 各种模型的特点和优缺点比较

* 编码修正模型
  + 优点
    - 成本可能很低
    - 只需要很少的专业知识，任何写过程序的人都可以
    - 对于一些非常小的、开发完后就会很快丢弃的软件可以采用
  + 缺点
    - 对于规模稍大的项目，采用这种模型是很危险的
* 瀑布模型
  + 优点
* 当有一个稳定的产品定义和很容易被理解的技术解决方案时，纯瀑布模型特别合适
* 当你对一个定义得很好的版本进行维护或将一个产品移植到一个新的平台上，瀑布模型也特别合适。
* 纯瀑布模型能够降低管理费用，因为你可以预先完成所有计划。
* 对于那些容易理解但很复杂的项目，采用纯瀑布模型比较合适，因为可以用顺序方法处理问题。
* 在质量需求高于成本需求和进度需求的时候，它尤为出色。
* 当开发队伍的技术力量比较弱或者缺乏经验时，瀑布模型更为适合。
  + 缺点
* 在项目初期很难进行充分的项目描述
* 缺乏灵活性
* 螺旋模型
  + 优点
    - 随着迭代的增加，风险程度降低
  + 缺点
    - 比较复杂，需要责任心、专注和管理等方面的知识
* 原型法
  + 优点
* 从实践中学习(Learning by doing)
* 改善的通信
* 改善的用户参与
* 使部分已知的需求清晰化
* 展示描述的一致性和完整性
* 可能可以减少文档
* 减少了维护成本
* 特征约束（利用工具构造原型可以将某些特性落到实处，而非在纸上写的那样容易失误）
* 试验是否能产生期待的结果
  + 缺点
    - 用户有时误解了原型的角色，例如他们可能误解原形应该和真实系统一样可靠
    - 缺少项目标准，进化原型法有点像编码修正
    - 缺少控制，由于用户可能不断提出新要求，因而原型迭代的周期很难控制
    - 额外的花费：研究结果表明构造一个原型可能需要10%额外花费
    - 运行效率可能会受影响
    - 原型法要求开发者与用户密切接触，有时这是不可能的。例如外包软件

1. 软件工作量评估
2. 软件工作量估算困难的原因

* 估算困难是由于软件的本质带来的，特别是其复杂性和不可见性。
* 软件开发是人力密集型工作的，因而不能以机械的观点来看待
* 传统的工程项目经常会议相近的项目做参考，不同的只是客户和地点，而绝大部分软件项目是独一无二的。
* 新技术的不断出现和应用。
* 缺少项目经验数据，许多组织无法提供原有项目数据，而即使提供了这些项目数据，也未必非常有用。

1. 估算的渐进性，准确性和精确性，准确是结果与目标之间有多近，精确是结果有多少有意义的位数。
2. 何时度量

* 策略计划：选择合适的项目
* 可行性分析
* 系统描述：实现各个需求的工作量需要被衡量
* 评估供应商的建议
* 项目计划：
  + 项目进行过程中，估算越来越准确
  + 在项目开始阶段考虑的是用户需求，不考虑实现，但是为了估算，有时需要考虑一些实现方法

1. 过高和过低估计的问题

* 过高估计
  + - Parkinson法则：给的时间越多，工作花费的时间也越多
    - Brook法则：当人数增加后，项目所需的工作量 将不成比例的增加。当团队规模变大后，由于管理，协调和通信的增加，将造成工作量的增加。因而“投入更多的人将使延期的工作更加延期
* 过低估计
  + - 质量降低
    - Weinberg的可靠性零法则“如果系统不必可靠，那么它可以满足任何目标”。

1. 估算的基础和技术
   * 基础
     + 历史数据
     + 工作度量
     + 复杂性

* 技术
* 算术模型
* 专家判断
* 对比法
* Parkinson：能够使用的参与该项目的人力
* 赢利价格：赢得合同的价格
* 自顶向下：首先定义整个项目的工作量，然后分解到各个部分
* 自下而上：各个部分的工作量先估算出来，然后进行合成

1. **Albrecht功能点分析**

* 它是Albrecht在IBM工作时发现的自顶向下的方法
* 功能点法（Function Points）的基本点是计算机信息系统包括五个主要部件或者外部用户类型，它们是：
* 外部输入：应用数据
* 外部输出：提供给用户的面向应用的信息
* 内部逻辑文件：逻辑主文件
* 外部接口文件：与其它系统交换信息
* 外部查询：在线的输入以获得立即的结果
* 加权因子的确定



1. COCOMO模型 参数化模型