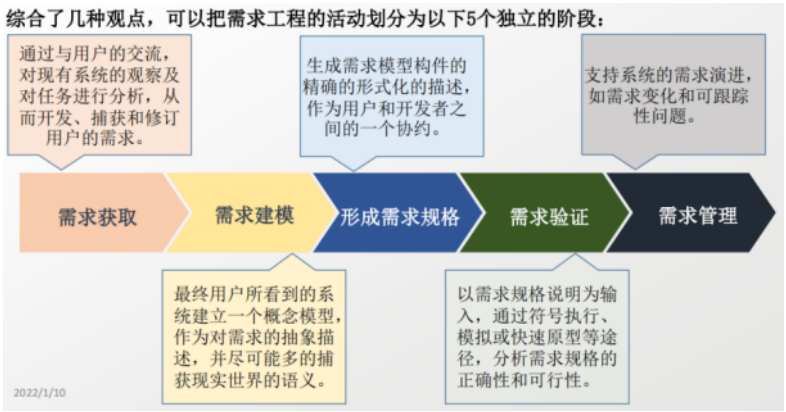
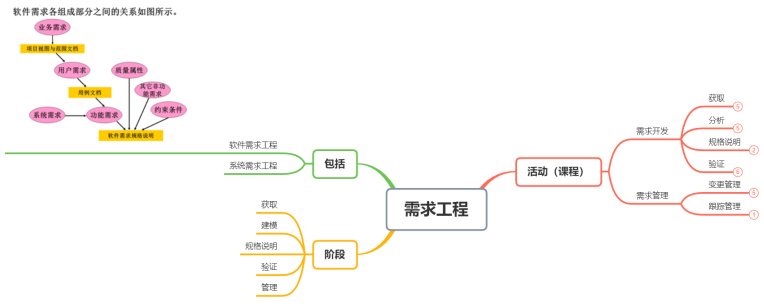
软需汇总

- 背完之后
  - 第一章ppt作业
  - 样卷答案
- 第一章　需求工程与软件需求
  - 需求工程
    - 概念（定义、特点、分类）
      - **1）需求工程定义**：需求工程是指应用已证实有效的技术、方法进行需求分析，确定客户需求，帮助分析人员理解问题并定义目标系统的所有外部特征的一门学科。
      - 区分：
        - 软件需求工程是一门分析并记录软件需求的学科，它把系统需求分解成一些主要的子系统和任务，把这些子系统或任务分配给软件，并通过一系列重复的分析、设计、比较研究、原型开发过程把这些系统需求转换成软件的需求描述和一些性能参数
      - **2）需求工程特点**：它通过合适的工具和记号系统地描述待开发系统及其行为特征和相关约束，形成需求文档，并对用户不断变化的需求演进给予支持。
      - **3）需求工程的分类**：系统需求工程和软件需求工程。
    - 需求工程包含的阶段

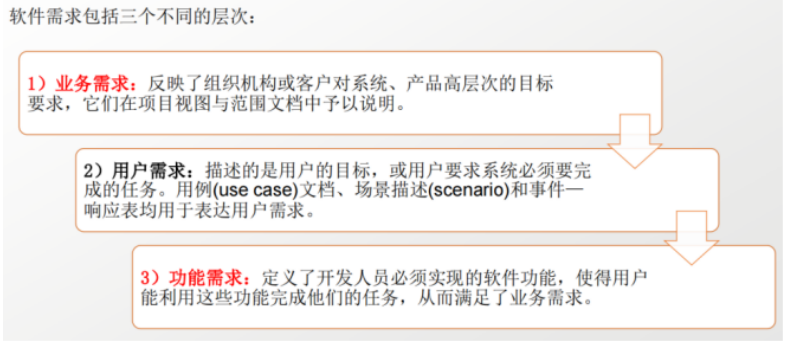


    - 思维导图



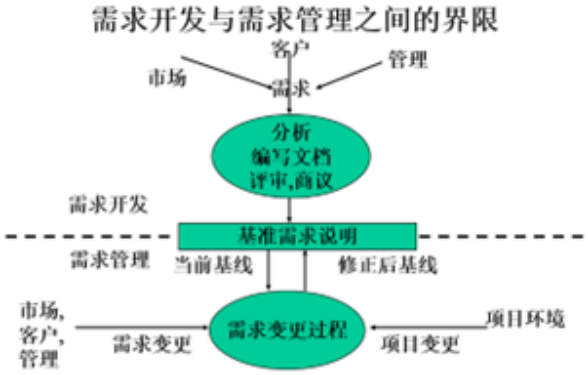
  - 软件需求

- 定义
  - 本课程：
    - 软件需求是指用户对目标软件系统在功能、行为、性能、设计约束等方面的期望。
  - IEEE软件工程标准词汇表（1997年）对需求的定义
    - 1）用户为解决某个问题或达到某种目标而需具备的条件或能力(Capability)
    - 2）系统或系统部件要满足合同、标准、规范或其它正式规定文档而必须满足的条件或必须具备的能力。
    - 3）一种反映上面(1)或(2)所描述的条件或能力的文档说明
- 层次和分类



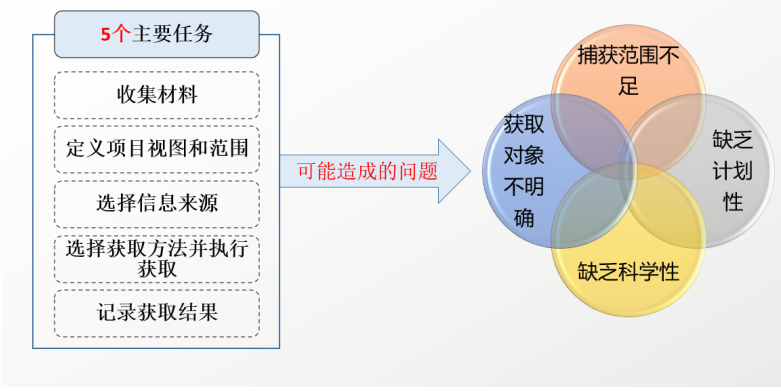
- 特点/对于整个开发过程的重要性
  - 需求分析是介于系统分析和软件设计阶段之间的桥梁
    - 1. 需求分析以系统规格说明和项目规划作为分析活动的基本出发点，并从软件角度对它们进行检查与调整
    - 2. 另一方面，需求规格说明又是软件设计、实现、测试直至维护的主要基础
    - 3. 良好的分析活动有助于避免或尽早剔除早期错误，从而提高软件生产率，降低开发成本，改进软件质量

- 软件需求工程的内容
  - 需求工程的内容及层次分解
  - 需求获取
  - 需求分析
    - 定义：需求开发的核心任务，是获取用户需求后的一个粗加工过程，通过修正错误、补充遗漏，消除不一致等，以获得用户对软件系统的真正需求。
    - 具体描述
      - 1）是对业务的分析：对问题域进行研究，而非系统结构。
      - 2）是一种规范化活动。发现矛盾和冲突，并予以协调解决。
      - 3）是一种分解活动。按职责划分成不同的主题域、分解为组成该主题域的所有流程，在分解到业务活动（用例）、业务步骤。

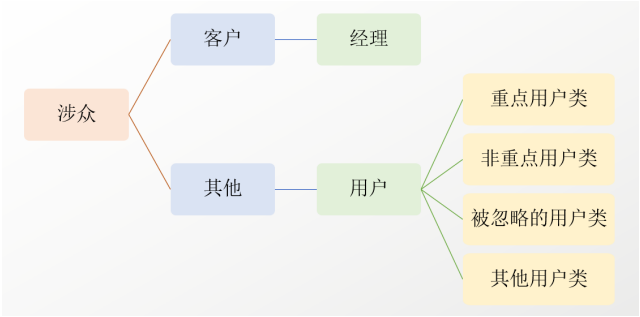
        - 4）是一种提炼和整合活动：将用户的原始需求合并到业务活动中，将分业务流程合并得到全局业务流程图，将各业务事件的用例图片段合并成全局用例模型等。
    - 编写需求规格说明书
        - 编写需求规格说明书是将需求分析结果文档化的过程。它阐述所开发的软件系统必须提供的功能和性能，以及所要考虑的限制条件，是系统测试、用户文档、设计和编码的基础。
    - 需求验证
        - 定义：指审查需求规格说明是否正确和完整地表达了用户对软件系统的需求。
    - 需求管理
    - 需求评审
    - 需求开发与需求管理之间存在的界限
        -
        

- 导致发生不合格的需求的原因
    - 无足够用户参与；
    - 用户需求的不断增加；
    - 模棱两可的需求；
    - 不必要的特性；
    - 过于精简的规格说明；
    - 忽略了用户分类；
    - 不准确的计划
- 不适当的需求过程所引起的一些风险
    - 风险1：用户参与不多导致产品无法被接受
    - 风险2：用户需求的增加带来过度的耗费和降低产品的质量。
    - 风险3：模棱两可的需求说明可能导致时间的耗费和返工。
    - 风险4：用户增加一些不必要的特性和开发人员画蛇添足。
    - 风险5：过分简略的需求说明以致遗漏某些关键需求。
    - 风险6：忽略了用户分类
    - 风险7：不完善的需求说明使得项目计划和跟踪无法准确进行。

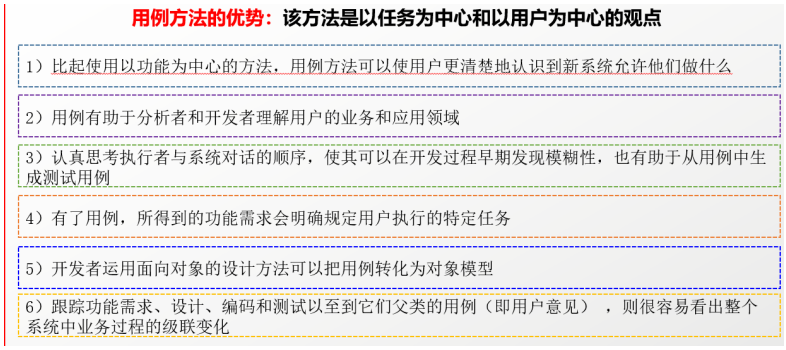
- 良好需求具有的特性
  - 完整性
  - 正确性
  - 可行性
  - 必要性
  - 划分优先级
  - 无二义性
  - 可验证性
  - 一致性
  - 可修改性
  - 可跟踪性
- 高质量的需求过程带来的好处
  - 1）开发后期和整个维护阶段重做的工作大大减少
  - 2）容易使得多方风险承担者在产品开发中通力合作
  - 3）收集需求能使开发小组更好地了解市场，在产品开发前了解这些比在遭到客户批评后才意识到要节约很多成本
  - 4）让用户积极参与需求收集过程能使产品更富有吸引力，而且有利于建立良好的客户关系
  - 5）将选定系统的需求明确地分配到各软件子系统，能简化软硬件的集成，确保软硬件系统功能的恰当匹配
  - 6）有效的变更控制和影响分析过程也能降低需求变更带来的负面影响
  - 7）清晰、无二义性的需求文档有利于系统测试，确保产品质量，使所有风险承担者感到满意
- 第二章　需求获取
  - 内容
    - 项目范围确定；用户确定；用例确定；系统事件和响应；
  - 获取方法
    - 讨论会议、观察工作过程、问答式对话、启发式诱导等
  - 主要任务



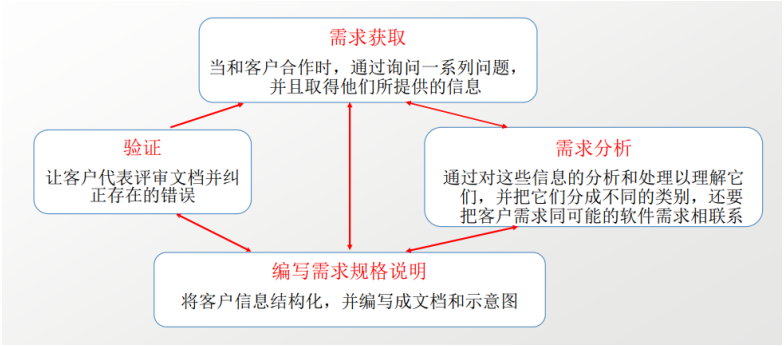
- 项目视图与范围
  - 项目视图和范围的基本内容
    - 项目视图：项目视图描述了产品所涉及的各个方面和在理想情况下最终所具有的功能；
      范围：范围描述了产品应包括的部分和不应包括的部分。
  - 业务需求的作用
    - 业务需求不仅决定了应用程序所能实现的业务任务(用例)的设置（所谓的应用宽度），还决定了对用例所支持的等级和深度
  - 项目视图与范围文档
    - 视图的解决方案
      - **项目视图的解决方案**为系统建立了一个长远的项目视图，它将指明业务目标。
      - **项目视图**为在软件开发生存期中作出决策提供了相关环境背景。
    - 关联图
      - 对范围的描述确立了正在开发的系统与周围所有事物之间的界限和联系，用关联图说明
- 寻找客户的需求
  - 征求客户意见的步骤
    - 1）明确项目用户需求的来源
      2）明确使用该产品的不同类型的用户
      3）与产品不同用户类的代表进行沟通
      4）遵从项目的最终决策者的意见
  - 需求的来源
    - 1）与潜在用户进行交谈和讨论
      2）把对现有产品或竞争产品的描述写成文档
      3）系统需求规格说明
      4）现有系统的问题报告和改进要求
      5）市场调查和用户问卷调查
      6）观察用户如何工作
      7）用户工作的情景和内容分析
      8）事件和响应
  - 用户类
    - 用户类的功能和非功能需求

- 组成
  - 

- 寻找用户代表
  - 用户代表定义
    - **产品（用户）代表：**组建开发组时，每一个工程项目都包括为数不多的核心参与者，这些参与者来自相关的用户团体，并提供客户的需求，称这些人为产品代表(project champion)。
  - 用户代表要求
    - 1）一个优秀的产品代表对新系统有明确的认识并有极大的热情
      2）产品代表必须是有力的交流者，且是同组成员的代表者
  - 产品代表的活动
    - 计划
    - 验证和确认
    - 用户帮助
    - 变更管理
- 基于用例的方法
  - 用例、执行者定义
    - **用例：**一个用例描述了系统和一个外部“执行” (actor) 的交互顺序，这体现执行者完成一项任务并给某人带来益处。
      - 1）一个单一的**用例**可能**包括**完成某项任务的**许多逻辑相关任务**和**交互顺序**
      - 2）一个用例是**相关的用法说明的集合**
    - **执行者：**是指一个人、或另一个软件应用、或一个硬件、或其它一些与系统交互以实现某些目标的实体。执行者可以映像到一个或多个可以操作的用户类的角色。
  - 用例和功能需求关系
    - 每一个用例可引伸出多个功能需求，这将使执行者可以执行相关的任务；并且多个用例可能需要相同的功能需求。
  - 用例的好处
    -

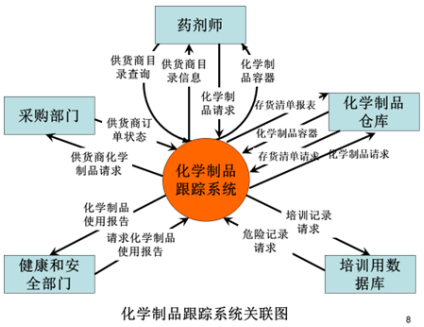


- 软件的质量属性
  - 定义
    - 用户对产品如何良好地运转抱有许多期望。包括：产品的易用程度如何，执行速度如何，可靠性如何，当发生异常情况时系统的处理能力。这些特性合起来被称为软件质量属性或质量因素。
  - 质量特性
    - 正确性、可靠性、效率、完整性、易用性、可维护性、测试性、灵活性、可移植性、可重用性、互操作性（McCall）
- 需求获取小结
  - 需求获取的要点
    - 项目视图和范围的确定、选择用户和产品代表、使用实例方法的应用
  - 需求开发中的四个过程及其相互关系
    -



  - 需求工程和软件需求工程的区别
    - 需求工程：是指应用已证实有效的技术、方法进行需求分析、确定客户需求、帮助分析人员理解问题并定义目标系统的所有外部特征的一门学科。它通过合适的工具和记号系统地描述待开发系统及其行为特征和相关约束、形成需求文档、并对用户不断变化的需求演进给予支持。
    - 需求工程可分为：系统需求工程（如果是针对由软硬件共同组成的整个系统）和软件需求工程（如果仅是专门针对纯软件部分）
    - 软件需求工程：是一门分析并记录软件需求的学科，它把系统需求分解成一些主要的子系统和任务，把这些子系统或任务分配给软件，并通过一系列重复的分析、设计、比较研究、原型开发过程把这些系统需求转换成软件的需求描述和一些性能参数。

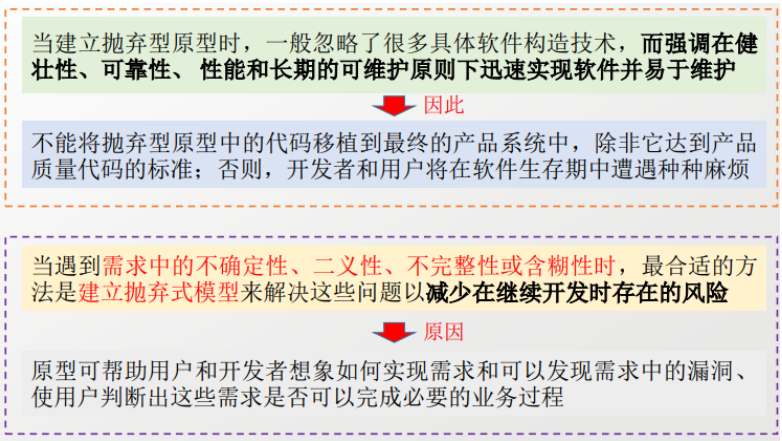
- 第三章　需求分析
  - 需求分析概述
    - 需求分析内容和目的
      - 需求分析(requirementanalysis)包括提炼、分析和仔细审查已收集到的需求，以确保所有的风险承担者都明白其含义并找出其中的错误、遗漏或其它不足的地方。
      - 分析员通过评价来确定是否所有的需求和软件需求规格说明都达到了优秀需求说明的要求。
      - 分析的目的在于开发出高质量和具体的需求，这样就能作出实用的项目估算并可以进行设计、构造和测试。
      - 需求分析主要是通过需求建模来实现。
    - 需求分析的主要任务
      - 绘制系统关联图、建立数据字典、建立用户界面原型、为需求建立模型、确定需求优先级
  - 系统关联图
    - 定义：系统关联图是用于定义系统与系统外部实体间的界限和接口的简单模型，同时它也明确了通过接口的信息流和物质流。
    - 例子：



化学制品跟踪系统关联图

  - 创建用户界面原型
    - 原型定义
      - 软件原型是所提出的新产品的部分实现或可能的实现
    - 用户界面原型的优势
      - 软件原型是一种技术，可以利用这种技术减少客户对产品不满意的风险
      - 可以解决不明确、不清晰的需求
      - 可以通过此去想象软件产品在特定环境中如何运行’
      - 建立有趣的原型可以使新产品实在化
      - 更直观，用户更愿意看
    - 使用原型的目的、原因
      - 1）目的：
        - 明确并完善需求
        - 探索设计选择方案

        - 发展为最终的产品原型
    - 2）原因
        - 解决不确定性问题
        - 解决二义性问题
        - 使原型更易于理解
- 原型类别
    - 水平原型（行为原型或演示原型）（所提出功能经常并没有真正的实现）
    - 垂直原型（结构化原型）（实现了一部分功能）
    - 抛弃型原型（探索型原型）（达到预期目的后将会被抛弃）
        - 什么时候使用抛弃型原型
        -

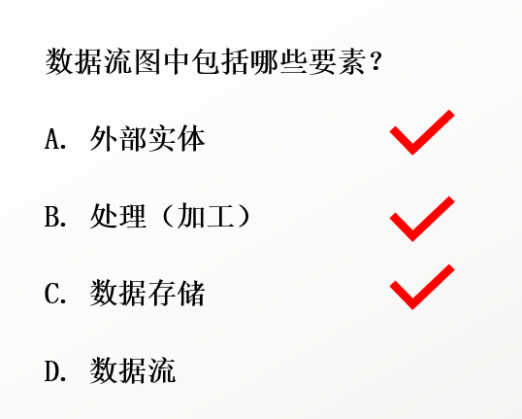


    - 进化型原型（清楚定义了需求的情况下，进化型原型为开发渐增式产品提供了坚实的构造基础）（适用于web开发项目）
    - 原型方法的使用
        -

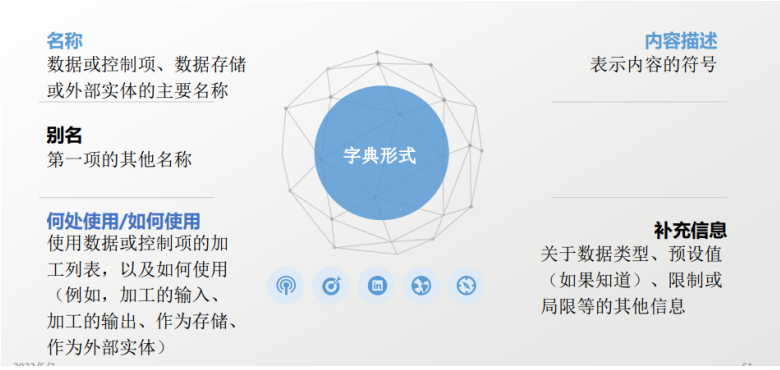
	抛弃型	演化型
水平原型	1）澄清并细化使用实例和功能需求 2）识别遗漏的功能 3）研究用户界面方法	1）实现核心的使用实例 2）根据优先级实现附加的使用实例 3）使系统适应快速变化的业务需求
垂直原型	证明技术的可行性	1）实现并扩充核心客户端／服务器功能层和通信层 2）实现并优化核心算法 3）测试并调整性能

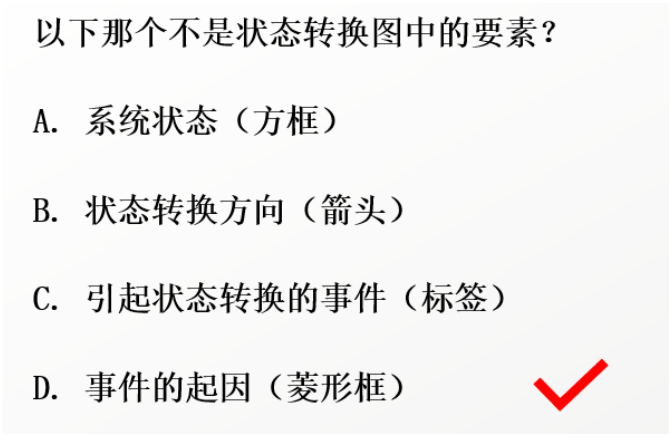
    - 书面原型（可以呈现系统某部分是如何实现的）
    - 电子原型
- 原型风险
    - 1）最大的风险是用户或者经理看到一个正在运行的原型从而以为产品即将完成。
    - 2）用户重点关注的是系统“how（如何）”的那些方面，他们关注用户界面的外观如何，以及如何操作这些界面。如果使用看起来很真实的原型，用户就容易忘却在需求阶段他们应该关注那些“what（什么）”方面的问题。
    - 3）用户将根据原型的性能来推断最终产品的期望性能。

- 确定需求的优先级别
  - 需求优先级的意义
    - 若无:
      - 当项目接近尾声时，开发者必须抛弃掉一些不必要的功能以保证，若没有提前设置优先级，就只能匆忙决定
      - 若在判断出需求的低优先级前已实现了将近一半的特性，则这将是一种浪费
    - 若有:
      - 在项目的早期阶段设定优先级有助于你逐步作出相互协调的决策，而不是在最后阶段匆忙决定。
- 需求的图形化分析
  - 数据流图(DFD)
    - 定义
      - 一个数据流图确定了系统的转化过程、系统所操纵的数据或物质集合(存储)，以及过程、存储、外部世界之间的数据流或物质流。
    - 要素
      -

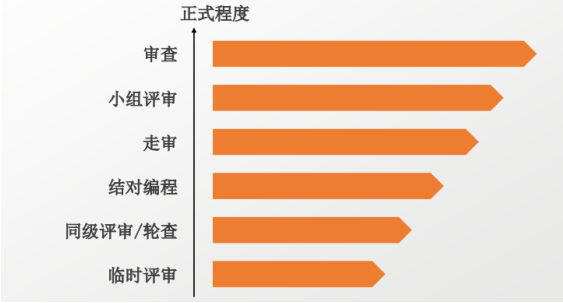


  - 实体关系图(ERD)
    - 定义
      - 实体-关系图(Entity—Relationship Diagram，ERD)描绘了系统的数据关系。
      - 实体(entity)是物理数据项(包括人)或者数据项的集合，
  - 状态转化图(STD)
    - 定义
      - 所有软件系统都包含功能行为、数据操作和状态转变
      - 当满足所定义的标准时，状态就会发生改变
  - 对话图
    - 定义
      - 许多用户界面可以用状态转换图中的一种称为对话图来建模

        - 对话图代表了一个高层抽象的用户界面体系结构
    - 类图
        - 定义
            - 是用图形方式叙述面向对象分析所确定的类以及它们之间的关系
- 数据字典
    - 定义
        - 是一个对系统所用到的所有数据项和结构定义的共享仓库
    - 功能
        - 需求阶段，数据字典至少应该定义客户数据项以确保客户与开发小组是使用一致的定义和术语
        - 数据字典可以把不同的需求文档和分析模型紧密结合在一起
        - 数据字典的维护独立于软件需求规格说明，并且在产品的开发和维护的任何阶段，各个风险承担者都可以访问
    - 组成
        -
        

        -
        

- 第四章　编写需求规格说明
    - 定义
        - 精确的阐述一个软件系统必须提供的功能和性能以及他所要考虑的限制条件或约束。
    - 角色

        - **软件需求规格说明不仅是系统规划、设计和编码的基础，也是系统测试和用户文档的基础，也是所有子系统项目规划、设计和编码的基础。**
        - 它应该尽可能完整地描述系统预期的外部行为和用户可视化行为。**除了设计和实现上的限制，软件需求规格说明不应该包括设计、构造、测试或工程管理的细节。**
    - 基线
        - 指正在开发的软件需求规格说明向已通过评审的软件需求规格说明的过渡过程。
    - 把用户界面的设计编入SRS
        - 消极方面
            - 1）屏幕图像和用户界面描述的是解决方案而不是需求。
            - 2）用户界面的布局不能替代用户需求和功能需求。
        - 积极方面
            - 1）对可能的用户界面进行研究会使需求无论是对用户，还是对开发人员都变得实实在在。
            - 2）直观的用户界面更有助于项目计划的制定和预估。
- 第五章　需求验证
    - 概念
        - 就是审查需求规格说明是否**正确**和**完整**地表达了用户对软件系统的需求。
    - 主要方法
        - 评审需求文档
        - 以需求为依据编写测试用例
        - 编写用户手册
        - 确定合格的标准
    - 需求评审
        - 概念
            - 需求评审是高质量需求文档要求对其功能的正确性、完整性和清晰性，以及非功能需求给予评价。
        - 6种评审方法
            -
            

            - 非正式评审（不需记录在案）
            - 正式评审（需要记录在案）最有效的技术评审----审查

- 审查过程
  - 参与者
    - 可能需要包括一个开发人员、一个测试人员、一个项目经理和一个用户文档编写人员
  - 审查角色
    - 作者、调解者、读者、记录员
  - 审查阶段
    -

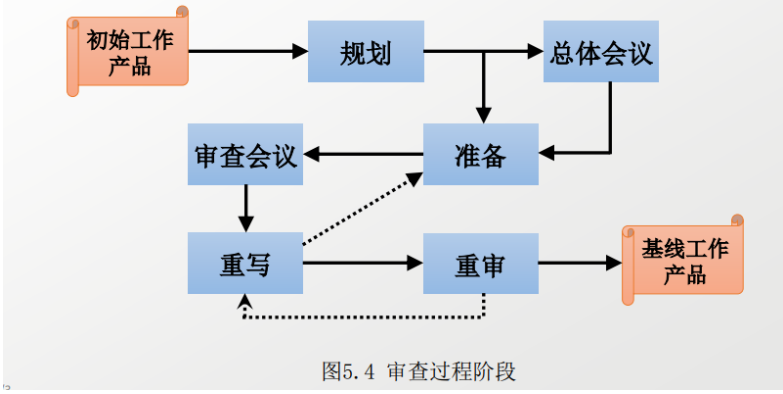
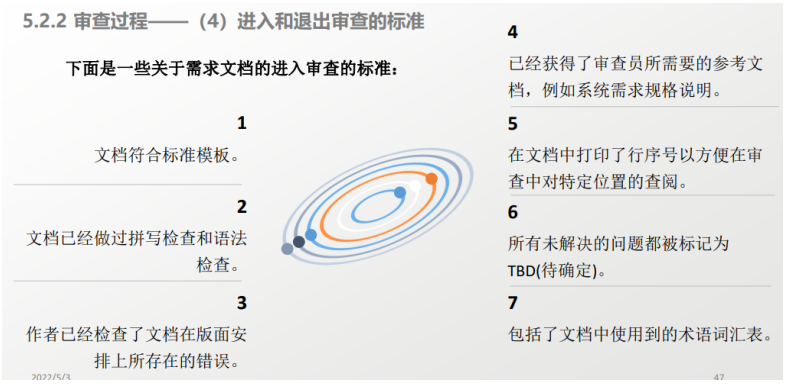


图5.4 审查过程阶段

  - 进入和退出审查的标准
    - 进入
      -



    - 退出
      -

- 困难及解法
  - 大型的需求文档
    - 只在把软件需求规格说明作为基线时才进行审查

        - 在开发软件需求规格说明时，可以采用非正式的、渐增式的审查
    - 庞大的评审小组
        - 审查组分成若干小组并行地审查
    - 审查员在地域上的分散
        - 视频会议等方式
    - 审查员准备不足
        - 正式之前，确保审查员们过了一遍
- 第六章　需求管理
    - 需求管理的原则和实践
        - 需求管理内容
            - 变更控制、版本控制、需求跟踪、需求状态跟踪
        - “ 需求基线 ”
            - 是团队成员已经承诺将在某一特定产品版本中实现的功能性和非功能性需求的一组集合
            - 对比：基线 👆 （ctrl F）
        - 跟踪需求的状态
            -

已建议	该需求已被有权提出需求的人建议
已批准	该需求已被分析，估计了其对项目余下部分的影响(包括成本和对项目其余部分的干扰)，已用一个确定的产品版本号或创建编号分配到相关的基线中，软件开发团队已同意实现该项需求
已实现	已实现该需求代码的设计、编写和单元测试，已被跟踪到相关的设计元素和编码元素
已验证	使用所选择的方法已验证了实现的需求，例如测试和检测；该需求已被跟踪到测试用例。该需求现在被认为完成了
已删除	计划的需求已从基线中删除，但包括一个原因说明和做出删除决定的人员

        - 评估需求管理的工作量
            - 每个项目除了给需求开发分配资源外，也必须规划需求管理活动的任务和资源
        - 配置管理（CM）：
            - 定义
                - 是通过技术或行政手段对产品及其开发过程和生命周期进行控制、规范的一系列措施。
        - 软件配置管理（SCM）：又称软件形态管理。是一种标识、组织和控制修改的技术。
            - 内容
                - Version Control-版本控制
                - Change Control-变更控制
                - Process Support-过程支持

- 变更管理
  - 需求蔓延
    - 定义
      - 需求蔓延是指在软件需求基线已经确定后又要增添新的功能或进行较大改动。
    - 管理与控制
      - 第一步：就是把新系统的视图、范围、限制文档化并作为业务需求的一部分
      - 另一个有效的技术：原型法。这个方法能够给用户提供预览所有可能的实现，以帮助用户与开发者沟通从而准确把握用户的真实需求
      - 最有效：敢于说不
  - 变更控制
    - 好处
      - 一个好的变更控制过程给项目风险承担者提供了正式的建议需求变更机制
      - 严格控制变更管理策略可以减少变更造成的混乱。
      - 改进的需求开发技术可以减少面临的需求变更的数量。
      - 效率高的需求获取和管理策略将增强按时交付的能力。
    - 角色
      -

表6.2 变更管理活动中可能的项目角色

角 色	描述及责任
CCB主席	变更控制委员会的主席，在CCB意见不一致情况下可以独自做出决定.选定评估者和修改者.
CCB	决定采纳或拒绝针对某项目所建议的变更请求的团体
评估者	应CCB主席的要求分析所建议的变更带来影响的人员
修改者	负责实现已经被认可的请求变更，按时更新变更状态的人员
提议者	提交新变更请求的人
请求接受者	接收提交的变更请求的人
验证者	负责决定变更是否已正确执行的人

    - 变更控制过程
      - 开始条件：通过正式的渠道接受一个合法有效的变更请求。
      - 结束条件：请求状态为“已否决”、“已结束”或“已取消”。
      - 状态转换图
        -

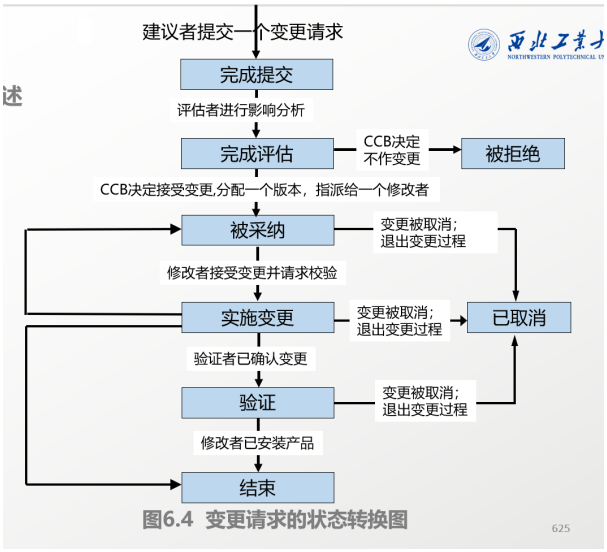


图6.4 变更请求的状态转换图

- 变更控制委员会
  - 职责
    - （1）制定决策
    - （2）交流情况（交流状态）
    - （3）重新协商约定
    - 变更总是有代价的。
- 需求变更影响分析
  - **变更只能在项目时间、预算、资源的限制内进行协商。**
  - 分析过程
    - **A:（变更涉及的问题核对表）**
    - **B:（变更影响的软件元素核对表）**
    - **表6.6：（评估需求变更的劳动时数表）**
    - 1)按表A进行一遍。
    - 2)按表B进行一遍，要使用有效的跟踪能力信息。有一些需求管理工具包含影响分析报告并且能发现受变更影响的系统元素。
    - 3)使用如表6.6所示的一张工单来评估预期任务要求的工作量，绝大多数的变更仅要求工单所列任务的一部分。
    - 4)求评估工作值的总和。
    - 5)确认任务执行的顺序，这些任务如何同当前的计划任务配合?
    - 6)决定变更是否处于项目的临界路径。如果一个处于关键路径的任务延期，项目的完成之日将遥遥无期。每个变更都会消耗资源，如果能避免变更影响关键任务，则变更不会造成整个项目延期。
    - 7)估计变更如何影响进度和费用。
    - 8)通过与其它任意需求的收益、代价、成本和技术风险的比较来评估变更的优先级。
    - 9)向变更控制委员会报告影响分析结果，他们可以在采纳或拒绝变更的决策过程中使用这些评估信息。

- 需求跟踪
  - 定义
    - 需求跟踪包括编制每个需求同系统元素之间的联系文档。
    - 这些元素包括别的需求、体系结构、其他设计部件、源代码模块、测试、帮助文件、文档等。**跟踪能力信息使变更影响分析十分便利，有利于确认和评估实现某个建议的需求变更所必须的工作。**
  - 内容
    - 记录了单个需求之间的父层、相互连接、依赖的关系。
  - 意义
    - 如果不能把设计元素、代码段或测试回溯到一个需求，就可能有一个“画蛇添足的程序”。然而，若这些孤立的元素表明了一个正当的功能，则说明需求规格说明书漏掉了一项需求。
  - 方式
    - **表示需求和别的系统元素之间的联系链的最普遍方式是使用需求跟踪距阵。**
    - 表示跟踪信息的另一个方法是通过矩阵的集合
    - 跟踪能力链：
      - 定义
        - 跟踪能力(联系)链使我们能跟踪一个需求使用期限的全过程，即从需求源到实现的前后生存期。
      - 用法
        - 由于开发过程中系统需求转变为软件需求、设计、编码等，所以通过**定义单个需求和特定的产品元素之间的(联系)链可从需求向前追溯。**
      - 对应关系
        - 一对一
        - 一对多
        - 多对多
      - 信息源
        -

联系链的源对象类型	联系链的目的对象类型	信息源
系统需求	软件需求	系统工程师
用例或业务规则	功能性需求	需求分析员
功能性需求	功能性需求	需求分析员
功能性需求	软件体系结构元素	软件架构师
功能性需求	其他设计元素	设计人员或开发人员
设计元素	代码	开发人员
功能性需求	测试用例	测试工程师

- 需求管理工具
  - 分类

    - 这些工具最大的区别是以数据库还是以文档为中心。
    - 以数据库为核心的产品把所有的需求、属性和跟踪信息存储在数据库中。
    - 以文档为中心的方法使用Word或Adobe公司的FrameMaker等字处理程序制作和存储文档。
- 使用工具的好处
    - 1)管理版本和变更
    - 2)存储需求属性
    - 3)进行影响分析
    - 4)跟踪需求状态
    - 5)访问控制
    - 6)与涉众进行沟通
    - 7)重用需求
- 工具选择
    - 需求管理工具使用多用户数据库保存与需求相关的信息
    - 小型项目：电子表格或简单的数据库
    - 大型项目：商业需求管理工具
    - 开发工具：在考虑自行开发工具前可以先调查一下现存可用的成熟工具