需求

第二章

1. 需求定义：
   1. 1用户为了解决问题或达到某些目标而需要具备的条件或能力
   2. 2系统或系统部件为了满足合同标准规范或其他正式文档所规定的要求而需要具备的条件或能力
   3. 对1或2中一种条件或一种能力的文档化表述
2. 问题域特性：
   1. 问题域自治的规律称为问题域特性
   2. 包括结构特性和行为特性
3. 解系统：
   1. 软件系统通过影响问题域，能够帮助人们解决问题，称为解系统
4. 需求和问题域关系：
   1. 需求是用户对问题域当中实体状态或事件的期望描述
   2. 理解需求要以问题域为背景
5. 规格说明：
   1. 解系统为满足用户需求而提供的解决方案，规定了解系统的行为特征
   2. 主要包括两个部分：
      1. 对共享现象的描述
      2. 系统对共享现象所施加的操作的描述
6. 共享现象：
   1. 软件系统能够与问题域进行交互和相互影响的原因在于，软件系统中的某些部分对问题域中的某些部分具有模拟特性
   2. 换句话说，软件系统当中含有问题域某些部分的模型
   3. 问题域中的某些信息能够和模型中的信息建立映射关系
   4. 这些通过映射建立的共同知识，就是问题域和解系统之间的共享现象
7. 业务需求：
   1. 系统建立的战略出发点，表现为高层次的目标，它描述了组织为什么要开发系统
8. 用户需求：
   1. 执行实际工作的用户对系统所能完成的具体任务的期望，描述了系统能够帮助用户做些什么
9. 系统级需求：
   1. 用户对系统行为的期望。一系列的系统行为联系在一起可以帮助用户完成任务，满足业务需求
10. 需求分类
    1. P31、32、35、36、37、38、39、42、43、44、45、46、47，注意可用性和易用性
    2. P48例题
11. 最后的思考题没好好看

第五章

1. 问题分析：
   1. 过程：明确问题、发现业务需求、定义解决方案及系统特性、建立系统边界
   2. 明确问题：
      1. 过程：
         1. 获取问题
         2. 对问题达成共识
         3. 发现问题背后的问题
      2. 描述问题格式：ID、提出者、关联者、问题、影响 参见P15‘
      3. 问题明确性：易于理解、能指明解决方向 P16
   3. 发现业务需求：
      1. 是问题的反面
      2. 尤其要有可验证性
   4. 定义解决方案及系统特性：
      1. 确定高层次解决方案，确定系统特性和解决方案的边界，确定解决方案的约束
      2. 解决方案用表格描述 P22
      3. 确定问题解决边界可以用用例图（OO），上下文图
         1. 上下文图的内容：它需要的信息由谁提供？它产生的信息由谁使用？谁控制它的执行？谁会影响它的执行？ P25、P27
      4. 确定解决方案的约束，见P28表
2. 目标分析
   1. 目标的形式化表达 P46 P52没看
   2. 高低层次目标示例 P48
   3. 功能目标、非功能目标 P50
   4. 软硬等分类 P51
   5. 各种图示 P55-P67
   6. 过程：
      1. 高层目标获取
      2. 底层目标获取
      3. 目标分析：精化与分解，建立系统的目标模型
      4. 目标实现：讨论可能的候选解决方案，确定最终的系统详细需求和解决方案
   7. NFR P91
3. 业务过程分析
   1. 活动图：令牌平衡
   2. 图示：还是看书吧
   3. 要注意的点 P113
4. 定义系统边界
   1. 用用例图或上下文图表示。用例图里的参与者不一定是人 P123

第六章

1. 涉众：所有能够影响软件系统的实现或者被实现后的软件系统所影响的关键团体和个人
2. 用户参与的重要性？
3. 小型系统、组织级系统、战略信息系统、组织间系统
4. 涉众分析过程包括哪些步骤？
   1. 涉众识别、涉众描述、涉众评估、涉众代表选择、参与策略制定
5. 涉众识别
   1. 先膨胀后收缩：简单易用，如果涉众群体比较复杂，可能会出现遗漏
   2. 检查列表
   3. 涉众网络
   4. ADM图：葫芦软目标，椭圆目标，六边形任务，方框资源
6. 涉众描述
   1. 需要描述的信息：P28 涉众、主要目标、态度、主要关注点、约束
7. 涉众评估
   1. 优先级评估：Power-Interest划分观众、环境设定者、被影响者、参与者
   2. 风险评估：Power-Attitude 划分强弱反对者、支持者
   3. 共赢分析：P36 Stakeholder/Issue关系图 P40图看不懂
8. 涉众代表选择
   1. 代表采样：完整采样、态度积极、数量适中、比例恰当
   2. 用户替代原：市场人员、服务咨询人员、技术支持人员、领域专家
9. 参与策略制定
   1. 参与策略安排表：P45

第七章

1. 需求获取轮次安排 P5
2. 获取方法：
   1. 面谈 常规
   2. 集体面谈 快速
   3. 头脑风暴 发明需求
   4. 原型 不确定性
   5. 观察 情景性
3. 要把需求获取笔录整理成用例与场景
4. 从用例/场景描述得到用户需求列表 P15
5. 从场景得到用例 P16
6. 用例书写：ID 名称 参与者 描述 优先级 触发条件 前置条件 后置条件 正常流程 扩展流程 相关用例 业务规则 特殊需求（尤其是非功能性需求） 假设（建立用例时做的假设）、
7. 基于场景/用例模型进行需求获取： P38 （基于……迭代……验证……维护）
8. 如何指导获取：
   1. 初始系统用例涉及的主题需要获取
   2. 概要用例描述中发现的新主题需要获取
   3. 具体用例中发现的模糊、不正确、不完备的细节需要获取

第八章

1. 面谈过程：
   1. 准备：
      1. 阅读背景资料
      2. 确定面谈主题和目标
      3. 选择被会见者
      4. 准备？？？
      5. 确定问题和类型
   2. 主持：
   3. 报告整理
2. 开放式问题
   1. 被会见者对答复的选择可以是开放和不受限制的
   2. 在希望得到丰富（具有一定广度和深度）的信息时选择
   3. 优点：
      1. 让被会见者感到自在
      2. 提供丰富的细节
      3. 让被会见者更感兴趣
      4. 会见者可以在没有太多准备的情况下进行面谈
   4. 缺点：
      1. 可能产生太多不相干的细节
      2. 面谈可能失控
      3. 会花费大量时间才能获得有用的信息量
      4. 可能会使会见者看上去没有准备
3. 封闭式问题
   1. 答案有基本的形式，被会见者的回答是受限制的（“列出头两项”也是）
   2. 优点：
      1. 节省时间
      2. 切中要点
      3. 保持对面谈的控制
      4. 短时间内探讨大范围问题
      5. 得到贴切的数据
   3. 缺点：
      1. 使被会见者厌烦
      2. 得不到丰富的细节
      3. 不利于和面谈者建立友好关系
4. 问题准备注意事项
   1. 前期
      1. 开放式问题为主
      2. 决策层和专家为主
      3. 遵循问题——目标——解决方案路线
      4. 分析基本的涉众特点
   2. 后期
      1. 封闭式问题为主
      2. 抓住主题与线索，如任务分解、流程图、界面示意
      3. 问题有针对性（针对任务分解关系、流程正确性、异常、界面中的行为和数据项）
      4. 事先准备面谈记录材料
5. 面谈开始
   1. 握手
   2. 简要重申面谈目标
   3. 准备好笔记本、录音机或其他设备
   4. 用一般的、轻松的、开放式的问题作为开始
6. 面谈主体阶段
   1. 保持有礼貌倾听
   2. 控制面谈过程
   3. 保持面谈主题
   4. 使用探究式问题
   5. 观察被会见者
   6. 使用道具支持
7. 面谈结束
   1. 45分钟到1小时结束，并非要问完所有问题
   2. 总结要点，请被会见者快速检查笔记，确保记录下所有重要信息
   3. 感谢被会见者，给时间让他们问一些感兴趣的问题
   4. 握手话别
8. 面谈的优点和缺点

第九章

1. 原型是一个系统，他内化了一个更迟系统的本质特征。原型系统通常被构造为不完整的系统，以在将来进行改进、补充或替代。如果在最终物件产生之前，一个中间物件被用来在一定广度和深度范围内表现这个最终物件，那么这个中间物件就是最终物件在该广度和深度上的原型。
2. 使用原型是为了应对不确定性
   1. 因为对未来知识有限，而无法确定某种决策的结果
   2. 不确定性广泛存在
   3. 软件工程中存在着大量的不确定性。原型、迭代和验证是人们解决不确定性的主要手段。
3. 不确定性体现在何处
   1. 创新型产品的需求是潜在的，存在不确定性
   2. 用户对相关产品缺乏经验，产品的细节需求存在不确定性
   3. 用户在完成工作的方式上存在困难，产品在整体方案的可行性上存在不确定性
   4. 用户在清晰表达自己需求上存在困难，这些需求具有不确定性
   5. 需求工程师在准确理解用户表达的需求上存在困难，在澄清和理解之前，这些需求存在不确定性
   6. 需求本身的可行性，即可满足性，存在不确定性
4. 过程：
   1. 根据需要确定原型类型
   2. 进行原型开发
   3. 获取用户反馈
   4. 定义所得需求
5. 在需求阶段，由于涉众的需要不明确，所以使用“严格意义上的原型”（用来阐明用户界面或系统功能的某些特定方面）。
6. 原型一定要开发的容易修改
7. 类型：探索式和实验式又叫抛弃式
   1. 探索式：
      1. 以缺陷需求开始，继而不断调整和修正需求的原型开发方式成为探索式
      2. 要尽可能调整各种设计选项
   2. 实验式
      1. 以清晰的用户需求和模糊的实现方法、实现效果、可行性开始，明确需求的可行性和技术实现方案
      2. 定义一个对原型的评估方法，确定评估的属性
   3. 演化式
      1. 以清晰的原型化需求和项目积累下来的原型资产为开始
8. 抛弃式：
   1. 花费最小代价，争取最快速度
   2. 可能使用简易的开发工具和不成熟的构造技术
   3. 可能忽略或简化和原型目的不相关的功能
   4. 必须坚决抛弃
9. 演化式
   1. 质量要达到最终系统的要求
   2. 要易于进行扩展和改进，必须重视其设计
   3. 仅应该用于清晰的需求、规格说明和技术方案
10. 水平原型（绝大部分）：
    1. 仅仅实现选定功能的所有层次之中的某些特定层次
11. 垂直原型：
    1. 触及选定功能的所有层次
    2. 保证真实实现其功能
12. 控制成本：
    1. 用尽可能低的成本开发水平原型
    2. 用简单的介质降低成本
13. 按照成本和真实感排序：
    1. 纸面
    2. 幻灯、动画
    3. 快速语言和工具
    4. 程序语言
    5. 前两种是低保真原型，后两种是高保真原型

第十章

1. 情景性：局部（异常）、模糊（默认知识）、暂时、突现（群体促成）、涉身
2. 突现和涉身要靠民族志，其他三个民族志和采样观察都可以
3. 采样观察

|  |  |  |
| --- | --- | --- |
|  | 时间采样 | 事件采样 |
| 优点 | 通过随机观察减小偏差；  对频繁发生的事件取代表性事件进行观察。 | 允许在行为展开过程中进行观察；  允许对指定的重要事件进行观察 |
| 缺点 | 用分段方式收集数据不能提供全面的信息；  可能漏掉不经常发生但是很重要的事件 | 消耗大量时间；  漏掉频繁发生事件的代表性样本 |
| 适用情景 | 发现异常流程；  验证用户知识和实际工作的不一致的地方 | 获取默认知识；  验证用户知识和实际工作不一致的地方 |

1. 民族志
   1. 优点：
      1. 能够得到信息的深度理解
      2. 能够让真实世界的社会性因素可见化
      3. 打破人们已有的一些错误假设和错误观念
   2. 缺点：
      1. 需要耗费很多时间
      2. 调研结果很难传递到开发过程
   3. 关注点：
      1. 工作的分布式协同（特别要注意利用物件实现的协同和创建这些物件的文书工作）
      2. 工作的计划和程序（关注它们在组织活动中的应用方式，发现实际工作和文档化程序之间的偏离）
      3. 工作的意识（活动是如何对协同中的其他人可见或可理解的）
   4. 规则
      1. 定期记录发现
      2. 尽快记录可能会在观察过程中发生的面谈
      3. 定期的复查和更新自己的想法
      4. 确定管理海量数据的应对策略
2. 文档审查没看

第十一章

1. 分析模型的基本思想：
   1. 是对系统进行思考和推理的一种方式。建模的目标是建立系统的一个表示，这个表示以精确一致的方式描述系统，使得系统的使用更加容易。
   2. 建模的方法有：抽象、分解、投影
   3. 抽象：
      1. 只关注重要信息，忽略次要内容。强调本质特征，减少复杂性。
      2. 另一方面要求人们将认知保留在适当的层次，屏蔽更深层次细节
   4. 分解：
      1. 分而治之，将复杂问题分解为多个相对容易的子问题，并掌握子问题之间的联系
      2. 分解的方案往往还能提供问题的解决方案
   5. 投影：
      1. 多视点方法
   6. 软件分析模型是介于计算模型和业务模型二者之间的模型形式，使用计算模型的组元形式，在组元的表现上采用业务模型的表现方式
   7. 半形式化：不像计算模型那么严谨，比业务模型更加严格
2. 需求建模的通常做法
   1. 先根据获取的问题域信息建立初步的模型
   2. 分析用户需求，对模型进行调整，得到一个中间形式的模型形式
   3. 最后，对调整后的模型进行逻辑推理和验证。如果符合预期的期望，那么它就是最终的解决方案模型
3. 结构化方法
   1. 数据流图
   2. 实体关系图
   3. 上下文图
   4. 状态转移图
   5. 功能实体矩阵
   6. 以数据流为中心，以DFD为核心技术，以lambda验算为数学基础
4. 面向对象方法
   1. 用例图
   2. 类图
   3. 交互图（顺序图、通信图）
   4. 活动图
   5. 状态图
   6. OCL
5. 面向对象方法是如何借鉴结构化方法的
   1. 类图借鉴了实体关系图
   2. 活动图借鉴了数据流图
   3. 状态图借鉴了状态转移图
6. Wieringa框架 P23
   1. 外部功能 用例图
   2. 外部通信 用例图，交互图
   3. 外部行为 交互图
   4. 概念组元 类图
   5. 组元功能 OCL
   6. 组元通信 交互图
   7. 组元行为 活动图，交互图
7. Zachman框架，六行六列，怎么考？P25

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | 数据 | 功能 | 网络 | 人 | 时间 | 动机 |
| 目标范围 |  |  |  |  |  |  |
| 企业模型 |  |  |  |  |  |  |
| 系统模型 |  |  |  |  |  |  |
| 技术模型 |  |  |  |  |  |  |
| 组件模型 |  |  |  |  |  |  |
| 实际运行系统 |  |  |  |  |  |  |

需求：

数据 类图

功能 交互图，活动图，用例图，OCL

网络

人

时间

动机 OCL

1. P32 P33 什么鬼
2. 前期分析手段：
   1. 面向目标分析
   2. 面向问题域分析
   3. 领域分析
   4. 企业建模
3. 需求分析的子活动
   1. 背景分析
   2. 问题分析
   3. 目标分析
   4. 业务分析
   5. 确定系统边界
   6. 需求建模
   7. 需求细化
   8. 确定需求优先级
   9. 需求协商
4. 中间全不懂
5. “需求细化”示例：P43
6. 确定优先级

|  |  |  |
| --- | --- | --- |
|  | 重要 | 不重要 |
| 紧急 | 高 | 不予处理 |
| 不紧急 | 中 | 低 |

1. 优先级计算公式：(价值%) / (成本% \* 成本权值 + 风险% \* 风险权值)

第十四章

1. 领域模型图：3步
2. 顺序图
   1. 同步消息、异步消息、返回消息
   2. Opt没有else
   3. Alt只能选一个执行，可以有else
   4. Ref P42 P43
   5. Strict和seq的区别
   6. 例 P49 P51
3. 状态图
   1. 触发[监护]/行为
   2. 子图：P58-61
   3. 决策菱形、汇集点实心圆 P62、P63
   4. 中止和历史状态 P64
4. OCL
   1. P77 P78 P79 《invariant》{}
   2. P82例
5. CRC：Candidates Responsibilities Collaborators

第十五章

1. 避免歧义词汇

|  |  |
| --- | --- |
| 歧义词汇 | 改进方法 |
| 可接受的、足够的 | 具体定义可接受的内容，说明系统怎样判断“可接受”或“足够” |
| 大概可行的、差不多可行的 | 不要让开发人员来判断“大概”和“差不多”到底是否成立。应将其标记为待确定问题并标明解决日期 |
| 至少、最小、不多于、不超过 | 明确指定能够接受的最大值和最小值 |
| 在……之间 | 明确说明两个端点是否在范围之内 |
| 依赖 | 描述依赖的原因，数据依赖？服务依赖？还是资源依赖？等等 |
| 有效的 | 明确“有效”所意味的具体实际情况 |
| 快的、迅速的 | 明确指定系统在时间或速度上可接受的最小值 |
| 灵活的 | 描述系统为了响应条件变化或需求变化而可能发生的变更方式 |
| 改进的、更好的、更快的、优越的 | 定量说明在一个专门的功能领域内，充分改进的程度和效果 |
| 包括、包括但不限于、等等、诸如 | 应该列举所有的可能性，否则就无法进行设计和测试 |
| 最大化、最小化、最优 | 说明对某些参数所能接受的最大值和最小值 |
| 一般情况下、理想情况下 | 需要增加描述系统在异常和非理想情况下的行为 |
| 可选择地 | 具体说明是系统选择、用户选择还是开发人员选择 |
| 合理的、在必要的时候、在适当的地方 | 明确怎样判断合理、必要和适当 |
| 健壮的 | 显式定义系统如何处理异常和如何响应预料之外的操作 |
| 无缝的、透明的、优雅的 | 将词汇里面所反映的用户期望转化成能够观察到的产品特性 |
| 若干 | 声明具体是多少，或提供某一范围内的最小边界值和最大边界值 |
| 不应该 | 试着以肯定的方式陈述需求，描述系统应该做什么 |
| 最新技术水平的 | 定义其具体含义，即“最新技术水平”意味什么 |
| 充分的 | 说明“充分”具体包括哪些内容 |
| 支持、允许 | 精确地定义系统的功能，这些功能组合起来支持某些能力 |
| 用户友好的、简单的、容易的 | 描述系统特性，用这些特性说明词汇所代表的用户期望的实质 |

1. 优秀需求规格的特点
   1. 完备性
      1. 描述了用户的所有有意义的需求，包括功能、性能、约束、质量属性、对外接口等
      2. 定义了对所有可能的输入（包括无效的和有效的）所做的响应
      3. 为文档中所有插图、图表、术语等提供了完整的引用和标记
   2. 一致性
      1. 细节的需求不能和高层次的徐小秋相互冲突
      2. 同一层次需求之间也不能相互冲突
   3. 根据重要性和稳定性分级
   4. 可修改
      1. 可以对任一需求进行容易的、完整的、一致性的修改，同时不会影响现有的结构和风格
      2. 这就要求要求有着条理分明并易于使用的组织方式，包括目录、索引、显式的交叉引用
      3. 没有重复冗余
      4. 独立表达每个需求
   5. 可跟踪：
      1. 前向跟踪：找到需求的来源，例如和更早期的文档的显式关联
      2. 后向跟踪：能找到需求所对应的设计单元、实现源码、测试用例等，要求每个需求都有唯一的标识或可供引用的名称

第十六章

1. 需求验证：专指在需求规格说明完成之后，对需求规格说明文档的验证活动
2. 方法：
   1. 评审：由其他人检查，主要的静态分析手段，每一条需求都要评审
   2. 原型与模拟：涉及到复杂的动态行为时，成本较高
   3. 开发测试用例
      1. 如果无法为某条需求开发测试用例，那么它就可能存在模糊、信息遗漏、不正确等缺陷
      2. 例外：
         1. 排斥性需求：要求特定行为绝对不会发生
         2. 全局性非功能需求
      3. 测试用例套件
   4. 用户手册编制
      1. 验证功能需求
      2. 验证项目范围（没有实现的功能描述）
      3. 验证异常流程需求（问题和故障的解决）
      4. 验证环境与约束需求（系统的安装和启动）
   5. 利用跟踪关系
      1. 从业务需求到系统级需求。如果业务需求和用户需求不能得到后项需求（用户需求和系统级需求）的充分支持，那么需求规格说明就存在不完备的缺陷
      2. 从系统级需求到业务需求。如果不能根据跟踪关系找到系统级需求的前项用户需求和前项业务需求，那么该需求就属于非必要的需求
   6. 自动化分析

第十七章

1. 需求管理作用？
2. 需求管理活动：
   1. 维护需求基线
   2. 实现需求跟踪
   3. 控制变更
3. 需求变化的原因
   1. 问题发生改变
   2. 环境发生改变
   3. 需求基线存在缺陷
   4. 用户变动
   5. 用户对软件的认识变化
   6. 相关产品出现

**需要会画的图表**

解决方案表格

问题分析的描述表

上下文图

确定解决方案的约束表

目标模型图

ADM图

涉众的共赢分析图

涉众描述表

Power-Interest

Power-Attitude

Stakeholder/Issue关系图

涉众参与策略安排表

用例

**思考题和案例题**

第五章：

P30 问题分析：问题，问题表，业务需求，问题解决方案表，SF，用例图，

P39 问题分析

P40

P70 目标分析

P130之后 思考题

第六章：

P54 以后全都是

第七章：

P41 场景用例模型如何指导获取

P48 什么意思

第八章

P44往后

思考题

1. 十七章
   1. 思考题1：
      1. 看法：需求的变化是难免的。：
         1. 问题会发生变化
         2. 环境会发生变化
         3. 用户会发生变化
         4. 用户对系统的认识会发生变化
         5. 其他相关产品可能发生变化
         6. 需求基线可能存在缺陷
      2. 怎样做：
         1. 维护需求基线。是已经通过正式评审和批准的规格说明或产品，它可以作为进一步开发的基础，只有通过正式的变更控制过程才能修改。
         2. 实现需求跟踪。包括前向跟踪和后向跟踪。
         3. 需求变更控制
2. 十六章
   1. 思考题1：异同？
   2. 思考题2：
      1. 评审
      2. 原型和模拟
      3. 开发测试用例
      4. 用户手册编制
      5. 利用跟踪关系
      6. 自动化分析
      7. 怎么结合运用？
3. 十五章
   1. 思考题1：什么时候
   2. 思考题2：
   3. 思考题3：
4. 十四章：