第二章

1. 解答：

**业务需求处理方面**：目标不明确（1）；未处理前景和范围（1）；教训：A.应当根据业务需求，找到所有的涉众并协调他们的立场，限定问题的范围。由于目标不明确，所以首先需要发现用户的问题，而后理解并明确用户的问题。B.针对明确的问题，了解用户的期望目标，定义业务需求，同时提出可能的解决方案以及系统特性，并和参与各方达成一致，建立项目的前景与范围  
**涉众分析方面**：未识别和分析涉众（1），对上述进行具体展开（2）；教训，提及：涉众识别（课本第99页）（2）；涉众评估（课本第102页）（2）；共赢分析（课本第104页）（2）

1. 解答：

BR1：实现客户账单和收据的有效追踪；

BR2：实现产品特价时的利润和相关销售情况检查；

BR3：实现一个客户数据库。

1. 解答：
   * 先定义明确的业务需求，获得开发系统的必要性
   * 根据业务需求，协调涉众的立场，限定问题的范围
   * 根据系统项目范围，确定问题与的范围，在范围内选择用户并采集硬数据，确定需求获取的源头
   * 根据系统特性，确定获取的主题和内容
   * 根据获取源头和获取内容的特性，选择需求获取的方法
   * 围绕获取的内容，运用需求获取的方法，和涉众沟通（即向业务人员了解相关的业务知识，业务流程；再和销售人员沟通，由于他们的顾客是流动的，不确定的，只能通过销售人员间接获取来自于顾客的用户需求，了解他们的背景和习惯）
   * 根据业务需求对用户需求进行过滤和选择，得到充分必要的用户需求。
2. 解答：
   * UR1：用户可以根据系统的明确操作提示做出正确的反应；
   * UR2：用户插入银行卡后需要输入密码，得到验证后才可进行有效的具体操作；
   * UR3：在用户进入系统后，可以选择使用查询金额、存取现金、转账的功能；
   * UR4：用户能够正确、安全地退出系统。
3. 解答

SR1：（1） 系统显示用户插入磁卡的动态图像，正确标明插卡位置；

（2）用户根据提示，正确插入磁卡；

（3）系统读取磁卡卡号，界面显示输入密码的提示；

SR2：（1）对用户输入的密码，系统自动进行字符匹配；

（2）匹配正确的话，进入具体操作界面；

（3）匹配不正确的话，警告密码不正确，并提示再次输入；

SR3：（1）若用户选择查询金额图标和查询金额币种，系统读取银行数据库中用户对应的信息，反馈在用户界面上；

（2）若用户选择取款图标和金额币种及输入金额数目，系统读取用户请求，接受金额，修改数据库中该用户对应的信息，并提示成功与否；

（3）若用户选择存款图标和金额币种，系统弹出存款框，用户放入现金，系统接收现金并辨认真伪，并反馈存入金额数目，得到用户确认后，修改数据库中该用户对应的信息，并提示成功与否；

（4）若用户选择转账图标和金额币种并输入对方账号和转账金额数目，系统读取用户请求，修改数据库中所涉及到的用户的信息，并提示成功与否；

SR4：（1）用户选择退出图标；

（2）系统提示拔卡信息。

1. 性能需求：在用户点击图标后，系统在3s内作出反应。

质量属性：易用、可靠、安全、容错、可恢复、可维护。

约束：当用户输入密码次数等于3次后就不再提示输入密码，并自动锁定银行卡。

第三章

1. 解答：

还有其他一些活动，例如：过程管理活动和项目管理活动（人力、资金的管理）。

* + 过程管理活动包括需求工程过程的建立和需求工程过程的改进，又可细分为建立过程框架、选择工作组建、应用实践方法，以及过程的评价和改进。原因：需求工程拥有自己的生命周期和活动过程，需求的各项处理活动应该按照系统化、有组织和可重复的方式较之进行，所以应该建立严格的需求工程过程。
  + 项目管理活动获得资源支持、规划生命周期、团队管风险管理。原因：为了保证重要的工作得到应有的关注，能够顺利有序的完成，项目需要建立计划，并跟踪、监督和保证计划的正确执行。围绕着项目计划而执行的各种项目活动就是项目管理。

1. 解答：

不是，一般对于业务领域不熟悉的项目，需求是具有迭代性的，需要对业务领域的认知，有一个从认识到知识重构的过程。对于某些固定需求且熟悉的项目，开发人员已经相当明确用户需求，这时，就不需要返回到需求获取阶段以继续用户需求的获取，比如学校课程的作业软件工程实践电梯系统，就不需要迭代开发

1. 解答：
   1. 需求开发的迭代性指的是对于开发者对知识的认知水平在某一点上，发生重构，使得知识体系复杂性下降，而继续积累知识的过程。由于知识的重构，导致实际的需求开发活动之间（尤其是获取和分析）交织在一起，重构会不断地进行直到最终知识体系的建立。
   2. 软件开发的迭代性指的是在软件生命周期整体开发迭代，针对变更的需求或者新增的需求一种减少风险的开发模式。
   3. 需求开发迭代不会导致软件开发过程的迭代，但有时会有影响。（需求开发的迭代特性只是软件开发过程的迭代式开发的一个子过程，软件开发过程是一个相当庞大的工程，需要在软件开发过程的各个阶段都需要进行开发工作的迭代，当然也包括需求开发中的迭代。它们之间互相影响。如果需求开发中的迭代不能很好地完成需求分析任务，就必将影响到软件开发过程的其他迭代阶段的进行。）
   4. 增量式：需求->需求->需求->。。。->需求->开发。。。 无需求迭代
   5. 演化式：需求->开发->需求->开发->需求->开发。。。 有需求迭代
2. 解答：

没有影响。其实是需求开发过程的差异性一定程度上导致了细节知识的实践性。现实世界问题的复杂性和差异性主要导致了需求开发过程的差异性。如果所开发系统的类型不同，或者开发公司的规模和文化不同，或者系统开发资源获取的途径不同，需求开发过程都会表现出极大的差异。

第四章

1. 解答：

原因：

* 未有效地选择参与项目的用户

解决：需求工程师依据“个人领域知识”选择参与的用户，并通过涉众识别、涉众评估的过程一步步地甄选用户

* 用户和开发者认识不到用户参与需求工程的重要性

解决：用户是最终使用产品的人，是需求获取的主要信息来源。所以用户和开发者应给与“用户参与”足够的重视

* 用户情绪抵制，消极参与

解决：对项目反对的用户也是重要的需求来源，应该发现有抵制情绪的用户，分析其需求方向和关注点，找到他们抵制参与的原因，尽可能地减少用户的抵制情绪。化解冲突需要先发现冲突并分析原因，所以即使是情绪抵制的用户也要给与足够的重视，尽量在项目开始之初就在开发者和用户之间建立一支和默契的工作氛围

* 没有明确的用户

解决：利用常与用户接触的人作为用户替代源，代替用户表达他们的需求

* 管理上的障碍

解决：保持参与用户对项目的权威影响力，同时尽量维持用户的本职工作和项目参与这二者之间的平衡关系

第五章

1. 解答：

* 高层次的解决方案来自问题，因此应首先明确问题
* 找出明确的问题对应的期望目标，即业务需求
* 确定高层次的解决方案，对每一个解决方案要进行概要的描述，并描述其业务优势和将花费的代价
* 进一步明确该解决方案需要具备的功能特征，即系统特性。然后依据这些功能特征，分析解决方案需要和周围环境形成的交互作用，定义解决方案的边界
* 考虑解决方案的约束

1. 【工作陈述】解答：

项目范围定义了解决方案的概念和范围，同时也说明了系统不能提供的功能，定义项目包含的需求与不包含的需求之间的界线，列出涉众可能希望得到（潜意识里认为系统应包含的）、但不在产品计划之内的功能和特性，防止范围蔓延

风险：

* + - * 省略工作陈述的风险是不能明确项目的前景和范围。如果省略了工作陈述的话，你就不能和用户进行很好的沟通与交流，这样，项目的问题也就不能明确，即，开发人员无法与涉众对问题达成共识；无法明确问题，也就无法发现正确的业务需求，无法定义良好的解决方案及系统特性，继而无法明确项目的前景和范围，这样就会造成项目的不稳定甚至失败
      * 由于不了解用户希望替换现有系统的原因、现有系统在哪些方面不能达到用户的期望以及用户需求的范围，有可能导致开发者对需求的理解错误，进而导致由于范围蔓延不能按时完成，或者由于需求定义不良好不能给出符合用户期望的系统

必要性：

* 工作陈述能够帮助开发者与用户沟通并明确问题，进而对问题达成共识
* 与用户确定业务需求及解决方案的概念和范围，帮助用户建立对系统的现实的期望，防止范围蔓延而带来的风险（与期望不符、返工、超期、超出预算、项目失败）

1. 【拖欠债务】解答：

|  |  |  |
| --- | --- | --- |
| 问题 | 业务目标 | 高层解决方案和系统特性 |
| 账户数量增加，liz必须通读报告并且按照规则要求逐项检查客户数据，导致工作量加大 | 减少用户为确定严重拖欠债务的账户所付出的工作量 | 由软件来根据规则分析拖欠债务的账户的特征，找出候选后提交用户处理 |
| 必须查阅拖欠债务的账户三年内的历史资料，否则不能做信用决定 | 筛选并提供拖欠债务的账户三年内的历史资料以加快信用决定的速度 | 软件根据用户确认的拖欠债务的账户，筛选账户三年内的历史资料并交由用户处理 |
| Liz需要问题账户的比例但是报告中没有展示 | 在报告中增加展示问题账户的比例 | 软件分析并计算报告中问题账户的比例 |

1. 【福利选择】解答：

|  |  |  |
| --- | --- | --- |
| 问题 | 业务目标 | 高层解决方案和系统特性 |
| 她需要手工回答雇员的问题，而雇员对新计划的不信任导致她的工作量增大 | 减少顾问手工回答雇员关于新计划内容和费用的问题所耗费的工作 | 客观问题由系统自动答复，未明确的主观问题交由雇员处理 |
| 从信息系统部门获得工资和个人数据的时间太长，以至于阻碍她的工作 | 减少获得工资和个人数据的时间（1个月内减少50%），令顾问的工作顺利进行 | 软件提前从信息系统部门获取数据，雇员需要时直接通过软件获得 |
| 雇员数据太过分散，更新不正确导致存在冲突的数据，进而阻碍她的工作 | 集中雇员数据并保证正确更新，排除冲突数据，令顾问的工作顺利进行 | 由软件来分析雇员数据的各种特征，集中处理雇员数据，同时检查不正确的更新以及识别冲突数据，而后提交人工处理或自行解决 |
| 她建立投资和退休假定的计算复杂，导致时间太长 | 简化建立投资和退休假定的计算，缩短计算时间（1个月内减少50%） | 由软件来分析处理投资和退休假定的计算的复杂过程 |
| 雇员对她所提供的安全性不信任，担心保险计划泄露 | 提供安全性保障，避免保险计划泄露给未授权的人 | 软件加密用户信息，同时允许对应的雇员监控自己的信息 |
| 可变条件太复杂，导致她经常出错并且难以发现错误 | 简化可变条件，避免出错，即使出错也能被立刻发现 | 软件分析各个条件的特征从而简化条件，同时监控雇员的计算错误并立刻提示 |

约束：

1. 资源：资源限制在信息系统部门所提供的工资和个人数据以及用户的个人信息，不使用其他的任何资源
2. 安全性：样本假定结果只在系统内部保留，不能泄露给任何的未授权个人
3. 设备预算：必须使用现有的服务器和主机，尽量少地增加公司的额外开销
4. 系统的：不对公司的其他任何系统造成影响（包括信息系统部门）
5. 技术要求：使用面向对象的方法

第六章

1. 【思考1】解答：

* 用户的重要性：用户是最终使用和操作产品的人，他们是使用软件的目的是为了更好的完成自己的任务，满足组织的目标要求。因此，一个成功的软件要能够协助用户有效的完成实际工作，用户也就自然应该是需求获取的主要信息来源。需求工程师需要了解用户实际工作的开展状况和用户希望软件系统能够给予他们的帮助。
* 用户为中心的方法：用户参与是以用户为中心的设计方法的核心思想，它要求**开发者建立和用户的直接联系，尽早地关注与用户和用户的执行过程，通过及时获得用户的反馈来调整软件设计，以完成高质量的设计**。另一方面，用户参与就是反对通过和市场人员、管理者等中间媒介来了解用户。在以用户为中心的设计方法中，用户需要参与软件开发的全过程，并且对最终软件设计和质量具有非常重要的影响，所以在该方法中参与用户的选择和普通的涉众代表采样有所不同，要把他们区分开来。
* 涉众分析的过程：
  + 涉众识别：循环迭代式，直到涉众列表趋于稳定
  + 涉众描述：包括个人特征、工作特征，以及：关注点和兴趣，赞同和反对、期望、成为赢家的条件、受到项目的影响及内容、对项目施加的影响及内容
  + 涉众评估：
    - 优先级评估：用户群体分优先级；涉众分优先级（参与者——有影响且被影响、环境设定者——类似政府的有影响但是不被影响、被影响者——关注度高但是无影响、观众——专家）
    - 风险评估：弱支持者、强支持者、弱反对者、强反对者

需要将被影响者和环境设定者转化为参与者，同时将反对者转化为支持者

* + - 共赢分析：涉众之间、涉众自身的期望和项目的整体目标会有冲突
  + 涉众选择：
    - 代表采样：完整（每一类都有）、态度积极、数量适中（太少可能导致共同看法无法表达，太多导致不必要的分歧）、比例恰当
    - 参与策略：在合适的时间参加合适的工作
    - 用户替代原：当用户不明确或范围太广时，通过用户替代源获取用户的想法

1. 【思考2】解答：

* 建立良好的合作关系能够促进用户和开发人员就问题和解决方案达成一致，使得需求获取的过程更加轻松和完善，同时更好地指导需求分析、需求规格说明这两个过程，并且在需求验证和需求管理这两个阶段也需要用户的参与。
* 重视涉众识别和涉众描述的过程，充分地理解涉众，了解他们的工作特征和任务特征、态度、期望等
* 在进行涉众评估后，除了给与高优先级和强支持的用户以足够的重视之外，也应提高对被影响用户的关注，提高她们的优先级；同时给与反对者充分发表和实现自身意愿的权利，化解反对者的忧虑；还应平衡和折衷用户和负责人以及用户和用户之间的冲突与矛盾，实现共赢
* 在选择用户时，要选择态度积极的用户参与项目

1. 【Jeannine】

根本就没有进行涉众分析。作为涉众的高层管理人员、财务审计员、职员、下级经理在她买进系统后才提出反对，说明她之前没有与任何一个涉众交流过，涉众没有参与她的购买工作，她也完全没有获取涉众的期望和态度。

1. 【涉众描述】

* 个人特征：年龄、学历、职业、生活方式、技能、能力及限制
* 工作特征：任务、使用项目的程度和频率、技能和经验
* 地理和社会特征：国家、文化背景等
* 关注点和兴趣：（经理希望使用该系统所节约的费用必须大于使用系统的费用）
* 目标期望：（经理希望提高利润）
* 被影响程度：
* 力量程度：

忽略描述可能造成的风险：

优先级评估需要用户的任务特征（涉众个人特征和工作特征的描述可以帮助更好的确定功能需求）、对系统的影响和被系统施加的影响，风险评估需要考虑涉众的态度、关注点和兴趣取向（涉众的重要性、影响力、关注点和兴趣取向可以用来发现项目的潜在风险），而共赢分析需要考虑用户的目标期望（涉众的输赢条件和受影响程度可以帮助解决涉众之间的需求冲突）。如果忽略了对涉众的描述，那么涉众评估就失去了根据。一旦不能对涉众进行正确完善的评估，那么涉众的期望、影响就变得不可控制，整个需求工程、甚至是整个项目都面临崩溃

1. 【面谈对象】（完整采样、态度积极、数量适中、比例恰当）
2. 选择面谈对象的时候要完整采样，从生产、会计、营销、系统、物流各选择6名左右的用户参与面谈。高层管理均要参加面谈。因为在选择面谈的时候要力争均衡的收集用户的需求，因此要涉及各方面受系统影响的人。注意选择态度积极的用户，并且要涵盖对计算机技能熟悉以及不熟悉的人、对业务细节非常了解的专家和仅有普通工作能力的人。
3. 涉众类别完整可以保证所有被该系统影响以及影响该系统的涉众类型都有渠道表达自己的期望和需求，用户态度积极保证需求获取的有效和准确性，数量适中使得用户提供有用的信息，恰当的比例让开发者更全面的了解涉众的能力以保证功能设置更加合理、更符合用户实际。
4. 【硬数据】
   1. 定量硬数据：
      1. 业务明细表
      2. 收发货物的历史清单
      3. 业绩报表
      4. 成本统计表
      5. 拖拉机、仓库、办公室的使用维护情况表

定性硬数据：

1. 日常业务描述文档
2. 公司组织结构图
3. 工作指南和规章手册
4. 发/收货人伙伴关系文档
   1. 不能研究组织内的每一个文档和记录，需要使用抽样技术来获得样本硬数据集，这里使用分层抽样的方式

通过公式“样本大小 = 0.25 \* （确定性因子/可接受的错误）2”计算每一种硬数据的样本总量，假设为n，而后在这十五年的所有这一种文档中分层抽取，每一年抽取n/15个文档

第七章

1. 【问题】
2. 你的下属告诉我，你非常渴望有一台计算机。这是真的么？（诱导性问题）

你对使用计算机的态度是什么？

1. 我是这个领域的新手，我有没有忽略什么呢？（与上下文无关）

你还有要补充的方面和内容吗？

1. 你在销售计算中最常用的信息资源是什么，使用频度如何？（双筒问题）

你在销售计算中最常用的信息资源是什么？

最常用的信息资源的使用频度如何？

1. 其它销售经理认为，把一些月度销售商品放到Web上，然后做趋势分析，将会是一种主要改进，你同意他们的做法吗？（诱导性问题）

你认为把一些月度销售商品放到Web上然后做趋势分析的方式会是一种改进吗？

1. 没有比你现在使用的陈旧的方法更好的销售方案吗？（“陈旧”不合适）

还有比目前方法更好的销售方案吗？

1. 【数字钟面谈】

面谈目标：

找出公司会计进行财务分析的信息资料的来源以及信息格式

获得会计部门在财务方面的决策频度和分布规律

探讨会计所适应以及所期待的决策方法和样式

分析并获取会计的需求的信息性质

获取分析信息资源和最终做出决策之间的发展情况

* 1. 提前打电话或者发送电子邮件通知Harry，告知面谈内容，商定面谈时间和地点；提前通知可以给Harry时间去考虑面谈事宜。如果要进行一次深入的面谈，可以把问题通过电子邮件提前发给Harry，让他有时间仔细考虑答复。
  2. （学长的答案说因为他是领导专家所以用漏斗结构，我不同意）使用菱形结构。因为对方是业务专家，面谈目标非常明确专业，而且由“更新”二字可了解用户曾经使用过自动化会计功能，所以应该由一些明确的问题开始面谈，而后逐渐放宽并借此获取会计对待新/旧系统的态度等问题，最后落实到面谈目标上
  3. 应当面谈，因为Harry和其下属对软件系统的开发和应用具有的发言权和决定权不同，经由涉众分析可知他们属于不同的涉众类别。Harry属于领域专家，而其下属属于该系统的用户。下属和领导使用这个系统的目标不同，下属是为了更好的完成自己的任务，满足组织的目标，他们是主要的信息来源，所以应当面谈。
  4. 开放式问题：
     1. 请解释你是如何做财务决策的？
     2. 你对以前的自动化会计功能看法如何？
     3. 描述一下你们获取和分析信息资源的方式

原因：

由于面谈中可能会实现很多目标，涉及很多复杂问题，而这些内容除了来自语言外还来自被会见者的语调、面部表情、肢体语言，只有依靠人的灵活和主观能动性才能使面谈达到最优效果，所以面谈一般应该由人而不是电子邮件来来管理

1. 【Phil】

进行三轮面谈，具体安排如下：

* 第一轮面谈：

面谈对象：高层管理员

面谈结构：漏斗式结构

因为第一轮面谈主要是为了获得项目的前景和范围，通过探讨一些高层次的问题来和项目目标推导出业务需求，并根据问题帮助确定系统高层次的解决方案和系统特性，从而到了项目的前景和范围文档。而这种问题的讨论需要高层的管理员和对整个业务了解的人，所以第一轮的面谈对象是高层管理员。

根据面谈结构的特性，漏斗式的面谈结构适合于领导和专家这样的被会见对象，所以选择漏斗式结构。

* 第二轮面谈：

面谈对象：各个职能区域的职员

面谈结构：菱形式结构

第二轮面谈的目标是为了通过业务需求获取详细的用户需求。详细的需求涉及系统的各个层次，而各个层的工作目标和工作特性各不相同，所以需要要各个层次的工作人员进行面谈。

* 第三轮面谈：

面谈对象：全体成员

面谈结构：金字塔结构

第三轮面谈的目标是为了验证已获取的需求。分析汇总了获取的需求后，将获得的需求分类罗列后，根据具体不同的需求需要向各个层面的涉众确认验证已经获取的需求，保证需求的正确性，完整性，一致性。

由于需求已经基本确定，所以采取封闭式问题。

1. 【面谈报告】
   1. 目标中提到“看最新建议的系统的观点是否满足目标”，但是在面谈的要点和会见者的观点里面都缺失了这方面的信息
   2. 有关公司的所有历史与面谈的目标无密切关系，所以是无关紧要的
   3. 建议：
      1. 控制面谈的过程。面谈开始的时候可以通过例如谈公司历史来酝酿一下交流的气氛，但是不能偏离主题。如果长时间的谈论不相关的信息的时候，需求分析人员就可以委婉的提醒面谈对象，并重新切回正题。如果觉得要深入探讨某个问题时，可以鼓励被会见者举例说明并详加探讨，避免不能深入探讨事先准备的问题的情况再次出现
      2. 注意保持面谈的主题。针对每个面谈的目标，要在面谈的过程中安排合适的提示，逐一引导面谈对象对各个主题的叙述，且不要缺失对任何一个主题的讨论
      3. 通过控制过程来把握时间，使面谈时间限制在45min到1hour
2. 群体面谈：
3. 目标和内容的确定（1分）

群体面谈之前应先确定会议的目标和内容，设定主题和范围

1. 场地的确定（1分）；解释场地的条件：提供各种开会需要材料，会议室、道具、餐饮等（1分）

资金允许的话选择校外较安静的会议中心作为会议场地，不允许则使用学校会议室即可。但是无论怎样，会议室都应拥有充足的空间，有必要的道具（投影、白板、纸笔、打印机、计算机、网络），并且能够提供餐饮

1. 时间的确定（1分）；解释时间要求：全职的2～4天；（1分）

由于群体面谈要求参与者能够2~4天全职参与会议，不能分心，所以较合适的时间是学生放假之后并待所有教职工工作完毕之后。

在面谈正式开始之前，需求人员应根据主题拟定一份议程，参与者可以选择其相关的议题和会议时间参与讨论

1. 人员的确定（1分）；解释多涉众的共同参与（1分）

群体面谈是将所有的涉众代表集中起来共同开会以获取需求，要确保所有真正风险承担者的参与，所以需要分析哪些涉众能为整个过程作出贡献并且为保证项目成功必须满足他们的要求。除相关涉众（包括开发人员，学校教务人员，学校行政人员，教师代表，学生代表）外，还包括主持人、负责人、分析人员、记录人员、观察员

1. 会议准备（1分）；准备会议讨论材料（1分）；议程（1分）

事先确定面谈的主题和范围，确定会议的议程，建立需求的预期和会谈的目标，提前准备好所有可能会用到和能够加速会议进行的各种材料

第八章

1. 【原型】
   1. 原型方法的过程：

（1）确定原型需求（根据需要确定原型类型）：对较为清晰和确定的需求使用演化式原型，否则使用抛弃式原型，等需求明确后再行使用演化式原型。

（2）原型开发：根据已选定的原型方法进行开发，同时依据原型的需求内容选择原型构建技术

（3）原型评估（获得用户反馈）：将开发完成的原型交由评估者（用户，技术细节需要开发者）评估，评估者观察、感受、实际使用原型，产生反馈（肯定、否定、修改建议）

（4）原型修正（定义所得需求）：根据评估者的反馈迅速调整错误的或不完善的想法，在连续的反馈和调整之中逐步接近正确的和完善的需求

* 1. （自己乱猜的，主题是“隐含知识”和“表述需求时的主观加工”）帮助用户：

首先初步获取用户的需求。此时的需求如果具有某种意义上的不确定性——一般表现为实现最终系统的隐含知识有所缺失以及表述需求时的主观加工，则会导致用户难以理解和阐明自己的需求。需求工程师应根据初步需求确定需要开发的原型类型，继而进行原型开发；开发完毕后请用户评估原型，同时鼓励用户操作原型，在操作的过程中开发者可以观察到用户的反应，并且用户可以通过对原型的操作发现需求中的部分隐含知识以及自己或者其他用户对需求的主观加工，而后据此提出反馈意见。在对原型的不断精化的过程中，用户会逐步理解自己的问题和需要，并能够做出精确的阐述。

1. 【是否使用原型】
   1. 原因：

* 原型仅仅是开发当中使用的一种手段，它利用得当可以加速开发的进程，但不能代替软件开发中的所有工作。它在软件开发中被当作是一种工具，可以在开发的各个阶段使用以达到该阶段的目的，但是软件工程的全局不能改变
* 原型开发最大的缺点就是：成本太高，高的让人难以接受。所以原型方法只在必要的时候使用原型方法。通常来说，如果用户需求出现了模糊，不清晰，不完整等具有一定不确定性的特征，就可以考虑使用原型方法。
* 原型方法的复杂性使得它会给项目引入了新的风险：
  + 当涉众看到了一个正在运行的原型时，就会以为产品几乎已经完成，从而提出快速交付产品，这会打乱本来的协议和计划。如果整个软件工程的过程都使用原型的话，则无疑会深化这个风险
  + 由于原型的不完全性，它所表现出来的非功能特性通常十分良好，不仅容易导致用户忽略功能特性，还可能使得用户对最终产品产生不切实际的期望
  + 由于开发原型需要耗费大量的时间和预算，直接为一切设计原型势必导致所有的需求——无论是否明确——都要经历原型过程的迭代，这样不仅不会加快速度、节省资金，还会造成时间和金钱的浪费
  1. 情形：（探索式——模糊的需求）（实验式——清晰的需求，开发者无把握）（演化式——清晰的需求）
     + 创新性产品，产品以前从未存在过且难以可视化（探索式）
     + 用户对该产品和技术没有经验，无法明确工作的具体细节，细节需求不确定（探索式）
     + 用户在完成工作的方式上存在障碍，无法判断解决方案是否可行，产品在整体方案的可行性上不确定（实验式）
     + 用户无法清晰地表达自己的需求（默认需求和潜在需求）（探索式）
     + 需求工程师在理解用户的需求时存在困难，在澄清之前需求是不确定的（探索式）
     + 需求的可行性值得怀疑，需求的可满足性不确定（实验式）
     + （演化式）将不断传递清晰的需求和原型资产到下一个原型开发过程的活动作为项目开发开发的一部分，并使得原型资产逐步成为最终的软件系统

1. 【是否观察】
   1. 原因：
      * 观察用户反应：可以发现很多关于原型的有用观点，包括他们是否满意，以及出售或者实现这个系统是否存在困难。比如：观察用户在操作时自然的寻找倾向和寻找目标，可以发现原型的功能设计和用于的自然倾向不符合的地方；关注用户较容易出错的地方，因为这里可能需要重新设计；用户迷惑的地方可能缺失一定的内容和功能
      * 观察用户建议：用户在潜意识里对自己的需要有一定的方向，所以这些建议可以帮助开发者改进、改变或调整原型，从而使原型更接近目标
      * 观察创新思想（开发原型前没有想到的功能）：用户不知道自己需要什么，但是这些需求是客观存在的，所以我们需要通过原型->观察/反馈，让用户逐渐知道自己需要什么
   2. 只有经理“确定原型需求->原型开发->获取反馈信息->原型修正”这一过程的原型活动才能被认为是完整的。如果系统的某部分已经被原型化，但是在后续系统中没有考虑用户的反馈信息，这个原形都不能算是一个符合要求的原型。这样会导致开出来的原型根本就不符合用户需求，开发出来以后用户不满意可能会受到用户的抵制。可能在后期才被发现，开发方需要做很大的调整修改，导致项目延期，严重者可能会导致项目的失败
2. 【直接使用原型】
   1. 误解：
      * 这则报道中提到“我们的配送信息系统原型一发布就投入使用了”可以看出作者误解了一点：开发出的原型不是最终的软件，而是最终软件的一个局部，原型不能直接发布使用。我们使用原型的目的是获取需求的内容、解决需求的不确定性，而不是直接产生可交付的产品。即便是最终会演化成产品的演化式原型，也需要经历整个项目开发过程。
      * 作者还说“绝对没有修改的必要”这句话显然有问题，原型开发的过程中要不断地根据评估者的反馈进行原型的修改，调整完后还要准备再次原型评估，如果不能通过，则再根据反馈、观察进行原型修正，所以不能说“绝对没有任何修改的必要”
   2. 问题：“绝对没有修改必要的原型“没有经历反馈和修正的过程，甚至都不能算是完整的原型开发，更别说是完整项目开发了。一次开发的原型意味着原本存在的不确定性依然保持，且由于原型只是系统交互、系统功能的体现，它没有经历大范围、长时间的测试，无法保证非功能需求的实现（质量需求，包括可靠性、可维护性等）。过多的漏洞会导致用户抵制该系统，可能会严重影响后期业务的进展，甚至导致项目最终失败。

第九章

1. 【情景性事件】
   1. 情景性事件是只有和它们发生时的具体环境联系起来、放在发生时的情景中进行解释，才能得到理解和明确的事件，具有突现性、局部性、暂时性、涉身性、开放性和模糊性
   2. 观察方法将发现的重点放在问题的上下文环境之上，包括组织的文化、组织的结构、用户的工作环境、用户的工作实践、法律与经济约束等。通过对上下文环境的理解，观察方法可以帮助需求工程师更好地理解问题发生的情景，进而更透彻的理解情景性问题
2. 【采样观察】
   1. 时间采样：需求工程师建立指定的时间间隔来观察用户的活动情况
      1. 优点：减少在任何某个单独时间段内进行观察时可能发生的偏差，将偶尔才发生的事件看作是重要的业务事件；可以只选取频繁发生的活动中一个代表样本进行观察，节省时间和成本
      2. 缺点：以分段方式收集观察的数据，无法为某些长事件提供充分的观察时间；一些很少发生但又非常重要的事件可能得不到观察，因为它们没有出现在采样时间之内，比如非常重要、影响很大的异常处理事件
   2. 事件采样：通过有目的的选取整个事件进行观察，而不是随机采样时间段
      1. 优点：为观察所提供的是在一个真实背景下的完整行为，所以不会遗漏重要的事件或者重要事件的某些片段
      2. 不能获得频繁发生事件的代表性样本
3. “我知道你有很多材料。那些材料里到底有什么？”Betty Kant问道，她是MIS特别工作组的负责人。MIS特别工作组是你的系统团队联络Sawder家具公司的桥梁。你拖了一大堆材料，正准备离开这栋楼。

“哦，是过去6个月的一些财政决算、生产报表，还有Sharon给我的一些业绩报表，业绩报表涵盖了过去6个月的目标和工作业绩。”你在回答时，有些纸掉到了地上，“你为什么问这个问题呢？”

Betty为你拾起纸并把它放到最近的桌子上，回答道：“因为你根本不需要这些垃圾。你来这里要做一件事情，就是和我们这些用户谈话。从这些材料中得不到任何有益的信息。”

(1) 只有告诉Betty你从每份文档中找到的东西才能使她相信每份文档都是重要的。用一段文字解释文档为需求工程师提供了什么帮助？

(2) 在你和Betty谈话的时候，意识到实际上也需要其他的定量文档。列出你缺少的东西。

解答：

1. 解释了文档含有问题域知识（2分）；说明了文档内知识的结构良好性（2分）；列出文档的示例（1分）；

文档分析是通过检查采集的硬数据来确定潜在的需求，是从旧的工作所使用的材料当中挖掘新的需求，在寻找将成为新产品一部分的需求。通过分析组织的硬数据，可以获取组织业务的问题与信息、得到组织的业务工作流程、发现业务细节中存在的问题。一个有经验的需求工程师会从文档中获取事实，理解问题域;

文档具有良好的知识结构。定量硬数据会以表格和流程的形式体现组织的信息流、业务目标以及业务执行状况，如生产报表和业绩报表，它们可以定量地描述组织的生产情况和业绩信息；定性硬数据会以文本形式描述组织的业务、流程、目标、结构等信息，如财政决算等，它可以描述组织在财政方面的决策情况

1. 按照逻辑给分：逻辑上正确的，每个1分，最多5分。

（定量文档不是定性文档）

生产成本统计表

销售情况统计报表

售后分析报表

财务数据及重点财务指标分析表

创新方案及业务目标统计报表

产业垃圾处理表

第十章

1. 【场景】
   1. 需求获取：（探索性场景）将场景方法与面向目标的方法结合起来，使用目标结构来组织场景，从而指导需求获取活动的开展。获取活动结束后，使用场景可以组织需求获取得到的信息——将得到的信息整理未对一个或多个场景的描述（描述性场景）
   2. 需求分析：（探索性场景）场景可以作为详细需求分析的信息基础，需求分析活动从场景中抽象出需求模型
   3. 需求规格说明：（描述性场景）场景将获得的场景描述文档化，产生系统的需求规格说明
   4. 需求验证：（解释性场景）场景可以在需求验证阶段被用来说明原因和可行性

从软件工程的角度来看，场景方法可以在需求工程的各个阶段发挥功效，而需求阶段获取的需求模型和需求规格说明书是后续设计和实现的基础，也是场景方法功效的延续。而测试阶段也会用到场景方法，如利用需求工程获得的场景描述来编写系统测试用例和验收测试用例。

第十一章

1. 【优先级方法】
   1. 累积投票：评定人员被给与一定数量的投票分数，而后根据自己的判断将分数分配给各个单独的需求，最后汇总排序（优点：将评定人员所认为的需求的重要性关系量化，在客观打分的前提下能够精确地排定需求优先级；缺点：由于分数的分配权掌握在评定人员手中，所以可能会将主观因素放大，比如导致某用户给与与自己相关的需求以极高的优先级）
   2. 区域划分：将评价优先级的特征分成等级而后划分区域，最后由评价人员将需求划分到不同的区域中（优点：最后的区域划定是由评价人员协商决定的，可以避免主观因素，而且需求所在的优先级区域也能充分解释它的重要性原因；缺点：评价是粗范围的，只能在大体情况下将需求分类，而属于同一区域的优先级不好判定）
   3. Top-N：在每次迭代之前，由评价人员选择他们认为最重要的N个需求（N的数值由实现这N个需求所要花费的代价决定）（优点：保证了每次迭代所实现的都是当前阶段最重要的需求，而且可以将实现代价限制在预期估计之内）（缺点：无法甄别需求的重要性原因，而且非量化的方式也很难在这N个需求内区分优先级）
   4. 数据量化：将评价优先级的特征量化，再根据一定的计算规则计算最终的需求优先级（优点：在客观量化的前提下结果精准，而且也能充分展示需求的各种特征）（缺点：评定优先级耗费较多时间）
2. 【边界定义】

系统边界通常是进行需求分析的起点，无论是结构化分析还是面向对象分析，都要从系统的边界图开始，逐一分析和细化系统与外界的交互，以保证最终产品的行为能够和环境形成互动，以满足用户的需求。边界是软件和外界互动的地方。解系统为自己做定位，首先要分析互动的反应。在需求分析的后期阶段，才会逐渐进入内部分析。

软件要完成用户的任务需要和外界协调互动，经过问题分析之后一般可以得到高层次的解决方案及系统特性。而一个系统通常会有很多高层次问题，虽然问题分析之后可以得到解系统为了解决某一问题而需要具备的知识片段，却无法将这些片段自动连接为整个系统的概要全图，所以很有必要将各个问题的分析结果进行综合与处理，已确定整个解系统的功能，建立系统的边界。

1. 【结构化技术】

|  |  |
| --- | --- |
| 技术 | 数学基础（上课笔记） |
| 上下文图、数据流图、微规格说明、数据字典 | 形式化方法，图灵机 |
| 实体关系图 | 关系代数和关系演算 |
| 功能实体矩阵 | （来自于状态转移图和数据流图，说明功能与实体的关系） |
| 状态转移图/矩阵 | 有限状态机思想 |

1. 【面向对象】

面向对象的技术【是】对结构化分析技术的继承和借鉴，它们没有数学基础

|  |  |
| --- | --- |
| 面向对象技术 | 所来源的结构化技术 |
| UML活动图 | 数据流图->业务过程模型->活动图 |
| UML类图 | 实体关系图->OMT->类图 |
| UML交互图 | （工业界方法） |
| UML用例图 | （工业界方法） |
| UML状态图 | 状态转移图/矩阵->状态图 |
| 对象约束语言 | （复杂的业务规则） |

1. 【事件/事物】

**二玉哥哥语**：事件就是行为，就是DFD；事物就是数据，就是ERD。

【结构化分析】结构化分析的两条路径——DFD（数据流图）一个是ERD（实体关系图）——分别来自于事件和事物。由发现业务事件、描述业务事件、细化业务事件的流程和思路可以进行过程建模，产生数据流图和微规格说明；由发现业务事物、描述业务事物、关联功能与数据的流程和思路可以进行数据建模，产生实体联系图和功能实体矩阵

【面向对象分析】没有在结构化分析中那么重要，因为面向对象的核心是多对象协同，多对象协同既不是事件，也不是事物。在面向对象分析中，事件就是行为，事物就是数据。事物可以帮助产生类图（来自ERD），而事件可以帮助产生活动图（来自DFD），而对产生除此之外描述多对象协同的UML图帮助不大

1. 描述一下，面向对象方法当中，从需求开发的一开始到结束，技术模型的选择和使用过程，详细地看一下\*\*\*。最开始的时候使用了什么技术，描述了哪些内容；后续使用了什么技术，描述了哪些内容，怎么衔接的

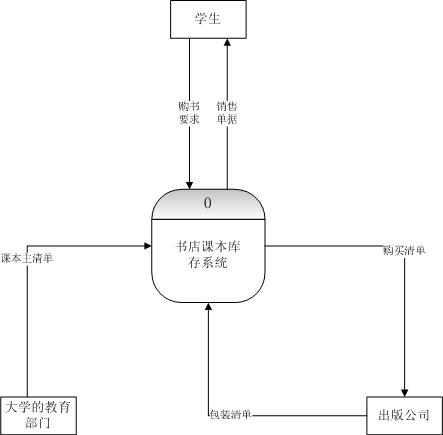
【Wieringa框架】从外到内，从全局到分解，功能式->通信式->行为式

1. 外部功能：系统用例图，系统对外交互的功能式描述
2. 外部通信：上下文图、用例图和交互图，系统对外交互的通信式描述
3. 外部行为：过程依赖图和交互图，系统对外交互的行为式描述
4. 概念组元：DRD、ERD、功能实体矩阵、类图、数据字典，描述交互所涉及的系统或者系统部分的分解关系
5. 组元功能：对象约束语言、微规格说明，系统分解后的系统内部交互的功能式描述
6. 组元通信：DFD、功能实体矩阵、过程依赖图、交互图，系统分解后的系统内部交互的通信式描述
7. 组元行为：实体生命历史、活动图、状态转移图/矩阵，系统分解后的系统内部交互的行为式描述

第十二章

1. 【上下文图】上下文图是DFD的第一个层次
   1. 外部实体：矩形
   2. 过程：圆角矩形，上部为ID（必须有输入和输出）
   3. 数据流：箭头（必须是过程的输入或输出，是事物）
   4. 数据存储：平行线

【重点】上下文图将整个系统看作是一个过程（只有一个编号为0的圆角矩形），有所有跟系统交互的外部实体以及系统的输入输出。没有数据存储。



1. 【决策表】

|  |  |
| --- | --- |
| 条件和行动 | 规则 |
| 条件声明（变量，有几个写几行） | 条件选项（取值，要取变量取值的组合，而后合并） |
| 行动声明（采取的动作，有几个写几行） | 行动选项（行动在每个条件下的选项——决策） |

存在使老师无法采取行动的条件，当初始期末成绩为F，并且旷课多于3堂时

改正错误：添加条件，当初始期末成绩为F，旷课多于3堂时，不做降级处理

（如图，这是个数学问题）

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 条件和行动 | 规则（10个） | | | | | | | | | |
| 头三次作业和测试总成绩 | >=90 | >=90 | >=80并<90 | >=80并<90 | >=70并<80 | >=70并<80 | >=60并<70 | >=60并<70 | <60 | >=0并<=100 |
| 第四次作业成绩 | >=70 | >=70 | >=70 | >=70 | >=70 | >=70 | >=70 | >=70 | >=0并<=100 | <70 |
| 旷课堂数 | <=3 | >3 | <=3 | >3 | <=3 | >3 | <=3 | >3 | 任意 | 任意 |
| 成绩A | X |  |  |  |  |  |  |  |  |  |
| 成绩B |  | X | X |  |  |  |  |  |  |  |
| 成绩C |  |  |  | X | X |  |  |  |  |  |
| 成绩D |  |  |  |  |  | X | X |  |  |  |
| 成绩E |  |  |  |  |  |  |  | X |  |  |
| 成绩F |  |  |  |  |  |  |  |  | X | X |

1. 【数据字典】
   1. String = 1{[‘a’-‘z’| ‘0’-‘9’| ‘A’-‘Z’]}n (n正无穷)

1｛｝n表示内部的结构循环，最少一次，最多n次

[ | ]表示内部多选一

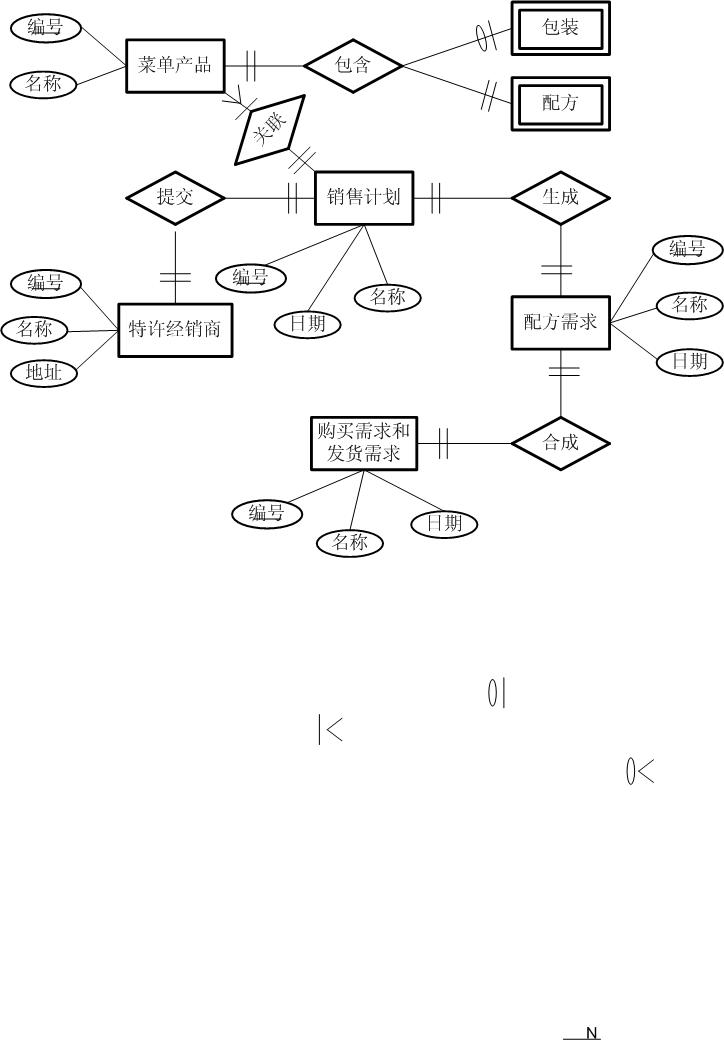
* 1. 32位Integer是0~232-1

Integer =

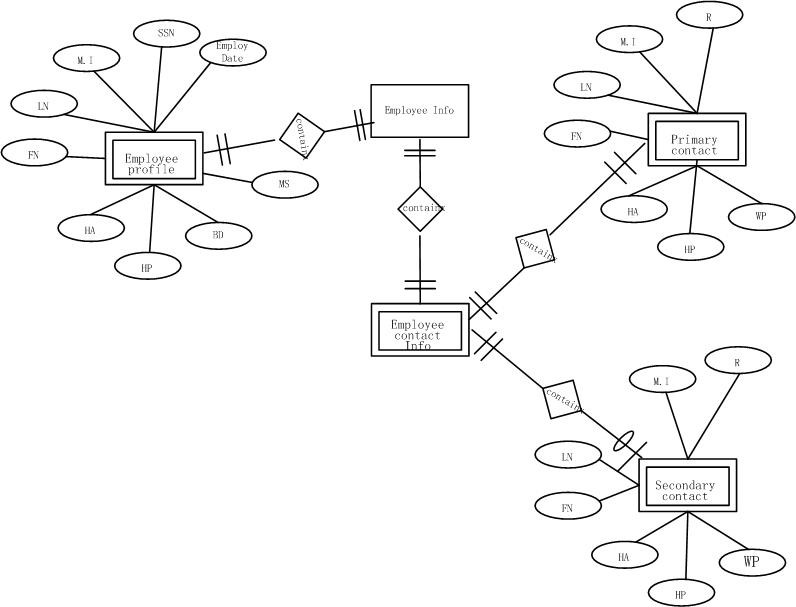
* 1. Date =

第十三章

1. 【ERD】
   1. 实体：矩形
   2. 关系：菱形
   3. 属性：椭圆
   4. 基数：（表示实体最少、最多的参与个数，最少为1没有圈，最少为0有圈）



1. 【数据模型和属性】
2. 【ERD】



第十四章

1. 【行为模型】行为模型有哪些不同的技术，它们在适用情境上（单/多个用例，单/多个对象，业务/流程）有什么区别？
   1. 交互图：对单个用例的典型场景的实现，描述在特定上下文环境中一组对象的交互行为，适用于事务性业务工作的表示
      1. 顺序图：突出消息的时间顺序（纵轴表示时间）
      2. 通信图：突出交互中协作对象之间的关系
   2. 状态图：以状态机模型的方式进行用例实现，可以用来实现单个用例，也可以用来实现多个用例，还可以从多个用例中抽取出和重要类相关的行为进行集中表示，主要用来描述对象的行为
   3. 活动图：依据处理流程进行的用例实现，以复杂的工作为主线表述多个用例的综合活动过程，描述组织业务流程（注意与顺序图的区别）
2. 【建模活动】面向对象分析的主要建模活动（整个14.7）

* 建立领域模型
  + 发现对象和类
  + 建立类之间的关联
  + 添加类的重要属性
* 建立行为模型
  + 建立系统顺序图（用户 --- 系统）
  + 在需要时
    - 建立详细的交互图（详细的顺序图）
    - 建立用例的状态图或者重要类的状态图
    - 建立活动图
  + 为系统行为添加契约说明（不变量、前置条件、后置条件）

1. 【样本】样本对面向对象建模的好处

* 在建立领域模型时，组织的业务范围文档等样本可以帮助开发者发现对象和类，建立概念类
* 在建立行为模型时，企业的业务描述文档和报表可以帮助开发者建立系统顺序图
* 企业的业务流程文档可以帮助开发者建立业务过程模型，继而建立活动图
* 企业的业务规则、约束之类的文档可以帮助开发者为系统行为添加契约说明

总之，企业的样本/硬数据在面向对象建模的过程中发挥了极大的作用

1. 比较包括CRC策略在内的各种对象与类的发现方法，说明各种的优缺点和适用场景。
2. 概念类列表：事先给出一个概念类的分类列表，然后由分析人员在需求信息中寻找相应类别的候选对象，最后对候选对象进行确定和归纳，形成概念类
   1. 缺点是不系统，不利使用（1分）
   2. 优点是简单，可用于简单情况的系统建模（1分析）
3. 名词分析：是一种运用语言分析的实用方法，名词分析从文本描述中识别出有关的名词和名词短语，然后将它们作为候选的对象，最后对候选对象进行确定和归纳，形成概念类
   1. 缺点是数据驱动（1）：数据驱动要求所有具有给定特性或共同特性集的实体组成一个类，但是现实中依据相似性分类的情况较多（指责驱动）
   2. 优点是适用于数据库依赖比较大的信息系统的建模（1分）；
4. 行为分析：行为分析师从需求描述中搜寻动词，识别出系统行为，然后找出系统的主动对象和被动对象作为候选对象。找出候选对象后，就按照对象的含义进行对象的确定，针对确定后的对象，以其发起行为的组合作为特征描述，并以特征的相似性进行归纳分类，产生概念类
   1. 缺点是缺乏对数据的处理，仅仅分析了行为（1分）
   2. 优点是适用于行为控制比较复杂和关键的系统建模；

（以上方法适用于简单情况，在需求较为明确、文档组织良好的前提下，它们能够发挥很好的作用。但是在复杂情况下很难起到好的效果）

1. CRC：基于CRC可以建立一种索引卡片，CRC卡。每个卡片代表了一个被发现的候选对象。卡的背面是关于候选对象的非正式描述，卡的正面记录了对象的职责（所维护状态和可以执行状态的行为）和协作者。CRC卡简洁方便可以随时被移动、修改或者丢弃，所以它特别适合在复杂的系统中充当进行对象的发现和设计思想的挖掘，即进行复杂情况下的面向对象的分析和设计
   1. 缺点是缺乏系统化支持，需要较多的经验（1分）：基于CRC的指责驱动方式较为复杂，需要更多的实践经验和技巧
   2. 优点是可以处理复杂情况下的系统建模（1分），而且明确支持职责驱动（1分），适用于需求不稳定、经常出现变更的情况

第十五章

1. 【读者】读者、阅读目的、要求
   1. 项目管理者：
      1. 目的：基于SRS进行软件的估算
      2. 要求：全面、准确地定义了软件的功能和非功能要求
   2. 设计人员和程序员
      1. 目的：依据SRS完成自己的任务
      2. 要求：文档中包含的功能性、非功能性要求具有可验证性（文档的内容是其工作是否正确的一个重要判断标准）
   3. 测试人员
      1. 目的：根据SRS设计测试计划，包括确定需要测试的功能和产生有效的测试用例的方法，而后指导测试活动的进行
      2. 要求：对功能性和非功能性要求的描述要准确、深入，并且包含功能的场景描述
   4. 文档编写人员
      1. 目的：按照SRS计划用户使用手册的编写，确定手册的内容和要点，在开发完成后进行最终手册的编写
      2. 要求：SRS准确的定义和描述了软件的对外接口，以及与外界的交互
2. 【原则和技巧】
   1. 原则：
      1. 写作是一门艺术（组织方式、情景处理、写作技巧、容易出错的地方）
      2. 文档化的目标是交流
   2. 技巧：
      1. 内容的组织：所有内容位置得当；引用或强化，但不重复
      2. 表达方式：形式依赖于内容；使用系统的表达方式
      3. 细节描述：定义术语表或数据字典；避免干扰文本；避免歧义词汇
3. 在需求获取和需求分析中采用哪些手段可以保证最终需求集的完备性、一致性和正确性
   1. 完备性：
      1. 描述了用户所有有意义的需求，包括功能、性能、约束、质量属性和对外接口
      2. 定义了软件对所有情况的所有实际输入（无论有效输入还是无效输入）的响应
      3. 为文档中的所有插图、图、表和术语、度量单位的定义提供了完整的引用和标记

手段：

* 为避免需求的遗漏，需求工程师要做好业务需求的获取与分析，建立并控制正确的项目范围。
* 建立业务需求、用户需求、系统需求的跟踪关系
  1. 一致性：
     1. 细节的需求不能同高层次的需求相冲突
     2. 同一层次的不同需求之间也不能互相冲突

手段

* 在需求获取时，各涉众对于某一问题的期望的冲突和矛盾要及早解决，采取分析共赢机会、折衷各方期望的措施，在各个确定的需求上达成一致
* 需求分析阶段所建立的用例模型等代表系统功能的模型需要被评审
* 开发人员和非开发人员对SRS进行手工评审
  1. 正确性：
     1. 保证文档中每个单一需求都是优秀的需求，正确地描述所需要的系统功能，真实地反映用户的意图

手段

* 在对业务需求和用户需求进行获段时就严格记录用户的意图，不能曲解，如有不理解的需求则需要标记
* 在需求分析后、需求提交给开发人员之前，必须请需求的提出者进行确认

第十六章

1. 多种需求验证的方法应该如何结合使用

* 需求验证的方法：需求评审（静态分析，需求验证的一种主要方法）， 原型与模拟，开发测试用例，用户手册编制，利用跟踪关系，自动化分析
* 每个需求都需要经过评审，对于动态行为评审不能完成的就要通过原型和模拟的方法来验证。在正常的工作当中，可以顺便用上用户手册，测试用例，跟踪等方法在一些错误之处或者一些需求上进行验证，也是比较有效的。总而言之，大多数情况下，需求都是在静态的方式下被加以验证的（评审的方法），也可以说几乎说的需求都要经过评审的方法进行验证，个别动态复杂的需求需要用原型与模拟的方法进行验证，工作之间产生的衔接可以用上开发测试用例，用户手册等方法，这样可以实现高效的综合运用