需求工程实践精要

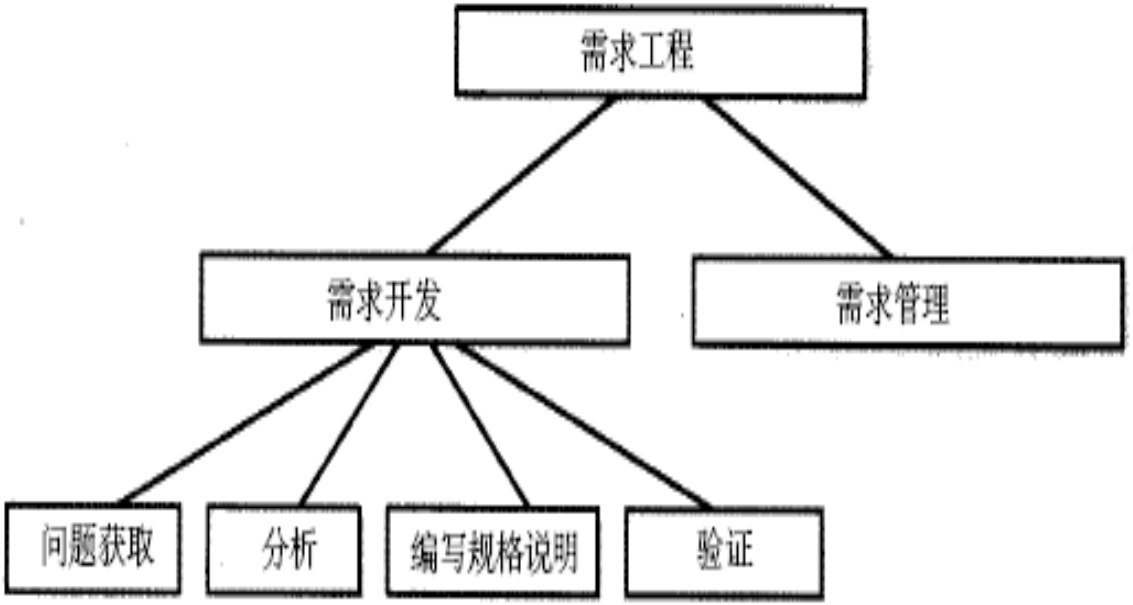
高级咨询顾问 任甲林
麦哲思科技（北京）有限公司

报告人

- 任甲林
  - 高级咨询顾问，主要从事提升软件研发能力的培训与咨询
  - 联系方式：
    - Mobile：13301172719
    - E-mail：renjialin@measures.net.cn
    - MSN: dylan_ren@msn.com
  - Blog:
    - http://dylan1971.blog.ccidnet.com/
  - 工程经验：
    - 93年从事软件开发,参与了50多个项目的开发
    - 曾为北京汉王、上海鹏开、深圳富士康、深圳鹏开、四川长虹、大连华信、成都虹微、昆山中创、北京信城通、南京诚迈、南京润和、珠海矩力等多家公司咨询

内 容

- 需求的基本概念
- 需求获取
- 需求分析
- 需求描述
- 需求验证
- 需求管理



讨论

- 你在需求的工程实践中，困扰你的问题是什么？

常见问题

- 甲方：
  - 客户需求本身不明确
  - 开始提不出需求，软件做完后才提出需求
  - 合同没有约束力，超出合同范围
  - 客户之间互相扯皮，没有人拍板需求
  - 客户方的需求负责人变更后，需求也变更
- 甲乙双方；
  - 随着时间的推移，客户的业务发生变化，导致需求变化
  - 双方沟通有失误，对需求存在误解
- 乙方：
  - 不知道如何引导客户提出详细的需求
  - 遗漏了客户的一些需求，比如操作习惯等
  - 编写的需求客户看不懂，客户不看到实际系统无法确认需求
  - 需求分析人员故意隐瞒了一些隐含的需求，得过且过
  - 开发人员对需求的理解比较差

Measures

需求的基本概念

需求的重要性

Measures

- 需求错误导致的成本是修复程序错误成本的100倍。(Boehm 1981;Grady 1999)
- 更正需求阶段发现的错误平均仅需要花30分钟，纠正系统测试期间发现的缺陷则需要花5-17个小时。(Kelly、Sherif、Hops 1992)
- 需求缺陷约占软件系统全部缺陷的1/3，是系统开发中最常见的缺陷(Capers Jones(1994)
- 需求缺陷可能消耗整个项目费用预算的25%－40%
- 预防缺陷所花费的1美元可以为修正缺陷节约3-10美元的费用。 (Capers Jones(1994)

需求—导致项目失败的罪魁祸首

Measures

- 根据Standish Group对23000个项目进行的研究结果表明，28%的项目彻底失败，46%的项目超出经费预算或者超出工期，只有约26%的项目获得成功。
- 而在于这些高达74%的不成功项目中，有约60%的失败是源于需求问题。
- 也就是说，有近45%的项目最终因为需求的问题最终导致失败。

对不知道航行目的地的人来说，没有顺风！

我们在哪里重重摔了一跤

Measures

- 在Standish Group的报告中总结了导致项目失败的最重要的8大原因中，有5个与需求相关：
- 不完整的需求(13.1%)；
- 缺乏用户的介入(12.4%)；
- 不实际的客户期望(9.9%)；
- 需求和规范的变更(8.7%)；
- 提供了不再需要的(7.5%)

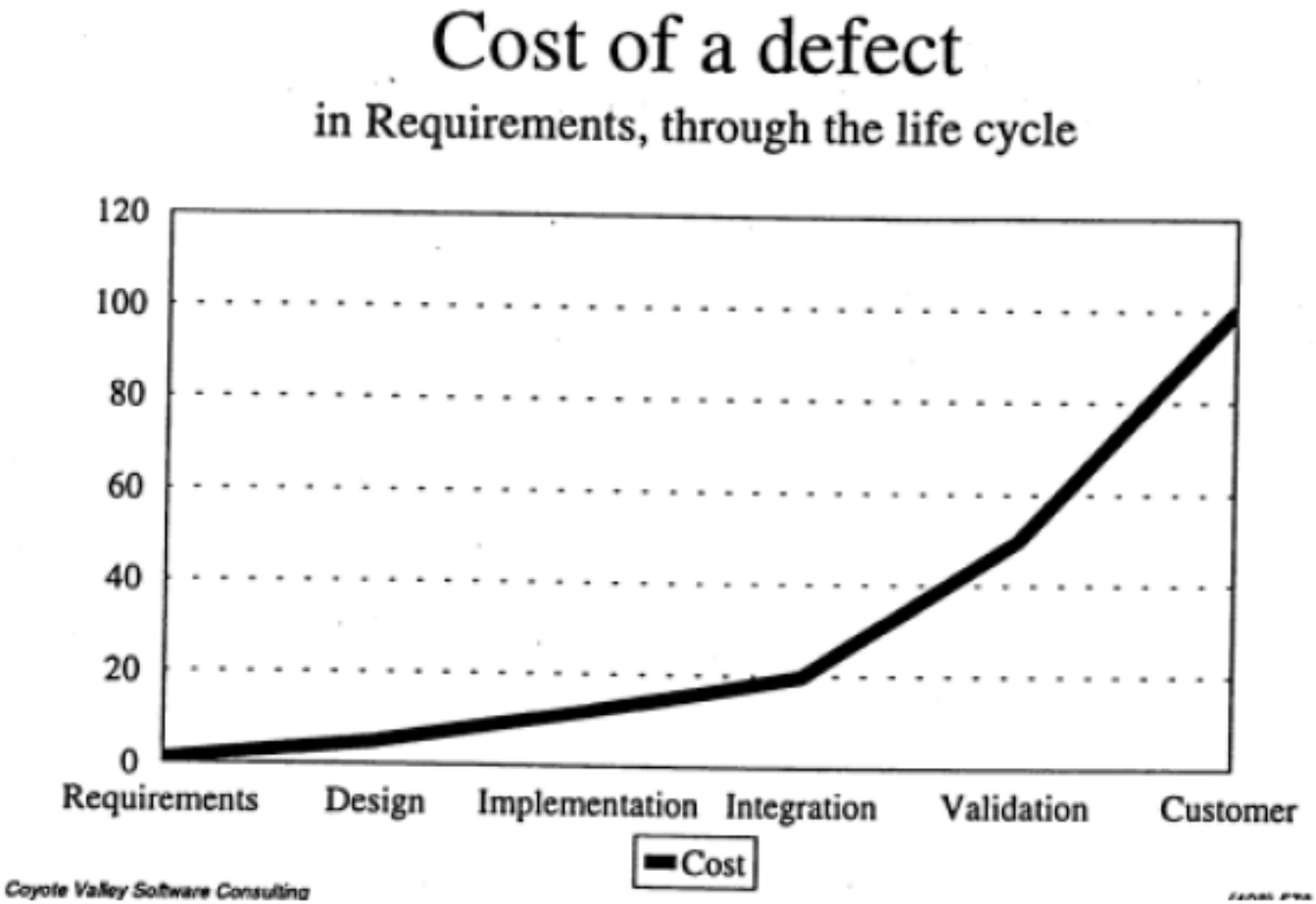
缺乏资源(10.6%)，没有执行层支持(9.3%)，缺少规划(8.1%)

项目成功的因素

- 用户的参与：15.9%
- 管理层支持：13.9%
- 清晰的需求描述(13.0%)；
- 合适的规划(9.6%)；
- 现实的客户期望(8.2%)；
- 较小的里程碑(7.7%)；
- 有才能的员工(7.2%)

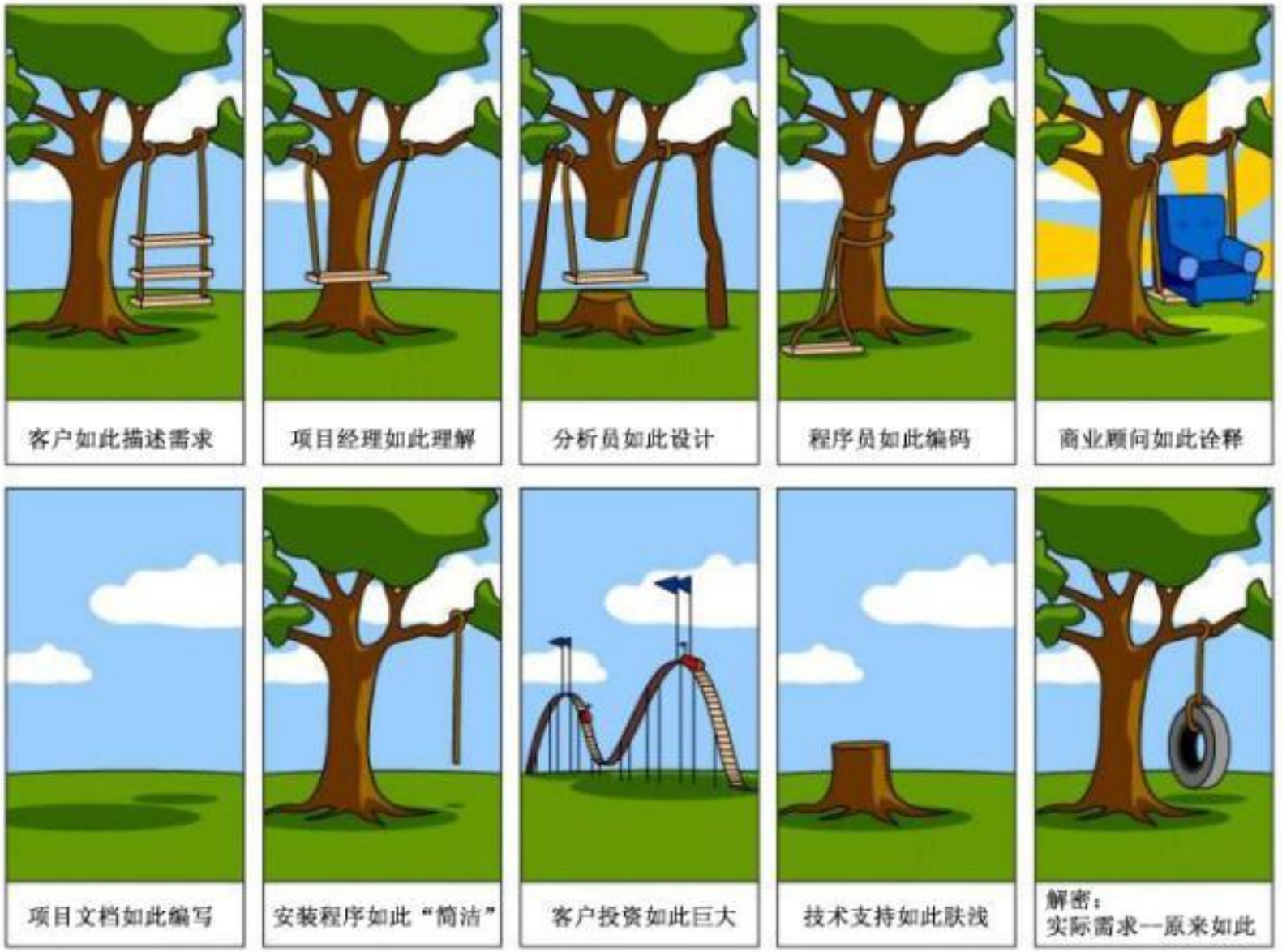
需求的重要性

- 需求错误导致的成本是修复程序错误成本的100倍



软件需求曾经让我们如此狼狈

Measures



什么是软件需求？

- ＩＥＥＥ软件工程标准词汇表（１９９７年）中定义需求为：
  - （1）用户解决问题或达到目标所需的条件或能力（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所需具有的条件或能力。
  - （3）一种反映上面（1）或（2）所描述的条件或能力的文档说明。

用户的类型

Measures

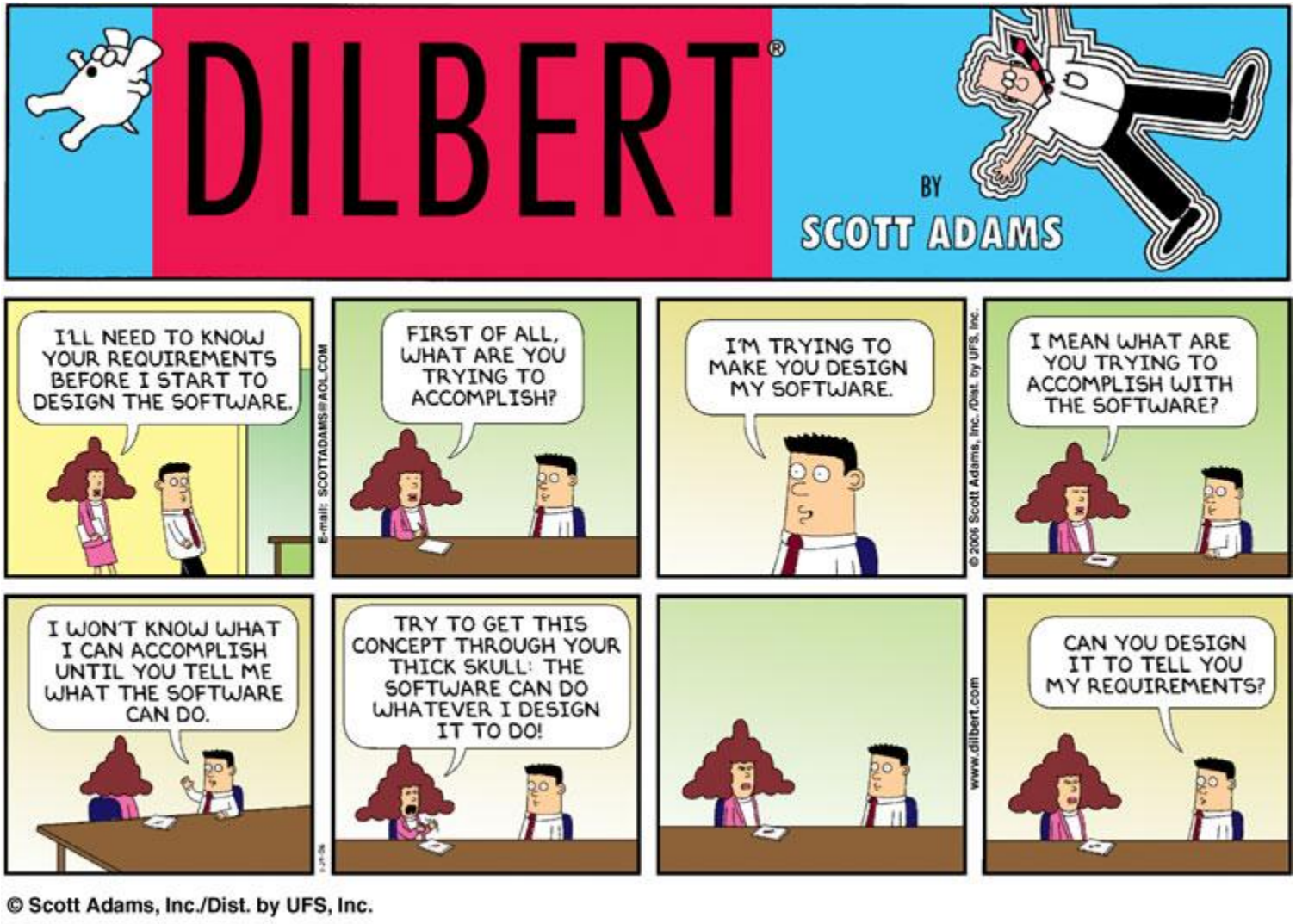
客户	• 掏钱买软件的用户
最终用户	• 真正操作软件的用户叫最终用户。客户与最终用户可能是同一个人也可能不是同一个人
间接用户	• 既不掏钱买该软件产品，也不使用该软件，但是它可能对软件产品有很大的影响

Measures

需求获取

需求获取的困难
需求获取的原则与步骤
需求获取的内容
需求获取的常用技术

需求分析人员的沟通技巧



需求获取的困难

Measures

- 客户不真正知道自己需要什么
- 难以清晰明白的表达自己的需求
- 不知道实现需求的成本，提出不切实际的需求
- 用自己的术语表达需求，需求分析人员没有领域经验
- 不同的需求提供者提供的需求互相矛盾

需求获取的3个原则

Measures

- 深入浅出
  - 全面、深入的了解需求
  - 简单、概括的抽象需求
- 以流程为主线，采用IPO的思想获取需求
  - 从用户的角度理解需求
- 要从系统的全生命周期过程来考虑需求

以业务流程为主线获取需求

- 站在使用者的角度获取需求
  - 业务是如何流转的？
  - 是如何使用系统的？
- 采用输入、处理、输出的思想获取需求
  - 视同整个系统为一个盒子
  - 视同整个业务单位为一个盒子
  - 视同某个人为一个盒子

获取系统全生命周期的需求

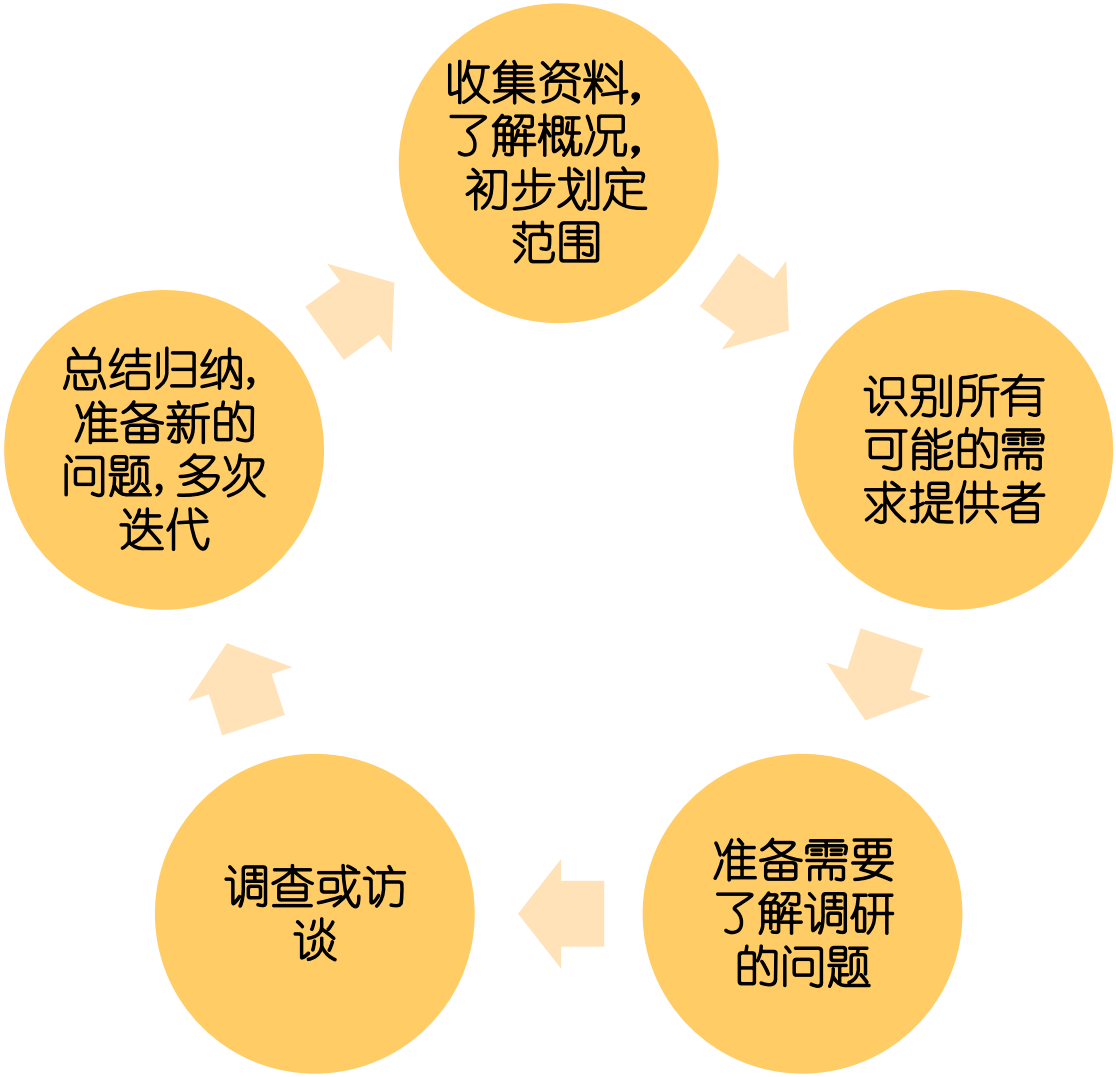
- 从系统的全生命周期的角度识别系统的功能
  - 如何使用？
  - 如何生产？
  - 如何安装？
  - 如何培训？
  - 如何维护？
  - 如何报废？
- 在每个生命周期的阶段涉及到了哪些人、设备或系统？他们会有哪些需求？

需求获取的方式

Measures

- 与用户个别交流
- 需求讨论会
- 查阅相关文档(文件)
- 分发问卷调查表
- 现场访问客户
- 业务流程分析
- 同类产品分析
- 根据现有系统推导出需求
- 回顾以往项目
- 观察用户对原有系统的使用

需求获取五步法

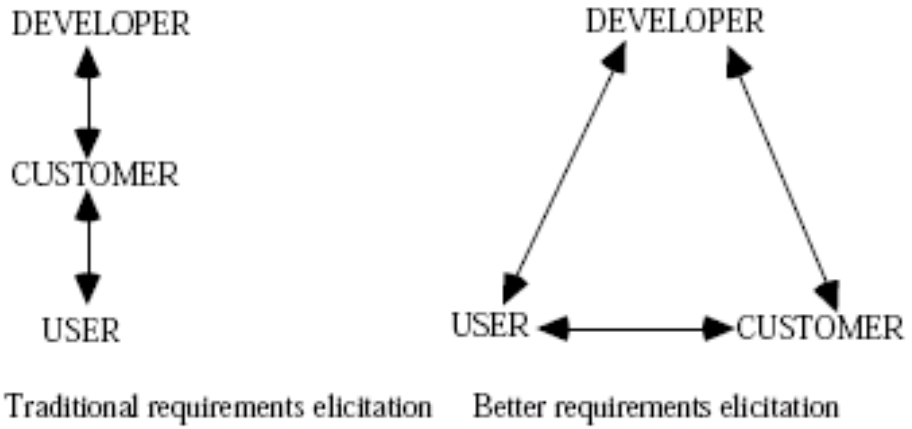

收集资料，了解概况，初步划定范围
识别所有可能的需求提供者
准备需要了解调研的问题
调查或访谈
总结归纳，准备新的问题，多次迭代

收集资料，了解概况

- 通过各种方式收集客户的资料，了解系统的背景
- 进行总体情况的调研
- 初步确定系统的目标与范围

识别所有可能的需求提供者

- 谁使用该系统?
- 谁维护该系统?
- 谁需要从系统中获取数据?
- 系统的运行会影响到谁?
- 谁推广该系统?
- 谁测试该系统?
- 谁生产该系统?
- 谁购买该系统?



Traditional requirements elicitation　　Better requirements elicitation

制定详细的需求调研计划

- 确定需求调查的方式，例如：
  - 与用户交谈，向用户提问题。向用户群体发调查问卷。
  - 参观用户的工作流程，观察用户的操作。
  - 与同行、专家交谈，听取他们的意见。
  - 分析已经存在的同类软件产品，提取需求。
  - 从行业标准、规则中提取需求。
  - 从Internet上搜查相关资料。
- 策划需求调研的活动、时间、地点、参与的人员
- 一次访谈不宜客户太多
- 需求分析员应事先了解用户的身份、背景

准备调研

- 通用问题列表
  - 现有系统是如何运作的？
  - 现有系统存在什么问题？
  - 希望新系统解决什么问题？
  - 客户希望如何解决问题？
  - 希望交付哪些工作产品？
  - 最终用户的背景如何？
  - 对系统的速度、可靠性、安全性、数据容量的要求？
  - 系统的运行环境是什么？
  - 业务流程的启动条件、终止条件、正常事件流、异常事件流、输入数据、处理规则、输出数据
  - 数据的名称、来源、计算方法、类型、计量单位、精度、取值范围、去向、生成时间、产生的频度、高峰期的频度、存储方式、保密要求
  - 最重要的3项需求是什么？
  - 将来有何变化?
- 可以采用表格的方式列举上述问题，并记录在表格中

需求获取的常用技术-需求访谈

Measures

- 在调研前和用户讲清楚调研的意义、过程、以及需要注意的问题。
- 3人访谈小组：1人提问，1人记录，1人辅助
- 衣着得体，体现专业精神
- 准时到达，限制面谈时间
- 先了解宏观，再了解细节。
- 以IPO思想作为主线贯穿始终
- 在用户讲解时,不要中断用户,使对方有充分的演说机会。
- 注意寻找异常和错误情况
- 注意交谈的技巧，并尽可能多的记住用户的姓名、职务、爱好等。

需求获取的常用技术-需求访谈

Measures

- 详细记录
  - 时间、地点、被访谈的人员、角色、持续的时间、参与的人员
  - 记录需求的来源
    - 通过需求源的信息可以知道在变更时需要咨询哪些人和哪些文档
    - 需求源信息有利于理解需求存在的原因
    - 常见的5个来源
      - 需求的项目相关人员
      - 组织的标准
      - 技术文档
      - 事件报告
      - 其他需求
- 指出和记录下未回答条目和未解决问题
- 同样的需求要从不同的渠道（多于2人）进行验证

面谈之后

- 复查笔记的准确性、完整性和可理解性
- 把所收集的信息转化成适当的模型和文档
- 确定需要进一步澄清的问题
- 绘制组织结构图、业务流程图

需求发生冲突时谁决策？

Measures

- 由客户决策而不是由开发方决策
- 可以引导客户或用户进行决策，但不是替他们决策
- 最终用户通常比客户更有发言权

需求获取的常用技术-用例

Measures

- 使用用例来抽取需求
  - 用例指与最终用户和系统之间某个交互类型有关的交互会话. 最终用户使用用例来模拟他们的交互
  - 用例可以看作是解释如何使用系统的经历.
  - 用例描述中包含的信息:
    - 在进入用例之前系统状态的描述
    - 用例中正常的事件流
    - 正常事件流的异常
    - 可以同时运行的其他活动的信息
    - 用例完成后系统状态的描述

ATM 取款Use Case描述

- Use Case编号:001
- Use Case名:ATM取款
- Use Case描述:储户使用信用卡，在ATM机上取款
- actor：储户
- 前置条件：ATM机器处于正常准备状态
- 后置条件：若成功，则储户取出钱，帐户上扣除钱；若失败，储户没有取到钱，帐户上钱数不变。
- 基本路径

1，储户插卡；

2. ATM机提示输入用户口令；

3. 储户输入口令；

4. ATM机口令验证通过，提示输入钱数；

5. 储户输入钱数；

6. ATM机进行钱数有效性检查，提示操作成功，吐出卡和钱；

7.储户取走卡和钱；

8.ATM机屏幕恢复为初始状态。

- 扩展点

4a. ATM机验证用户口令不通过

4a1. ATM机给出提示信息，并吐出信用卡；

4a2. 储户取出卡；

4a3. ATM机屏幕恢复为初始状态.

6a. ATM验证用户输入钱数超过3000

6a1. ATM机给出提示信息，并吐出信用卡；

6a2. 储户取出卡；

6a3. ATM机屏幕恢复为初始状态.

。。。。

补充说明

需求获取的常用技术-原型法

- 原型法的目的
  - 需求原型：获取需求
  - 设计原型：验证技术路线
  - 产品原型：构造产品
- 原型构建方法
  - 纸上原型
  - 界面原型
  - 可执行的原型
- 原型的分类
  - 抛弃型原型
  - 演化型原型：必须易于升级和优化

界面原型

Measures

My Order

订单编号：0000000001
订单状态说明：未付款

商品名称	地图下载链接	激活码	订单状态	定价	优惠价	数量	小計
C720地圖更新版	› 地圖下載	123456789	未完成付款	2,500	2,500	1	2,500
C720地圖更新版	› 地圖下載	123456789	未完成付款	2,500	2,500	1	2,500

优惠券抵扣：1,000

總計: 5,000

購買人資料

購買人姓名 | Mio
電　　話 | 123456789
地　　址 | 12345678912345678912345678912345678
Email | 12345678912345678

(兩聯式發票)

統一編號 | 123456789　　公司抬頭 | 123456789

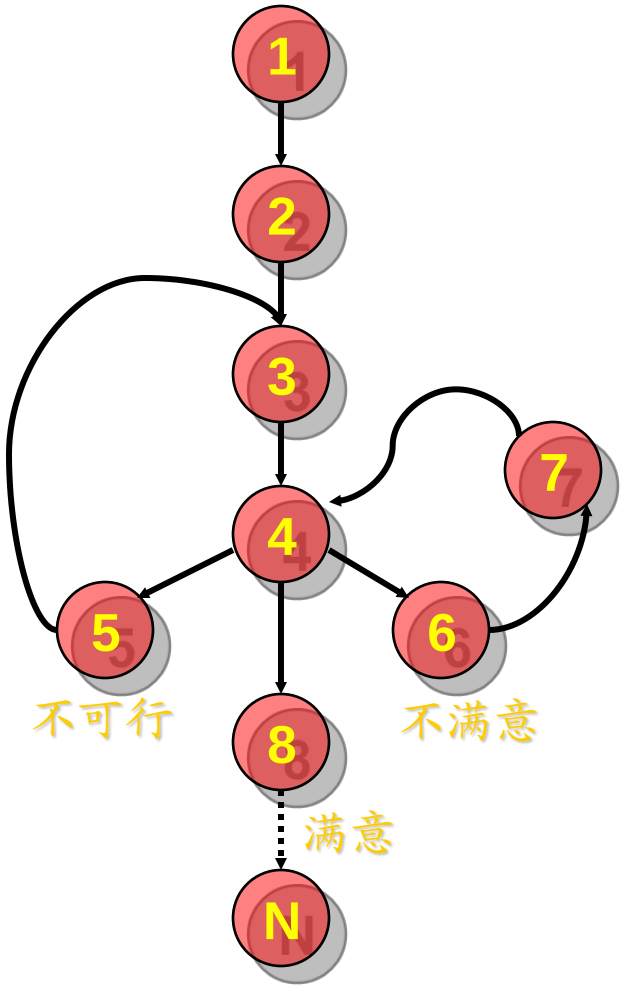
收貨人資料

收货人姓名 | Mio
收货人电话 | 123456789
邮编 | 12345
地址 | 12345678912345678912345678912345678

› 回上頁

原型法工作流程

1、用户提出系统要求
2、识别、归纳上述要求
3、开发一个模型/原型
4、评价模型
5、模型不可行处理
6、模型不满意处理
7、修改模型
8、确定模型后的处理
N、实际系统开发、运行、维护等



原型法的特点

- 优点：
  - 1、开发效率高；
  - 2、开发工具先进，与用户交流直观；
  - 3、符合人们认识事物的规律；
  - 4、能及早暴露系统实施后潜在的一些问题；
  - 5、能调动用户参与的积极性。
- 缺点：
  - 1、不适合大型系统的开发；
  - 2、不适合大量运算及逻辑性强的模块；
  - 3、对原企业基础管理工作要求较高；否则容易走上机械模拟原手工系统的轨道。
  - 4、不适合批处理系统。

关于原型系统的原则

- 区分两种原型系统
  - 抛弃型原型
  - 演化型原型：必须易于升级和优化
- 原型化难以理解的需求
- 尽快将抛弃型原型交付给客户，不必考虑质量
- 必须注意精选正要开发的原型系统所包含的特性，使其能真正达到预期的目的
- 决不把抛弃型原型系统发展成为最终系统
- 利用原型减少软件开发的风险

与原型有关的度量数据

Measures

- 低于100个功能点(大约是12500行C语言代码)的项目可能不需要原型;
- 在100到1000个功能点之间的项目，需要建立原型;
- 超过1000个功能点的项目，需要比原型更形式化的技术;
- 对于超过50000个功能点的大型项目，原型并非总是有效
- 原型规模一般是最终应用系统规模的10%

需求获取的常用技术-专题讨论会

Measures

- 需求专题讨论会
  - 封闭1-2天
  - 确保合适的需求提供者参与
  - 作好后勤保障
  - 事先准备相关材料

讨论

- 你还采用哪些了需求获取的方法？
- 你对上述的哪些实践有切身的体会？

练习

Measures

- 场景：
  - 假如我们公司（以下简称丙方）要为中海集团开发一套需求管理系统，系统的使用者包括了中海集团（以下简称甲方）与其软件供应商们（以下简称乙方，如吉联），甲方通过该系统提出需求变更，监督每个需求的实现进展，乙方通过该系统实施需求变更，记录需求的状态，统计需求变更的次数等。
- 指令：
  - 分组练习：每组不超过7人，指定一名小组的负责人
  - 划分角色：需求分析小组3人（丙方），小组负责人要划分在需求分析小组，甲方与乙方扮演者4人
  - 需求获取：
    - 根据场景，甲方与乙方4人讨论如何表达需求，如何划分更详细的角色，丙方3人讨论如何获取需求，如何访谈哪些角色，5分钟
    - 三方就访谈的角色达成一致，5分钟
    - 丙方访谈甲方与乙方小组 20分钟
    - 丙方描述访谈结果 5分钟
    - 第2轮访谈 10分钟
    - 三方就访谈记录达成一致 5分钟
  - 点评 30分钟

Measures

需求分析

需求分析的目的
设定目标, 划分范围
需求分析的思维方式
需求分析的方法
需求分析的检查单

需求分析的目的

- 消除原始需求中存在的:
  - 冲突
  - 重叠
  - 遗漏
  - 不一致
  - 不切实际的
- 细化需求
- 划分需求的优先级
- 需求建模

需求分析的思维方式

Measures

穷举
- 确保需求无遗漏

分类
- 确保需求无遗漏并去除冗余的需求

分层
- 结构化表达需求

抽象
- 识别出稳定与变化的需求

需求的分类-1

Measures

- 功能需求
- 非功能需求
  - 界面需求
  - 性能需求
    - 速度
    - 容量
    - 精度
    - 吞吐率：单位时间内完成的事务个数
    - 可靠性：通常表示为2次故障间的平均无故障时间
    - 可用性：不停机时间，用户可以访问该产品
  - 易用性需求
  - 操作环境需求：例如路桥收费系统，抽油机伺服系统
  - 可维护性和可移植性需求：
    - 组织
    - 环境
    - 法律
    - 业务规则
  - 安全性需求
    - 保密性
    - 可存取性
  - 文化和政策需求
  - 法律需求

需求的分类-1-非功能性需求

Measures

- 非功能性需求并不改变产品的功能。它增加了一些处理，使产品更易于使用、更安全或者交互性更强。非功能需求描述了用户在使用软件时的体验。
- 功能性需求以动词为特征，非功能需求以副词为特征
- 如何度量非功能需求?
  - 能度量则度量
  - 不能度量则原型
- 客户会关注哪些非功能需求?会在什么时间关注这些需求?
- ”想当然” 现象

非功能需求规格说明可以使用的度量

Measures

系统特性	可以使用的度量
可靠性	出错时间
	错误发生率
有效性	请求后出错的可能性
性能	每秒处理的事务数
	对用户输入的响应时间
存储利用	系统最大的容量（MB）
可用性	学习75%的用户功能所需要的时间
	在给定的时间内由用户引起的错误的平均值
健壮性	系统出错后重新启动的时间
完整性	系统出错时，允许的数据丢失的最大程度

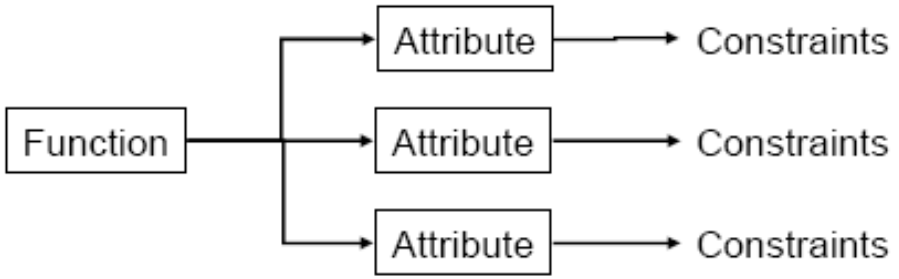
需求的分类-2 FURPS+

Measures

- Functional（功能性）：特性、能力、安全性
- Usability（可用性）：人性化因素、帮助、文档
- Reliability（可靠性）：故障周期、可恢复性、可预测性
- Performance（性能）：响应时间、吞吐量、准确性、有效性、资源利用率
- Supportability（可支持性）：适应性、可维护性、国际化、可配置性
- “+”是指一些辅助性的和次要的因素：
  - Implementation（实现）：资源限制、语言和工具、硬件等等
  - Interface（接口）：为外部系统接口所加的约束
  - Operations（操作）：系统操作环境中的管理
  - Packaging(包装)：提供什么样的部署、移交“介质”和形式等
  - Legal（授权）：法律许可、授权或其他有关法律上的约束。

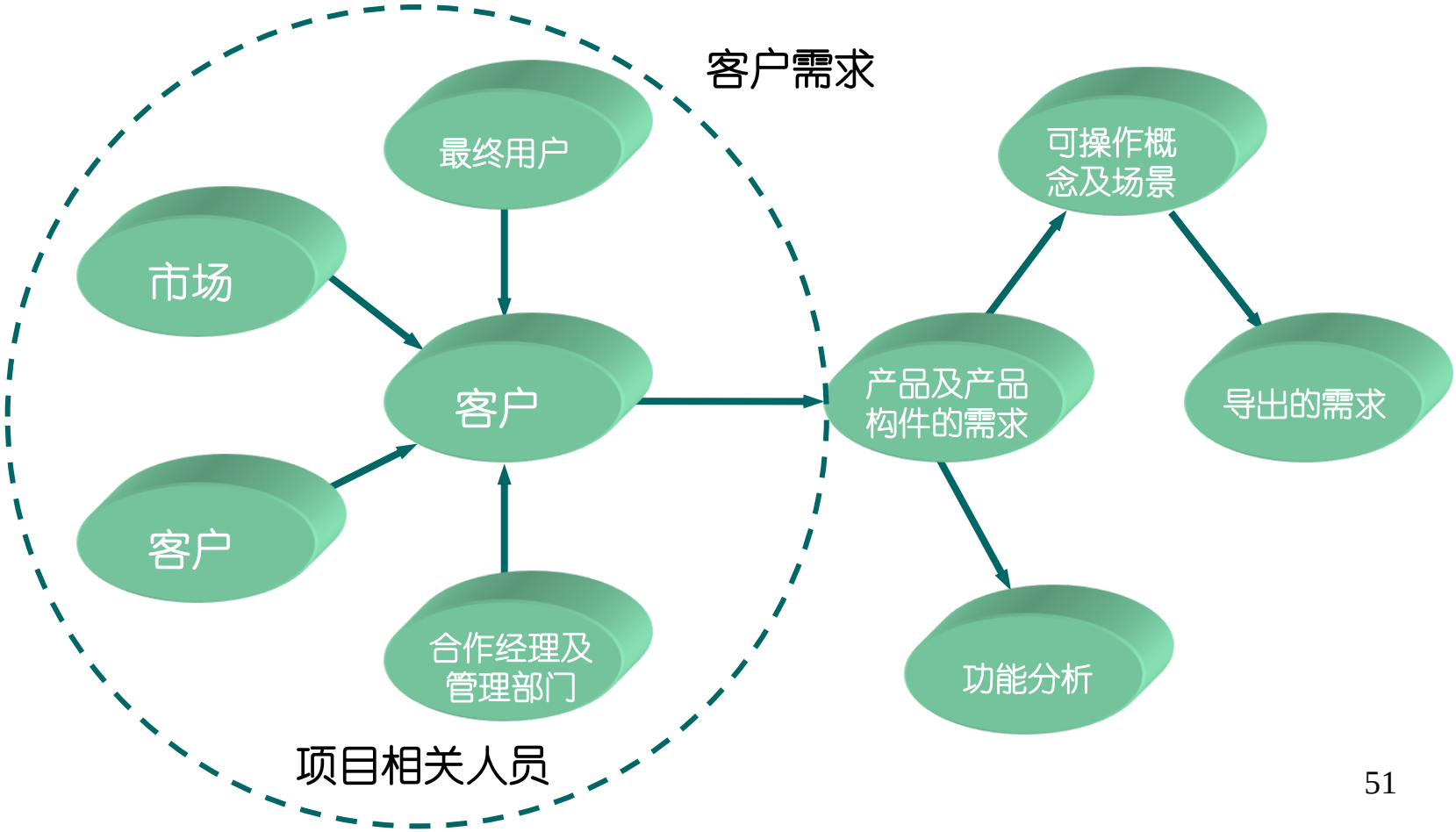
需求的分类-3

- 功能
  - 做什么
  - 采用动词描述
  - 存在也是一种功能
  - 可观察不可度量
- 属性
  - 功能的特征
  - 形容词或者副词
  - 可度量
- 约束
  - 属性的边界
  - 可度量，可测试



需求的分类-4

- 客户、产品及产品构件的需求



需求的分类-5

Measures

- 需要
  - 必须满足，不能裁剪
- 期望
  - 可以裁剪
- 约束
  - 对于系统的设计或开发系统的过程的限制，它不影响系统的外部行为，但必须被完成以满足技术、商业或合同的义务。如；
    - 操作环境：用VB编写软件
    - 与已有系统的兼容性：必须能够在新旧两种平台上运行
    - 应用标准：必须采用公司的类库，必须符合**标准
  - 处理设计约束的指南
    - 和需求区分开来
    - 集中存放
    - 确定设计约束的来源
    - 为每条约束的理由建档
- 接口需求
    - 接口需求为什么重要?
    - 针对接口如何安排后续的设计与编码,联调顺序?

需求的分类-6

- 全局性需求
  - 影响的面比较广，一旦发生变更，工作量比较大
  - 全局性的非功能的系统特性
    - 效率
    - 可靠性
    - 多平台的支持
    - 多语言的支持
    - 可复用的支持
    - 多数据库的支持
  - 系统中物料的编码规则
  - 任意条件查询
  - 系统的总体界面设计
  - 整个系统对操作员权限的限定
    - 数据权限
    - 功能权限
    - 时间权限
    - 部门权限
- 局部需求

需求的分层

	提出者	获取方法	文档量	文档形式	评审方式	稳定性	返工影响	优先级的确定者
目标需求	高层经理	访谈	几页	ppt,word	正规评审会	最稳定	最大	客户
业务需求	中层经理	访谈	几十页	excel,word	正规评审会	较稳定	次之	客户
操作需求	操作员+开发人员	原型	几十页,上百页	word	非正式评审会,正规评审会,分多次评审	最易变化	局部影响	客户+开发人员

讨论

- 在上述的多类需求中，对系统架构起到主要作用的需求有哪些类？

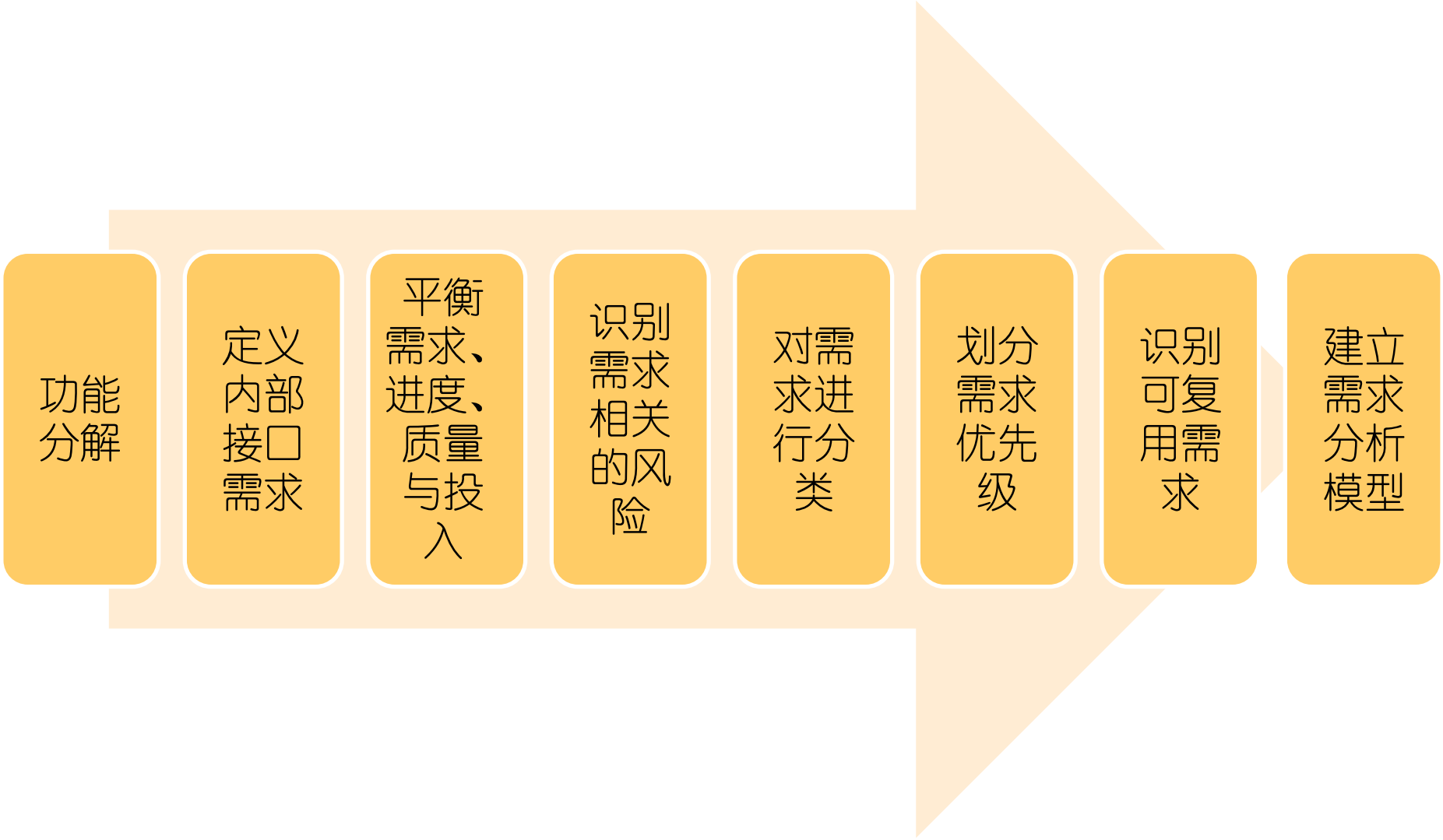
需求分析的方法

结构化方法

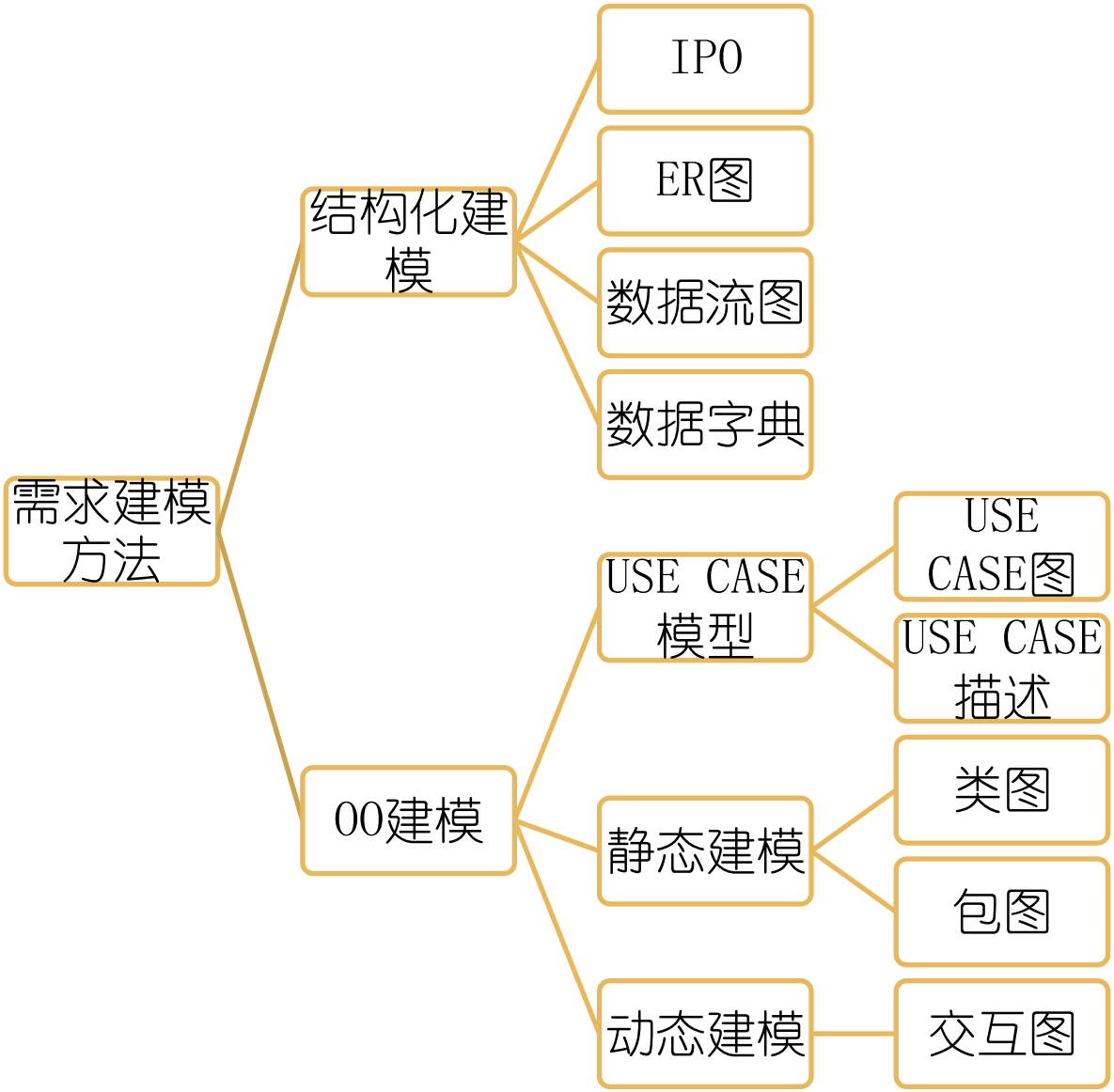
面向对象的方法

需求分析的步骤-列举需求

需求分析的步骤-整理需求



需求建模


需求建模方法
结构化建模
IPO
ER图
数据流图
数据字典
OO建模
USE CASE模型
USE CASE图
USE CASE描述
静态建模
类图
包图
动态建模
交互图

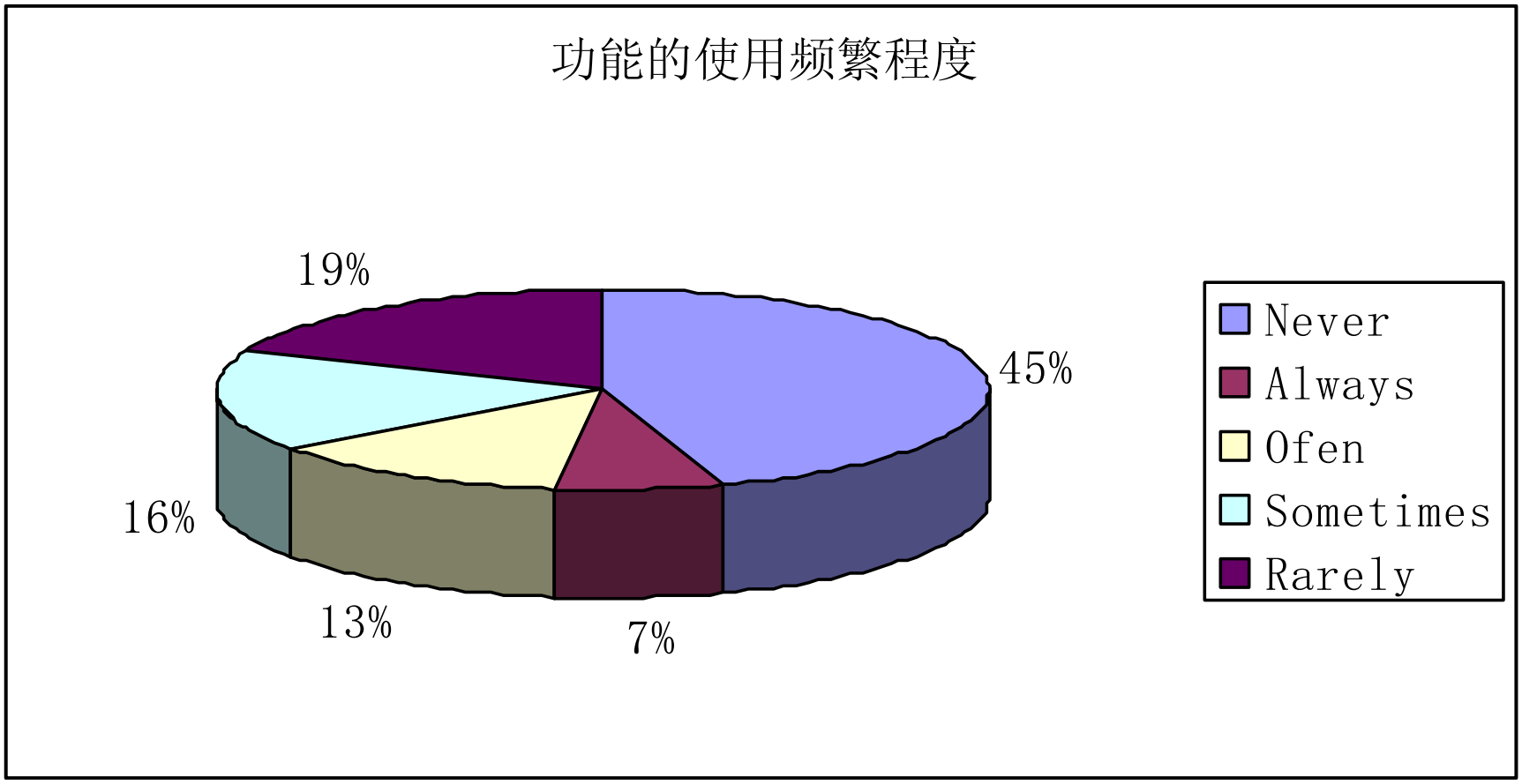
设定系统目标与划分系统范围

Measures

- 目标
  - 要解决的核心问题是什么?
  - 为解决该问题的约束有哪些?
- 范围
  - 哪些是系统应该完成的任务?
  - 哪些不是系统的责任?
  - 从广度与深度2个纬度考虑范围
    - 广度：覆盖的业务,部门,功能
    - 深度：做到什么程度？

- Gilb的模糊目标定律：一个没有明确目标的项目，是不可能明确地实现其目标的。

为什么需要划分需求的优先级



如何划分需求的优先级

Measures

- 优先级的分配由系统分析员和客户一起完成
- 优先级一般分分为3级，不宜定义太多的等级
- 帮助客户定义优先级的问题：
  - 最重要的3个需求是什么？
  - 是否有其他方法可以满足这一需求？比如手工处理是否更合理？
  - 如果忽略或者推迟实现这一需求，其后果是什么？
  - 如果不立即实现这一需求，对项目目标会有什么影响？
  - 如果将这一需求推迟到下一个版本中实现，用户为什么会不满意

使用需求分析检查单来分析需求

Measures

- 检查单中的问题
  - -需求中包含不成熟的设计或实现信息吗?
  - -这项需求还可以细分为不同的需求吗
  - -这项需求只是系统的装饰,而不是真正必需的吗?
  - -这项需求符合系统的目标吗?
  - -这项需求存在二义性吗?
  - -这项需求可以实现吗?
  - -这项需求是可测试的吗?
- 检查单最好不要超过10个问题.

Measures

需求描述

需求描述的内容
需求描述的文档格式
好的需求有哪些特征
需求描述的技巧

需求描述的内容

Measures

用户角色

业务流程

功能

界面原型

数据

处理规则

其他约束等

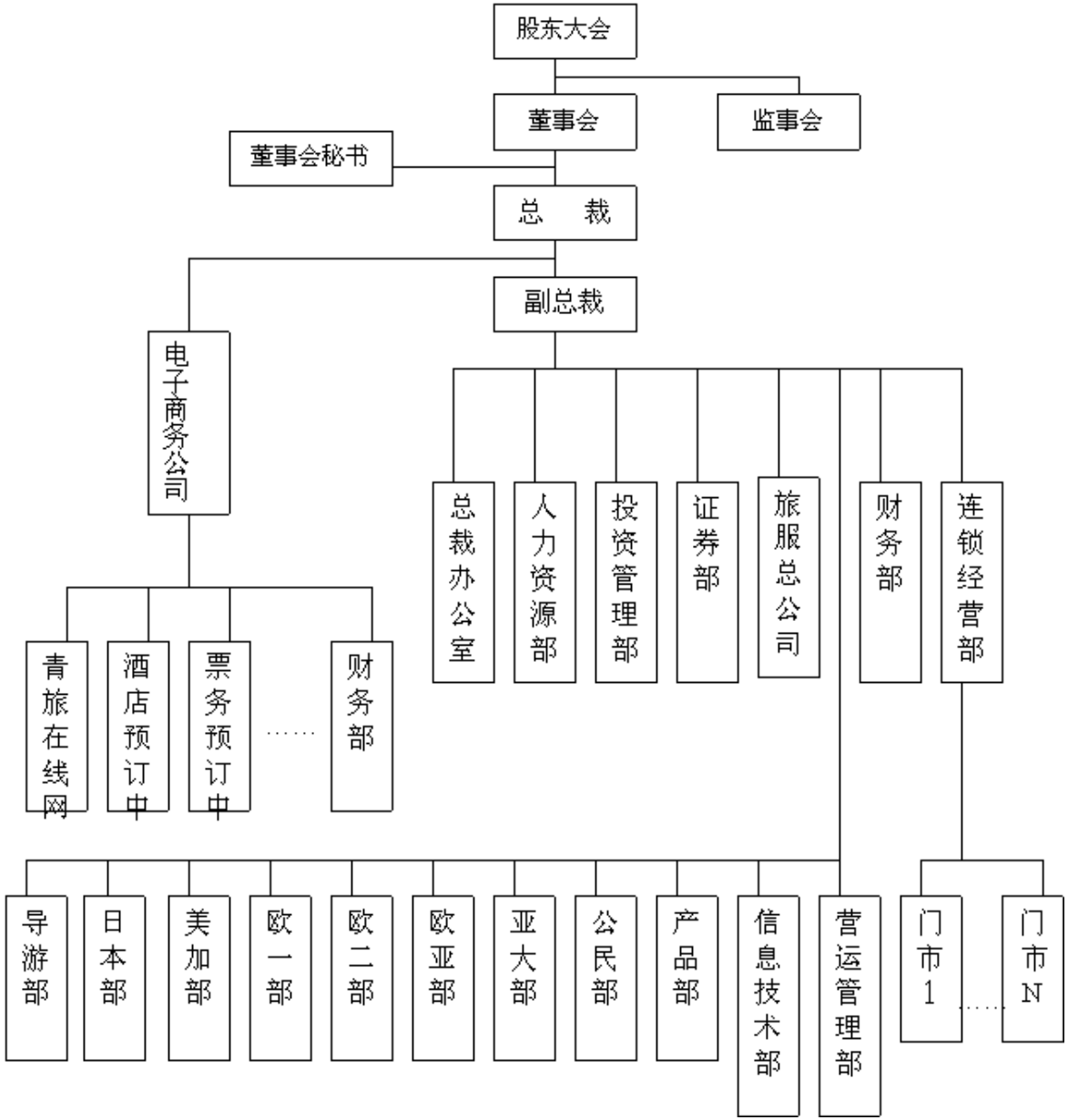
上述7个方面的关系

组织结构图

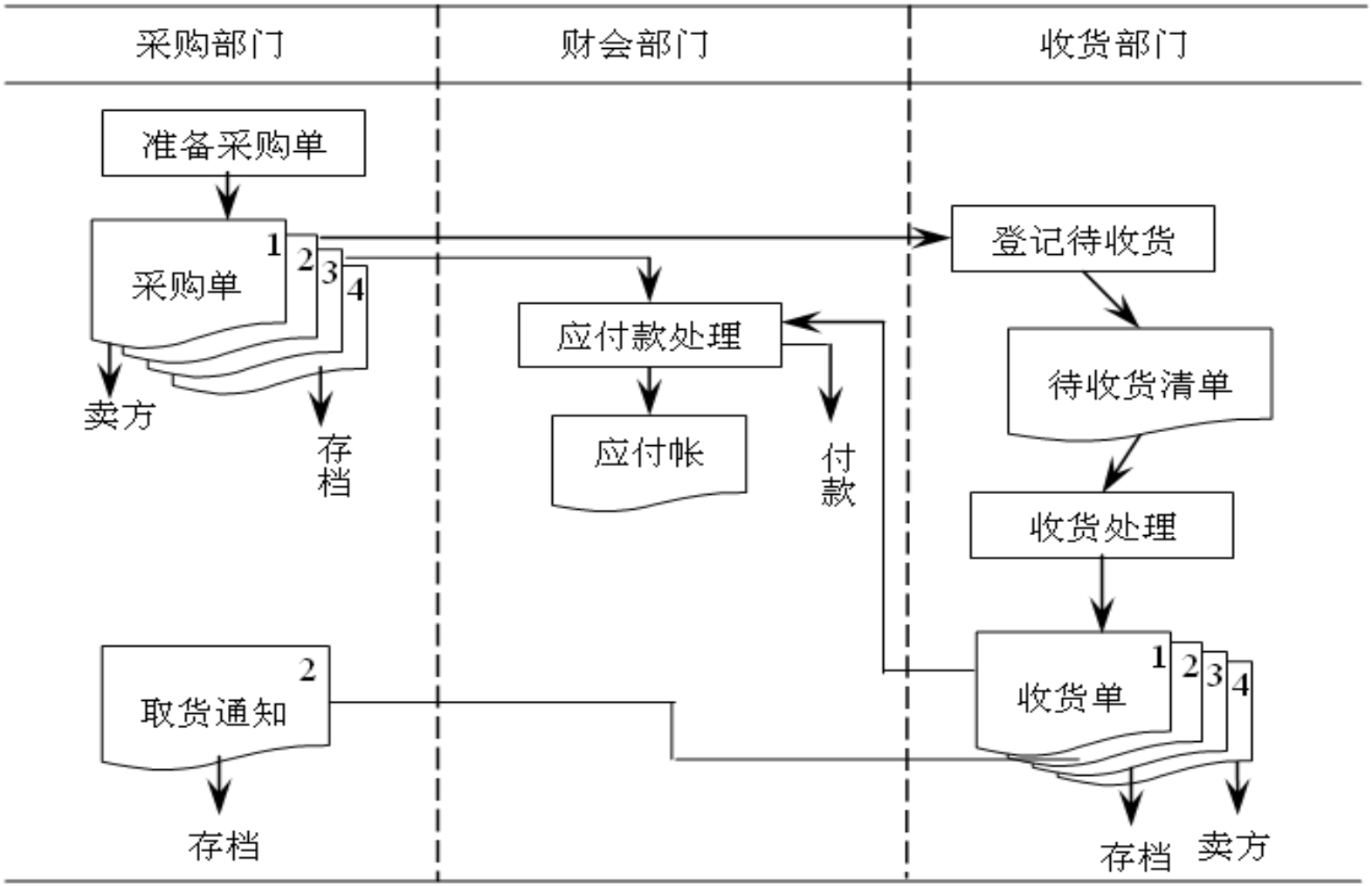
- 将调查中了解到的组织结构具体化
- 反映组织内部隶属关系的树形结构图
- 只包括与业务有关的组织和部门

组织结构图举例

某旅行社组织结构图



业务流程图


采购部门
财会部门
收货部门
准备采购单
采购单
1
2
3
4
卖方
存档
应付款处理
应付帐
付款
登记待收货
待收货清单
收货处理
收货单
1
2
3
4
存档
卖方
取货通知
2
存档

需求描述的文档格式

- PPT
  - -什么时机用PPT来表达需求?
- EXCEL
  - -需求列表
- WORD
- ROSE
  - -有什么优缺点?
- 问题
  - -需求描述详细到什么程度?

好的需求有哪些特征

Measures

- 正确
- 清楚
- 无二义性
- 一致
- 必要
- 完备
- 可实现
- 可验证
- 确定优先级
- 阐述“做什么”而不是“怎么做”

什么是好的需求规格说明书-1

Measures

- 正确
  - 如果软件必须在5秒之内响应所有的按钮消息，而SRS的描述是：“软件必须在10秒内对所有的按钮事件做出响应”，那么这个需求就是不正确的。
  - 为确保需求是正确的，开发方和用户必须对《需求规格说明书》进行确认。
- 清楚
  - 需求的目标读者是开发人员、测试人员、用户
  - 需求是否容易让各个相关人员读懂？

什么是好的需求规格说明书-2

- 无二义性
  - “无二义性” 是指每个需求只有唯一的含义，不同的人有相同的理解。如果需求存在二义性，将会导致人们误解需求而开发出偏离需求的产品。
  - 为了使需求无二义性，人们在写《产品需求规格说明书》时措词应当准确，切勿模棱两可。
- 一致
  - “一致” (Consistent) 是指《产品需求规格说明书》中各个需求之间不会发生矛盾。矛盾常常潜伏在需求文档的上下文中。

什么是好的需求规格说明书-3

Measures

- 必要
  - “必要”=“雪中送炭”
  - “必要”往前一步：
    - “画蛇添足”：吃力不讨好
    - “锦上添花”：用户不会为此多付钱
- 完备
  - “完备”（Complete）是指《产品需求规格说明书》中没有遗漏一些必要的需求。
  - 人们往往倾向于关注系统的特色功能，而忽视了其它一些不起眼的但却是必需的功能。
  - 不完备的《产品需求规格说明书》将导致产生功能不完整的软件，用户在使用该软件时可能无法完成预期的任务。

什么是好的需求规格说明书-4

Measures

- 可实现
  - “可实现”意味着在技术上是可行的，并且满足时间、费用、质量等约束。
  - 在合同前就要降低客户的期望值，不要纳入不可实现的需求
- 可验证
  - 各项需求对用户方而言应当都是可验证的（Verifiable）。
  - 例如，摩天大楼的一项需求是“抗十二级台风”，这个需求看起来堂而皇之，但是如何验证呢？当摩天大楼完工后验收时，用户又不是巫师，他怎能造个十二级台风来试验？如果双方都认可“采用计算机模拟十二级台风”等效于实际测试，那么这项需求就是“可验证”的。

什么是好的需求规格说明书-5

Measures

- 确定优先级
  - 为什么要确定需求的“优先级”？
    - 理论上讲，软件的所有需求都应当被实现。但是在现实之中，项目存在“进度、费用、人力资源”等限制。在项目刚开始的时候，开发方和客户比较乐观，什么都要做，可是做着做着，人们常常会面临“进度延误、费用超支、人员不足”等问题，这时就乱套了。
    - 人们想出了“取舍”办法：先做优先级高的需求，后做（甚至放弃）优先级低的需求，这样可以将风险降到最低。
  - 需求的优先级其实就是需求“轻重缓急”的分级表述，例如划分为“高、中、低”三级。一般地，由用户和开发方共同确定需求的优先级。

什么是好的需求规格说明书-6

Measures

- 阐述“做什么”而不是“怎么做”
  - 《产品需求规格说明书》的重点是阐述“做什么”，而不是阐述“怎么做”。“怎么做”是系统设计和实现阶段的事情。
  - 如果考虑了“怎么做”，可以写下来，但是不要将“怎么做”写到需求规格说明书里面。
  - 举例：走势信息以用VB写的直方图报告给出，x轴显示原因，y轴是发现的错误。
    - 为什么是VB?
    - 为什么是直方图？为什么不用条形图或者饼图或者其他方式？
    - 报告是显示出来，还是打印出来，拟或是其他输出方式？

讨论：

一份好的需求规格说明书，真正能够解决问题吗?

如何提高需求描述的易读性-1

Measures

- 多用图表
  - -使用图表的时机：
    - 当某个对象（系统、文档等等）由多个模块和组件构成，而你又希望阐明他们之间的关系时，可以使用结构图
    - 当需要描述一系列事件或者一个顺序的过程，而每个行为都有一些输入和输出时，使用图表来显示过程的各个步骤
    - 当需要说明空间布局时
    - 当需要使用一些分解结构时
    - 当需要表达复杂的逻辑时
    - 当表示稳定与变化两部分时，使用表格来显示共同点和不同点
    - 不要试图用自然语言描述复杂的关系，试着用图表

如何提高需求描述的易读性-2

- 多用图表
  - -不要使用复杂的图标，使用几个简单的图标比使用一个复杂图要好的多
  - -图表要有编号，如图2-1等

进货方式	发票联	随货同行联	进货报单	外采单	代销验收单	已销代销通知单
正常经销	√	√	√	√		
代销进货	√				√	√
延期付款	√	√	√	√		

如何提高需求描述的易读性-3

Measures

- 针对不同的读者，选择不同的描述格式
- 用短段。一个段落少于7个句子
- 一个句子表达一项需求
- 每行的文字要短
- 使用主动语气，而不是被动语气
- 不使用嵌套的条件从句表达需求，如：
  - 如果X那么如果Y纳入R1A否则Z那么R1B否则，可以试着采用其他方法，如决策表
- 应包含一个需求概要，需求文档越庞大，越需要概要

如何提高需求描述的可验证性？

- 定义系统的验收标准
  - 验收标准定义了客户可接受的最低要求
  - 验收标准有利于发现需求中不明确、不详细的部分
  - 甲乙双方要对验收标准达成一致
  - 验收标准尽可能量化
- 尽可能定量的描述需求
  - 对用户输入的响应时间
  - 学习75%的用户功能所需要的时间
  - 查询的时间
- 无法定量描述的需求，采用原型描述

如何提高需求描述的一致性？

Measures

- 在功能需求的描述中，对于类似的、统一的功能可以单独地进行详细描述，其他地方进行引用，或做为术语进行定义，以简化文档，减少重复。如；
  - 录入功能
  - 打印功能
  - 条件查询功能
  - 排序功能等等
- 定义标准术语，避免二义性
  - 业务术语
  - 技术术语

讨论

- 需求描述详细到什么程度?
  - 平均每个功能点大约需要有半页纸的篇幅描述需求（大概的比例对于C语言是1页需求4-5页代码）
  - 可测试
    - 测试人员可以设计测试用例

需求文档与设计文档的区别-1

Measures

	需求文档		设计文档	
	客户需求	产品需求	概要设计	详细设计
综述	传统的认知是需求描述系统“做什么”，设计描述“如何做”，其实这是和现实不相符合的。无论是在结构化方法中还是在00的方法中，需求分析的结果即包括了“做什么”也包括部分“如何做”，只不过“如何做”部分抽象的层次比较高而已。“做什么”是一个笼统的概念，在刻画“做什么”时，会刻画已有的系统是如何做的，如系统作业流程、处理规则等，这实际上也是在定义系统“如何做”，但是这里描述的“如何做”是从最终用户可见的角度来描述的。当然客户也可能对“如何做”提出一些约束条件，在需求描述文档中，一般称为“设计约束”。无论是数据流程、还是类图其实都包含了设计的成分。			
	从客户的角度用客户的术语描述的需求。客户需求的描述不必拘泥于形式。	从开发者的角度，用开发人员的术语描述的需求。是对系统具体功能与性能的描述。	主要是描述系统由什么产品构件（包、类、子系统、模块等）构成，这些构件之间是什么关系	描述每个构件的实现方法

需求文档与设计文档的区别-2

Measures

	需求文档		设计文档	
	客户需求	产品需求	概要设计	详细设计
编写者	由客户编写或者由客户叙述、需求分析人员编写的。也可有客户的代理编写。客户方内部要对该需求达成一致。	需求分析人员编写	需求分析人员或者设计人员	设计人员或者实现人员
来源	客户提供的各种资料 访谈记录	客户需求 设计方案 开发方附加的需求	产品需求	产品需求与概要设计
约束力	是客户验收的依据。是验收测试用例的主要依据。	是开发方验收系统的依据。也可以作为客户验收的依据。是系统测试用例的主要依据。	是集成测试用例设计的主要依据。	是单元测试用例的依据之一。

需求文档与设计文档的区别-3

Measures

	需求文档		设计文档	
	客户需求	产品需求	概要设计	详细设计
结构化方法	系统的目标 系统的范围 系统的运行环境; 系统的用户 系统的使用场景（组织结构图、业务流程图）； 功能性需求 非功能性需求; 其他约束	产品的功能性需求 非功能性需求; 产品的分解结构；（模块结构图，分解的层次应是客户可理解） 每个产品构件的需求; 产品的外部接口需求; 产品构件之间的接口定义; 需求的优先级与分类; 系统的数据视图（E-R图）； 系统的处理流程；（数据流图） 系统的设计约束; 系统的运行环境	系统的体系结构 系统的技术路线：核心设计思想 系统的模块划分 系统模块之间的接口关系 系统的内外部接口关系 系统的数据结构：数据库的设计、共享的数据结构的设计; 核心技术问题的解决方案; 系统复用构件的设计：可以复用的模块的设计; 界面风格设计：整体的界面设计: 设计约定：进行详细设计的风格与内容一致性要求;	实现的功能 输入数据 输出数据 实现算法 数据结构 交互界面

需求文档与设计文档的区别-4

Measures

	需求文档		设计文档	
	客户需求	产品需求	概要设计	详细设计
面向对象方法	系统的目标 系统的范围 系统的使用场景（组织结构图、业务流程图）； 系统的用户 业务用例 系统用例 非功能性需求 其他约束	系统的目标与范围概述 包图 业务用例图（可选） 业务用例描述（可选） 系统用例图 系统用例描述 对用例的补充性说明 领域模型：实体类的类图 系统的设计约束； 需求的优先级； 系统的运行环境	系统结构划分：分层结构图、包图 静态模型：描述了有哪些类？类与类之间有哪些静态关系？类的类型？主要通过类图的形式来表达。 动态模型：描述了类与类之间是如何交互的？主要通过顺序图来表达，也可以采用协作图、活动图等。 设计约定：进行详细设计的风格与内容一致性要求 复用设计：系统可复用元素的设 界面风格设计：整体的界面设计 持久对象设计：需要持久存放的对象与数据库的映射关系	类的具体责 类的外部接 类的属性与方法定义 方法的逻辑设计

需求文档与设计文档的区别-5

Measures

	需求文档		设计文档	
	客户需求	产品需求	概要设计	详细设计
描述的详细程度	目标与范围必须描述清楚	可测试、可验收。	在00方法中，要将产品拆分到包及类，类并不一定能覆盖所有的类，但是包对外的接口类一定要覆盖，包内部的核心类要覆盖。在结构化方法中，要详细到模块，即函数的上级系统元素。	方法或函数的逻辑设计要完成
可裁剪性	如果是产品开发，客户需求对应为市场需求。客户需求也可以合并到产品需求文档中。	该文档是必须的。	该文档是必须的。	该文档可以合并到概要设计文档中。 在敏捷的软件开发方法中，该文档可以合并到代码中，通过代码的注释及代码替代详细设计文档

Measures

需求的验证与确认

需求验证与确认的方式

需求评审

需求验证与确认的方式

Measures

- 检查文档是否符合标准
- 组织正式的需求审查
- 需求审查应有一个多学科的小组来进行
- 使用原型来确认需求
- 编写用户手册草案
- 设计测试用例

讨论：

为什么不去做需求的验证与确认?后果是什么?

需求评审的基本原则

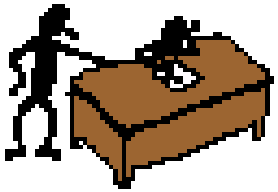
Measures

- 分层次评审
- 正式评审与非正式评审结合
- 分阶段评审
- 精心挑选评审员
- 对评审员进行培训
- 充分利用需求评审检查单:样例:需求评审检查单.xls
- 充分准备评审
- 定义评审结束的条件
- 做好评审后的跟踪工作
- 建立标准的评审流程

需求审查过程活动

Measures

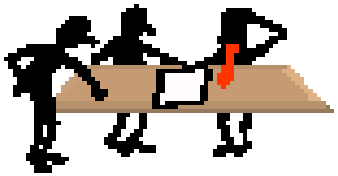
准　备



主持人和作者选择和准备材料

主持人选择小组和安排审查

概况会议



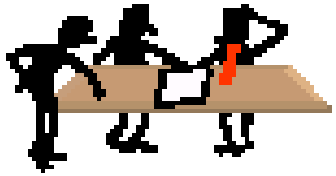
角色分配

材料分发

个人评审

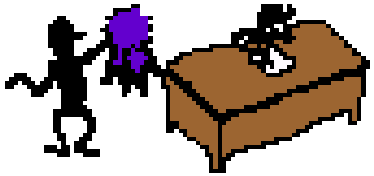
评审人审查工作产品

记录会议



小组记录缺陷

返　工



作者返工产品

主持人验证返工

活动一：准备评审

Measures

- 选择主持人
  - 作者或作者之一不可以是主持人
  - 作者的经理不可以是主持人
  - 主持人应该是组织经过筛选、培训的人员
  - 主持人可以不是技术专家，但是要对被评审的内容有所了解
- 选择评审员
  - 工作产品的上游工作者、下游工作者、同行作为评审员
  - 评审员要经过培训
- 选择评审材料
  - 一次评审的材料不能太多
  - 对于规模比较大的评审材料可以分片
  - 一次代码评审不要超过500行
- 为评审员分配角色
- 培训评审员
- 安排评审时间
  - 概要会议的时间一般为半小时，不能超过2小时
  - 记录会议的时间为2小时
  - 如果需要专家做1小时的个人评审，则为专家预留出1天的个人评审时间
- 检查入口准则的满足性

活动二：概况会议

Measures

- 主持人给审查组讲授
  - 要寻找什么
  - 使用的过程，对不熟悉该过程的新员工进行培训
  - 评审材料和参考材料
  - 每个评审人要扮演的角色（例如，客户，最终用户，设计人员，测试人员）
- 作者提供工作产品的背景
  - 审查材料的概况
  - 回答问题，澄清含义
  - 和其它参考工作产品的关系
- 会议时间可能会变，但不应该超过2小时

活动三：个人评审

Measures

- 每一个小组成员评审材料
  - 评审态度是找到所有缺陷
  - 基于检查单和模板进行考查
  - 使用已分配的角色观点
  - 记录工作产品中找到的缺陷
    - 项或问题的位置
    - 项的类型
    - 项的严重程度
  - 将时间/工作量和总结数据记录在表中
  - 如果审查的工作产品还没有准备好，或发生其它问题会造成小组无法召开记录会议，通知主持人

个人评审的度量数据

Measures

- 基本度量元
  - 评审规模：页或行
  - 评审时间：小时
  - 发现的缺陷：个数
- 派生度量元
  - 评审速率：页（行）/小时
  - 评审效率：缺陷数/小时
  - 缺陷密度：个数/页（行）

工组产品类型	每小时数
需求	250行（5页）
概要设计	200行（4页）
详细设计	150行（3页）
源码，测试用例	150行（无注释） （4页）
测试计划	200行（4页）

活动四：记录会议

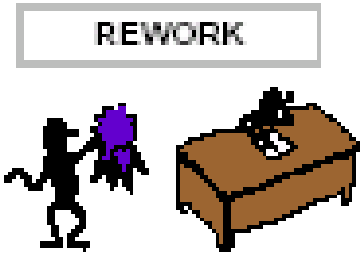
Measures

- 安排主持人、记录员、作者坐在一起
- 在会议开始前，把总结数据提交给主持人
- 在记录会议开始时
  - 建议限制用2分钟进行评论和讨论
  - 先收集与参考文档或检查单相关的项
  - 审查者先指出工作产品的正面因素，淡化缺陷集的负面效果
- 主持人保证记录会议达到目的
  - 只是捕获缺陷，而不讨论或解决缺陷
- 小组完成一个统一的缺陷记录
  - 对于缺陷要达成一致，评审员对评估结果共同负责
- 形成评审结论
  - 通过
  - 有条件通过
  - 返工，重新评审

活动五：返工及后续工作

Measures

- 对每个被发现的项，作者：
  - 如果是一个缺陷，则进行修复
  - 如果是添加，则要文档化并存档，以备后用
  - 在审查总结单上记录决定
  - 如果合适的话，可能调整严重程度
  - 处理审查过程中提出的问题和提问
- 主持人验证返工是否完成
  - 主要项已经返工
  - 达到或超过同意的缺陷度
- 后续替换工作
  - 主持人评审和批准返工
  - 小组只对返工部分进行评审
  - 整个产品必须重新审查
- 主持人发布审查报告
  - 工作产品的最终处理
  - 最终状态：按类型和严重程度记录项数，最终时间数据



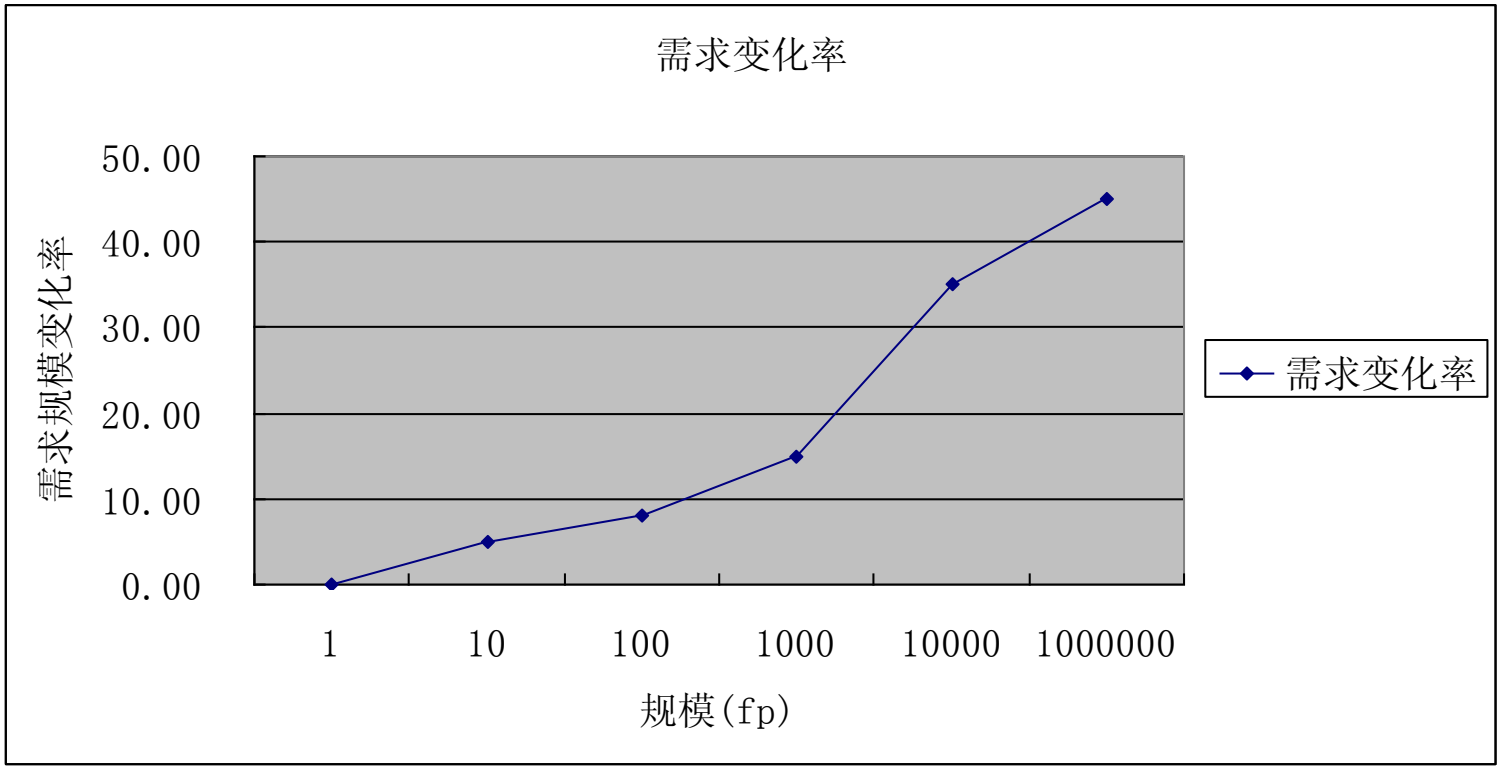
Measures

需求管理

商务手段
沟通手段
技术手段
管理手段

需求的变化是永恒的

- 需求变化的原因
  - 误解
  - 遗漏了需求
  - 外部环境发生了变化，产生了新的需求
- 在美国，软件项目的需求增长平均速度大约是每月1-3%，从需求在开始阶段的”固定”一直到设计和编码阶段



需求是渐变的

- 秃头论证
  - 秃头的标准是什么?是掉了10万根头发?掉1万根头发?还是掉了1000根头发?掉了100根头发?
  - 少一根头发是否造成1个秃头?回答说不能. 再少一根怎么样?回答还是不能. 这个问题一直重复下去, 到后来, 回答却是已成为秃头了.

  这就是欧布里德的:秃头论证. 突变是在人们不知不觉的渐变中发生的, 当你警觉时, 事物的性质早已走到了反面.
- 稻草原理

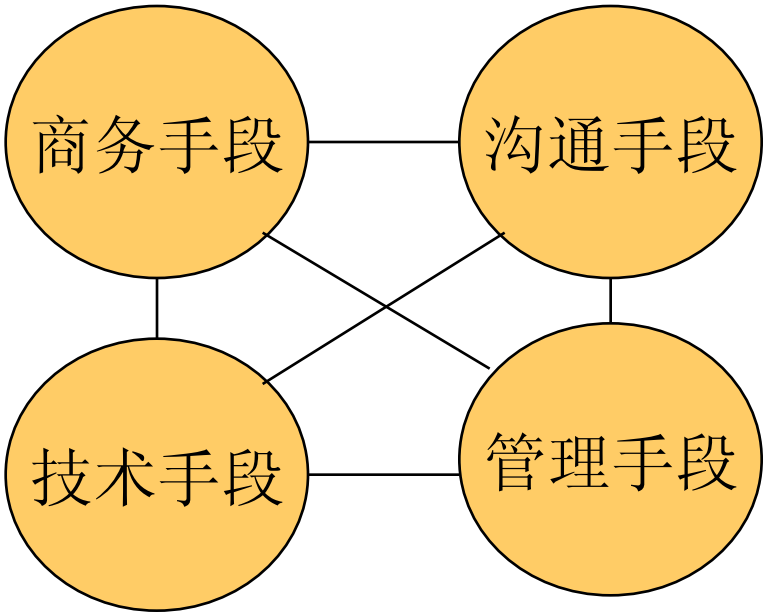
  往一匹健壮的骏马身上放一根稻草，马毫无反应；再添加一根稻草，马还是丝毫没有感觉；又添加一根……一直往马儿身上加稻草，当最后一根轻飘飘的稻草放到了马身上后，骏马就竟不堪重负瘫倒在地。这在社会研究学里，取名为“稻草原理”。
- 蚂蚁效应

  一群蚂蚁选择了一棵百年老树的树底安营扎寨。为建设家园，蚂蚁们辛勤工作，挪移一粒粒泥沙，又咬去一点点树皮……有一天，一阵微风吹来，百年老树轰然溃倒，逐渐腐烂，乃至最终零落成泥。生物学中，这种循序渐进的过程也有个名字，叫”蚂蚁效应”。

如何应对需求变更

Measures

- 需求变更的应对是一个系统工程
- 考虑预防与控制的手段
- 四种手段：



商务手段

Measures

- 需求的定义应该作为合同的附件
- 系统的验收标准应该作为合同附件
- 谁来承担需求的变更成本应该在合同中有定义，如：
  - 变更造成的工作量（成本）变化在20人日或者20%之内由乙方承担，否则由甲方承担
- 需求的变更的流程，决策权应该在合同中有约定，要求甲方要有正规的需求变更的决策流程，而并非随便变化
- 甲乙双方参与的需求变更控制委员会
- 两阶段合同法

沟通手段

Measures

- 需要和客户建立共同的成功理念：
  - 项目成功的基本原则：需求，进度，质量，资源的平衡原则，“多快好省”是客户的期望，但实际上这4个要素是互相约束的是需要平衡
  - 需求要分优先级
  - 系统的目标要明确，系统范围紧紧围绕目标来裁剪
  - 需求要分层次，各个层次的需求要要捕获到
  - 项目的目标是成功，而不是失败，成功可大可小
- 需要和开发人员详细沟通需求
- 客户参与开发的全过程，注意副作用
- 测试人员参与需求开发

技术手段-1

- 需求获取与分析技术
  - 保证需求的完备性
    - 分类
    - 穷举
    - 分层
    - 抽象
    - 原型
    - 产品的全生命周期考虑需求
  - 保证需求的适应性
    - 沟通
    - 经验
  - 保证需求的正确性
    - 确认

技术手段-2

Measures

- 复用技术
  - 架构
    - 领域架构
    - 技术架构
  - 分析模式
  - 设计模式
  - 构件技术
- 工具
  - 自动化的变更控制工具
    - 大于100个功能点的项目，不能仅仅使用版本管理工具
  - 需求管理工具，如DOORS

管理手段

Measures

- 成立正式的变更控制委员会
  - 对于大于5000个功能点的项目一般建立该组织
  - 通常3-7人构成，包括客户代表和开发代表
- 无论大小变更都要纳入正规的变更管理流程
  - 大于500个功能点的MIS系统都会使用需求变更控制流程
- 不同层次、不同级别的需求变更流程不同。如：
  - 变更级别：分为高、低二级。
  - 高：
    - 影响其他项目组的变更；
    - 影响其他项目或者影响项目外部承诺的变更；
    - 单次变更估算规模大于项目总体估算规模5%；
    - 单次变更导致工作量成本增加超过1人周；
    - 项目总体累计变更规模大于项目总体估算规模30%后的变更。
  - 低：其他情况。
- 建立需求跟踪矩阵
- 进行需求变更的影响分析
- 通过生命周期模型的选择来适应需求变化，如：
    - 迭代的方法

建立需求变更控制委员会

Measures

- 参与的人员：
  - 用户
    - 中高层、作业层
  - 开发人员
    - **PM**、需求人员、设计人员、测试人员，编码人员、**CM**等
- 职责
  - 决定要变更哪些需求
  - 变更是否在项目范围之内（包括：项目范围和合同范围。因为有时，在项目范围，但不在合同范围，需要项目进行二期合同开发）
  - 评估变更的波及
  - 决定变更是可以接受，还是放弃。
  - 对变更的需求设置优先级、制定版本规定等。

需求跟踪矩阵（RTM）有什么作用？

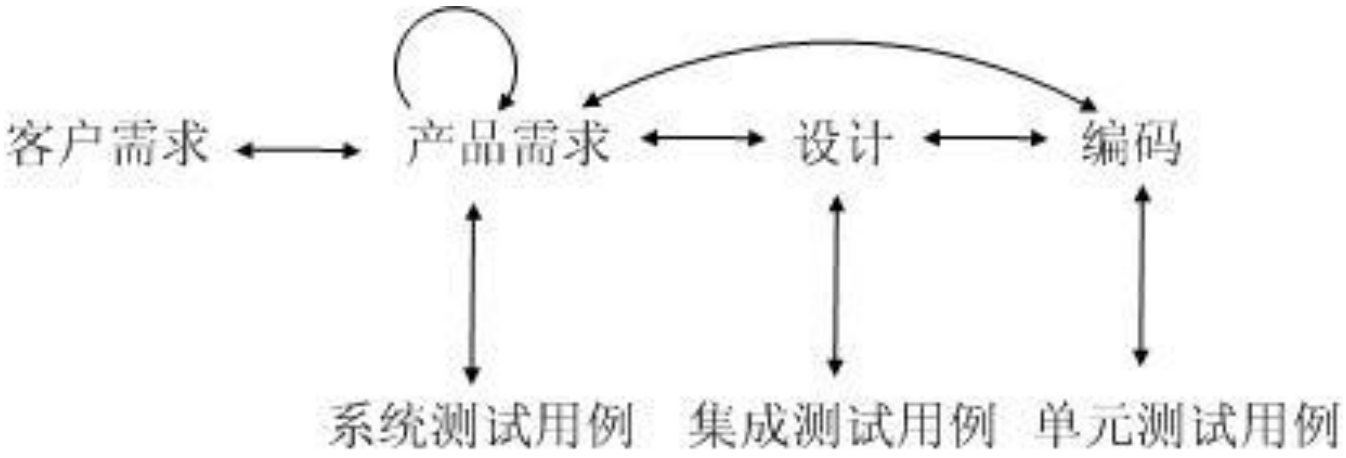
Measures

- 在需求变更、设计变更、代码变更、测试用例变更时，需求跟踪矩阵是目前经过实践检验的进行变更波及范围影响分析的最有效的工具，如果不借助RTM，则发生上述变更时，往往会遗漏某些连锁变化。
- RTM也是验证需求是否得到了实现的有效工具，借助RTM，可以跟踪每个需求的状态：是否设计了，是否实现了，是否测试了。

需求跟踪矩阵的2种跟踪方向

Measures

- 纵向跟踪矩阵，包括如下的3种：
  - 需求之间的派生关系，客户需求到产品需求
  - 实现与验证关系：需求到设计，需求到测试用例等
  - 需求的责任分配关系；需求由谁来实现
- 横向跟踪矩阵：
  - 需求之间的接口关系



讨论

- 在本公司内，应该建立哪些RTM？

需求跟踪矩阵由谁来建立？

Measures

- 有多个角色参与建立RTM。
- 需求开发人员负责客户需求到产品需求的RTM建立
- 测试用例的编写人员负责需求到测试用例的RTM建立
- 设计人员负责需求到设计的RTM的建立等等
- PPQA负责检查是否建立了RTM，是否所有的需求都被覆盖了。

需求跟踪矩阵的3种常见的建立形式

Measures

- 方式一：通过需求与设计、代码、测试用例的编号来实现跟踪，如需求为：r1,r2,……等编号，而设计的编号为：r1-d1,r1-d2,……,测试用例的编号为：r1-t1,r1-t2等等。需要注意的是需求与它们之间是多对多的关系，仅通过编号是无法实现这种关系的。
- 方式二：DOORS之类的需求管理工具
- 方式三：通过EXCEL来维护RTM

#	需求规格说明书（版本，日期）	设计文档（版本，日期）	代码（版本，日期）	测试用例（版本，日期）
1	标题或标识符，说明	标题或标识符，说明	代码名称，说明	测试用例名称，说明
2	…	…	…	…

需求的状态变化

Measures

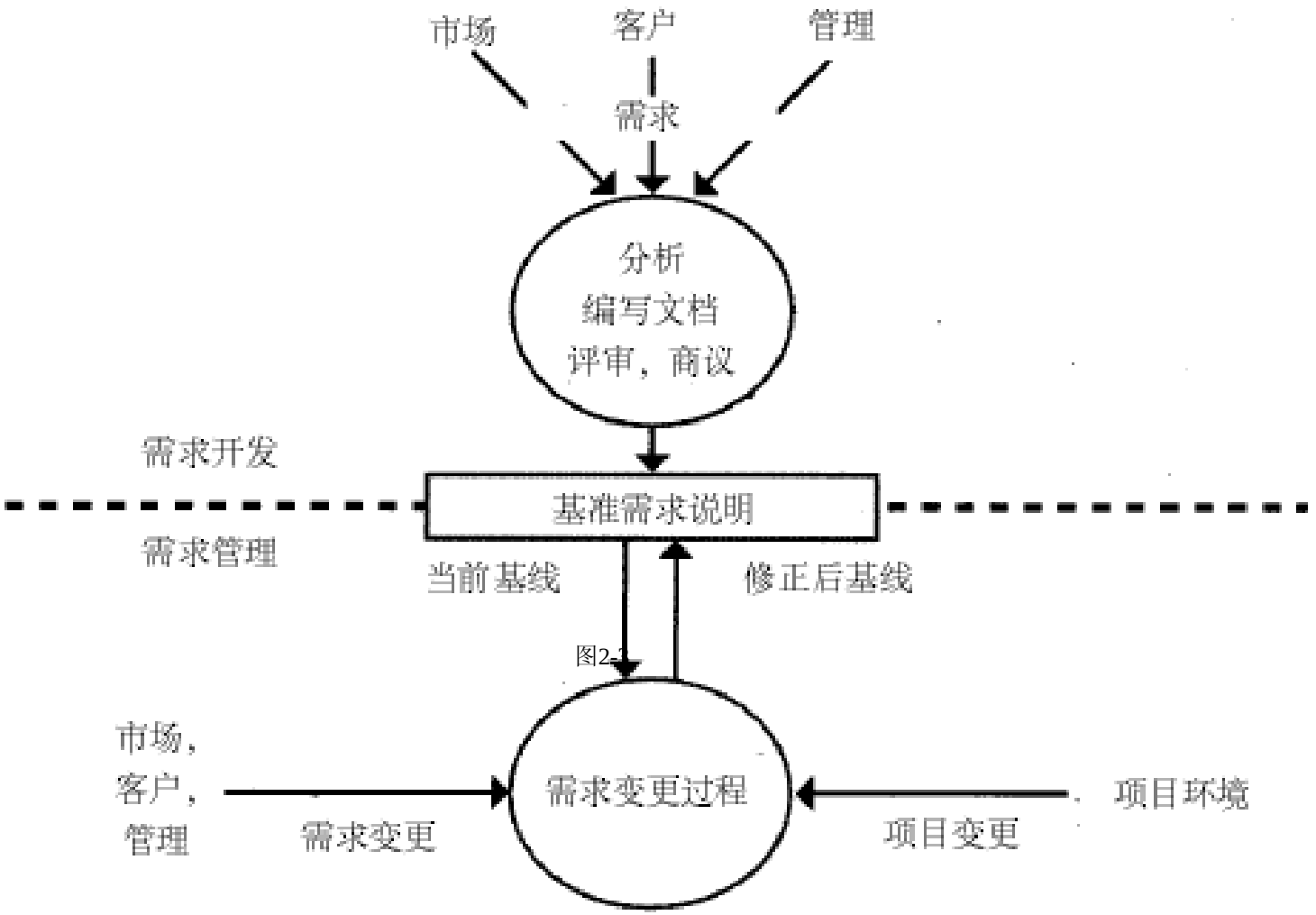
状态	定义
被建议	根据需求来源，责任、相关人提出了需求。
被拒绝	在一系列需求开发过程后，该需求没有被认可。
被批准	在需求（特别是变更需求）被分析，评估了合理、可行、成本、影响等要素，被确认可接受，被标注了新的版本号、给出了新的标号等需求属性、被加入到需求基线库中，进入实现过程。
被实现	已实现设计、编码、单元测试。
被验证	根据验收标准，已经通过集成以上的测试，被验证实现了需求的要求，被放置进配置基线库。表明需求已经被实现。
被丢弃	被批准的需求已从基线库中被丢弃。记录下丢弃的原因和决定责任人。
被交付	通过用户的验收测试，需求以交付物的形式，向用户提交。

需求变更的影响分析

Measures

- 在需求变更时需要考虑如下的影响：
  - 技术影响
    - 其他需求
    - 设计
    - 编码
    - 测试用例
    - 交付的文档
  - 管理影响
    - 规模
    - 工作量
    - 工期
    - 成本
    - 质量
    - 计划
    - 风险

需求开发与需求管理的关系



小结

Measures

- 以目标性需求为基础
- 充分获取3个层次的需求，紧紧围绕目标裁剪需求
- 需求的优先级甚至比需求本身更重要
- 需求分析的思维方法
- 区分稳定与易变的需求
- 需求描述要区分不同的阅读对象
- 需求验证要避免评审疲劳现象
- 管理需求的渐变，控制系统的范围

参考资料

- 软件需求 机械工业出版社 Karl E. Wiegers著,陆丽娜等翻译 2000年.
- 探索需求 清华大学出版社 Weinberg等著,章柏幸等翻译 2004年
- 需求工程 机械工业出版社 Ian Sommerville等著,赵文耘等翻译 2003年
- 软件需求管理—统一方法 机械工业出版社 Dean Leffingwell Don Widrig 著,蒋慧 林东译 2002年

Q?&A!

Measures

谢　谢！