第一章

1软件的模拟特性

软件在运行中表现出来的特性、行为应该和应用的现实情况保持一致。

“模拟”性具体是指以下几点：

1. 目的性。
2. 正确性。
3. 现实可理解性。

2需求问题具体原因分析

1. 非技术性和社会性因素重视不足
2. 传统需求分析方法的缺陷
3. 软件规模的日益扩大
4. 需求问题的高代价性

3需求工程定义

1. 它关注于软件系统所应予实现的现实世界目标、软件系统的功能和软件系统应当遵守的约束
2. 同时它也关注以上因素和准确的软件行为规格说明之间的联系
3. 关注以上因素与其随时间或跨产品族而演化之后的相关因素之间的联系

需求工程的任务：

浅层理解：

软件工程是要做出一个系统，其中，需求工程关注系统将要做什么，后续阶段关注系统将怎样被实现。

深层理解：

软件工程的产品是用来解决问题的。其中，后续阶段是关于**如何解决问题**，而需求工程的目的是定义**所需解决的问题**。

需求工程必要性

1. 软件开发是这样一个工程问题
2. 计算机应用于现实世界的广泛性
3. 容易忽略需求工程重要性的地方

4需求工程基本活动

1. 需求获取的目的是从项目的战略规划开始建立最初的原始需求。
2. 需求分析的目的是保证需求的完整性和一致性。
3. 需求规格说明的目的是将完整、一致的需求与能够满足需求的软件行为以文档的方式明确地固定下来。
4. 需求验证是需求开发中的最后一个活动。它的首要目的是保证需求及其文档的正确性
5. 需求管理的主要工作是跟踪后续阶段中的需求实现与需求变更情况，确保需求得到正确的理解和实现

5需求工程的复杂性

1. 处理范围广泛
2. 涉及诸多参与方
3. 处理内容多样
4. 处理活动互相交织
5. 处理结果要求苛刻

6需求工程师要重视“软技能”

1. 交流技能
2. 观察技能
3. 抽象分析与问题解决技能
4. 写作技能
5. 关系协调和团队工作技能

7需求分析师包括以下工作职责

1. 进行需求调研，获取用户需求
2. 进行系统规划，设计客户信息化管理的蓝图
3. 设计数据模型
4. 设计软件功能
5. 设计软件界面，绘制软件原型
6. 召开需求评审会，收集各方面的意见，根据意见修改设计
7. 指导开发，协助测试
8. 进行软件变更需求分析，设计变更解决方案

8需求分析师性格要求

1. 要善于沟通
2. 要善于听取不同的意见
3. 要有团队精神

第二章

1需求的定义

1. 用户为了解决问题或达到某些目标所需要的条件或能力；
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中的一个条件或一种能力的一种文档化表述。
4. 需求是用户对问题域当中的实体状态或事件的期望描述

2问题域

要解决问题，就需要改变现实当中某些**实体的状态或改变实体状态变化的演进顺序**，使其达到期望的状态或演进顺序。这些实体和状态构成了问题解决的基本范围，称为该问题的问题域（Problem Domain）。

问题域特性的重要性：

要想解决问题，它就需要了解问题域特性，将解决方案和问题域特性（需求背景）结合起来 。

要防止解系统的引入在问题域当中**引发未预见的连锁反应** 。

3解系统

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模拟与共享现象

问题域与解系统能够形成互动的基础是解系统部分模拟了问题域，[Jackson1995b]将这种模拟性称为共享现象(share phenomenon)。

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。

问题域中的某些信息能够和软件中的信息建立映射关系 。

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 ，是问题域和解系统实现**交互和互相影响的途径与接口**。

5直接需求和间接需求

1. 直接：和解系统直接相关，是对共享现象中实体状态或事件的期望，能够通过解系统直接改变共享知识而予以满足。
2. 间接：和解系统间接相关，所期望的实体状态和事件存在于问题域之中、共享现象之外。

6问题解决的困难性

1. 不存在描述明确的E。
2. 不存在确定的针对S的 评估标准R。
3. 根据问题域特性和系统行为推测系统应用效果是简单的推理过程，即E,S→R 是简单的，但根据问题域特性和期望的系统应用效果构建系统行为的过程是困难的，E,R=S是 一个创造性的过程

7需求分层

业务需求是抽象层次最高的需求，是系统建立的战略出发点，表现为高层次的目标(objective),它描述了组织为什么要开发系统。

用户需求就是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。

用户需求具有下面几个特点：模糊、不清晰；多逻辑混杂（任务本身往往含有前后相继的多个逻辑处理过程）；多特性混杂（功能需求和非功能需求混杂在一起）。

适合软件开发者的需求层次是系统级需求，它关注的是软件系统的行为，尤其是系统与外界的交互行为：在接收到一个外界请求时，软件系统应该给外界提供响应。

系统级需求比用户需求更加详细和准确，包含更多的技术细节。

8需求的分类

8.1广泛意义上的需求谱系

项目需求针对的对象是作为项目核心的计划，包括项目的成本、资源、时间和进度等

R10:项目的成本要控制在60万元人民币以下。

R11:项目要在6个月内完成。

过程需求针对的对象是软件开发过程，包括开发人员、工具和方法等

R12:在开发中，开发者要提交软件需求规格说明文档、设计描述文档和测试报告

要解决一个问题，人们需要将软件、硬件和人力资源联合起来，这种联合的形式被称为系统工程，包括软件工程、硬件工程和人力资源管理

R14:系统要购买专用服务器，其规格不低于……

R15:系统投入使用时，需要对用户进行为期一周的集中培训。

8.2严格意义上的软件需求分类

1. 功能需求(functional requirement):和系统主要工作相关的需求，即在不考虑物理约束的情况下，用户希望系统所能够执行的活动，这些活动可以帮助用户完成任务。

功能需求是一个软件产品得以存在的原因，是软件系统能够解决用户问题和产生价值的基础，也是整个软件开发工作的基础。

1. 性能需求(performance requirement):系统整体或其组成部分应该拥有的性能特征。

速度，容量，吞吐量，负载，实时性

1. 质量属性(quality attribute):系统完成工作的质量，即系统需要在一个“好的程度”上实现功能需求
2. 对外接口(external interface):系统和环境中其他系统之间需要建立的接口，包括硬件接口、软件接口和数据库接口等。
3. 约束(constraint):进行系统构造时需要遵守的约束，如编程语言和硬件设施等。约束：系统开发及运行的环境，问题域内的相关标准，商业规则，社会性因素。

系统为满足显式的及隐含的要求而需 要具备的要素称为质量。

为了度量一个系统的质量，人们通常会选用系统的某些质量要素进行量化处理，建立质量特 征，这些特征称为质量属性

1. 可靠性是指在规定时间间隔内和规定条件下，系统或部件执行所要求功能的能力
2. 可用性指软件系统在投入使用时可操作和可访问的程度，或能实现其指定系统功能的概率
3. 安全性是指软件阻止对其程序和数据进行未授权访问的能力
4. 可维护性指为排除故障、改进质量或适应环境变化而修改软件系统或部件的容易程度，包括可修改性(modifiability)和可扩展性(extensibility),
5. 可移植性指系统或部件能从一种硬件或软件环境转换至另外一种环境的特性
6. 易用性指与用户使用软件所花费的努力及其对使用的评价相关的特性

9优秀需求的特性

1.完备性2.正确性3.可行性4. 必要性5.无歧义6.可验证

第三章

1需求获取是从人、文档或环境中获取需求的过程，需求工程师必须要利用各种方法和技术来“发现”需求

需求获取任务

1. 收集背景资料
2. 获取问题与目标，定义项目前景与范围
3. 识别涉众，选择信息的来源
4. 选择获取方法，执行获取，获取功能与非功能需求
5. 记录获取结果

2需求分析的主要工作是通过建模来整合各种信息，以使人们更好地理解问题。检查需求当中存在的错误、遗漏、不一致等各种缺陷，并加以修正。需求分析活动最后会产生一个需求的基线集，它指定了系统(或当前版本的系统)开发需要完成的任务。

需求分析阶段，需求工程师主要的任务包括以下几方面。

1. 背景分析
2. 问题分析、目标分析、业务分析，确定系统边界
3. 软件需求建模
4. 细化需求
5. 确定优先级
6. 需求协商

3需求规格说明，编写文档的主要目的是在系统涉众之间交流需求信息，最重要的质量要求是简洁、精确、一致和易于理解。

需求工程师在这个阶段的主要工作包括以下几方面

1. 定制文档模板
2. 编写文档

4需求验证

文档的要求

1. 文档内每条需求都正确、准确地反映了用户的意图。
2. 文档记录的需求集在整体上具有完整性和一致性。
3. 文档的组织方式和需求的书写方式具有可读性和可修改性。

为了保证以上标准的满足，需求规格说明文档(尤其是最终定稿的文档)在传递给相关人员之前要进行严格的验证。

需求验证阶段的主要任务包括以下两方面

1. 执行验证
2. 问题修正

5在需求开发活动之后，设计、测试、实现等后续的软件系统开发活动都需要围绕需求开展工作。还需要有一种力量保证需求作用的持续、稳定和有效发挥，需求管理就是这样的一个管理活动

需求管理阶段的主要任务包括以下几方面。

1. 建立和维护需求基线集
2. 建立需求跟踪信息
3. 进行变更控制

第四章

1需求获取中的常见困难

1. 用户和开发人员的背景不同，立场不同

知识理解的困难

默认(tacit)知识现象

解：就要求开发人员在开展需求获取之初，尽力去研究应用的背景，理解组织的业务状况，形成一个能够和用户进行有效沟通的粗略的知识框架

为了避免这个问题，很多开发团队会选择一些对计算机比较了解的用户(尤其是系统管理员)来代表用户表述需求。

1. 普通用户缺乏概括性、综合性的表述能力

专家用户因其知识的渊博性而使得自己的知识结构具有概括性和广泛性，普通用户的知识结构就相对局限于一些具体的业务细节

解：为了解决这个困难，要求开发人员在与用户接触之前就先行确定获取的内容主题，然后设计具体的应用环境和场景条件，让用户在执行细节业务的场景中来描述问题和表达期望。

1. 用户存在认知困境

用户无法明确告诉开发者自己到底需要什么,但是当开发者提供一个明确的解决方案时，用户却能够迅速判断出该方案是否以及为什么解决了自己的问题。

解：引导用户去发现用户自己也没有形成明确认知的知识。

1. 用户越俎代庖

用户提出的不是需求，而是解决方案

用户固执地坚持某些特征和功能

解：要求开发者在需求获取的过程中注意保持业务领域和解决方案的区分界限。

用户还拥有一些重要的隐藏需求没有被发现，开发者应分析用户的深层目的，找到隐藏在背后的需求。

1. 缺乏用户参与

用户数量太多，选择困难

用户认识不足，不愿参与

用户情绪抵制，消极参与

没有明确的用户

解：求开发者在进行需求获取时，能够对系统用户以及用户的替代源等相关涉众进行分析，了解他们的特征、类别、任务和取向等，并在获取中采取对策避免用户参与不足现象的发生

2需求获取活动

1. 研究应用背景，建立初始的知识框架；
2. 根据获取的需要，采用必要的获取方法和技巧；
3. 先行确定获取的内容和主题，设定场景；
4. 父析用户的高(深)层目标，理解用户的意图；
5. 进行涉众分析，针对涉众的特点开展工作

3需求获取步骤

①确定待获取信息的内容；

②确定待获取信息的来源；

③确定应采用的获取方法；

④执行获取；

⑤记录成果

4获取信息的内容

1. 需求是获取的主要对象，是系统期望达到的目标。
2. 问题域描述是用来承载和解释需求的问题域特性，主要是现实世界的业务运行状况。
3. 环境与约束属于一种特殊的问题域特性，限定了解系统部署的环境和条件。

5获取信息的来源

①涉众：包括用户、客户、领域专家以及市场人员、销售人员等其他用户替代源。

②硬数据：包括登记表格、单据、报表等定量文档，以及备忘录、日志等定性文档。

③相关产品：包括原有系统、竞争产品及协作产品(和解系统存在接口的其他软件系统)。

④重要文档：包括原有系统的规格说明、竞争产品的规格说明、协作产品的规格说明及客户

的需求文档(委托开发的规格说明、招标书)。

⑤相关技术标准和法规：包括相关法律、法规及规章制度，行业规范、行业标准及领域参考

模型

6获取信息的过程注意事项

1. 在整体上制定组织方案
2. 维护项目的前景和范围
3. 接受需求的不稳定性
4. 控制探索性工作

7防止遗漏需求

1. 务必让所有涉众都表达出自己的意见。
2. 不要以抽象和模糊的需求作为结束。
3. 使用多种方法表达需求信息。
4. 注意检查边界值和布尔逻辑。

8结束获取

①用户想不出更多的用例；

②用户想出的新用例都是导出用例(通过其他用例的结合可以推导出该用例);

③用户只是在重复已经讨论过的问题；

④新提出的特性、需求都在项目范围之外；

⑤新提出的需求优先级都很低；

⑥用户提出的新功能都属于后续版本，而非当前版本

9获取信息的成果

笔录；录音和摄像；项目前景和范围文档，用例文档

第五章

1前景描述了产品用来干什么以及最终将是什么样。前景声明将所有涉众都统一到一个方向上来。范围则指出了当前项目是要解决的产品长远规划中的哪一部分。范围声明则为项目划定了需求的界限。

2问题分析

获取问题：

问题分析的前提是获取问题，这可以通过收集背景资料或与涉众沟通来实现。

收集背景资料时，要收集业务描述及其统计数据(详见第6章6.9节),关注业务困难与问题。与涉众的沟通主要通过面谈完成(详见第8章)

明确问题：

对问题达成共识：具体的方法就是用标准化的格式描述问题，并在涉众之间取得认同

收集背景资料，判断问题的明确性：对于明确的问题，可以直接进行“发现业务需求”活动。对于不明确的问题，就需要进行下一步骤“分析不明确问题”。

分析不明确问题，发现问题背后的问题：直接咨询涉众；利用收集的资料和业务数据；使用一些简单的问题分析技巧

发现业务需求：

每一个明确、一致的问题都意味着涉众存在一些相应的期望目标，即业务需求。

定义问题解决方案及系统特性：

建立问题解决方案

确定系统特性和解决方案的边界

确定解决方案的约束

3目标定义

目标(goal)定义为业务、组织或者系统的高层短的(objective),它们说明了系统被开发的原因，并用于指导企业内各种层次上的决策。

目标是软件系统应该通过行为者(agent)的协作而予以实现的目的,这些行为者来自于被开发的软件系统内部或者系统的应用环境之中。

目标是系统被开发的目的。

4目标的层次

它可以在战略高层进行描述，也可以进行技术上的低层描述，它可以是针对系统坊能的自标，也可以是针对非功能的目标

5目标分类

功能目标是期望系统提供的服务，非功能目标是期望系统满足的质量。

功能目标可以分为满足型目标(satisfaction goal)和信息型目标(information goal

非功能目标可以依据质量模型的属性细分为安全目标(safety goal)、性能目标(performance goal)和可用性目标(usability goal)等。

软目标是指无法清晰判断是否满足的目标。

硬目标则是那些可以通过一些技术确认其是否满足的目标。

6目标关系

目标之间的关系，包括精化(refinement)关系（AND精化关系，OR精化关系）、阻碍(obstruction)关系与冲突(conflict)关系。

支持、冲突关系是对多个目标之间关系的考虑

支持关系处理为OR 精化关系

7目标分析过程

高层目标的获取:

对系统的现状、背景和问题的分析往往能够发现高层次目标，在这一点上与问题分析过程基本一致，只是目标分析需要将识别出来的目标、系统特性等组织为相互联系的目标模型

目标精化:

对一个高层目标的精化需要获取高层目标的相关信息并进行分析，发现其子目标

获取对高层目标的描述

从高层目标描述中发现 AND精化关系

从高层目标描述中发现“候选办法”,发现 OR精化关系

考虑阻碍目标(avoid 目标)实现的情况

发现目标冲突关系

对高层目标问“How”,对低层目标问“Why”,完善层次结构

目标实现:

这个阶段的主要任务是收集与目标相关的需求信息，讨论可能的候选解决方案，确定最终的解决方案。

最底层目标分配给主体；是设计实现最底层目标的操作

8非功能需求分析的困难

1. 非功能需求不集中，在系统中散布
2. 非功能需求不独文，依赖于功能需求
3. 非功能需求的质量需求比较复杂
4. 非功能需求相互冲突、依赖

9目标模型适合于非功能需求分析

在早期信息较为抽象和不充分时建模为软目标，后期添加目标满足标准将其修正为硬目标

让非功能目标支持、阻碍或精化功能目标，描述非功能需求对功能需求的依赖关系

让单个非功能目标依赖多个不同的功能目标，描述非功能需求的散布特性。

使用支持、阻碍(冲突)精化描述不同非功能目标之间的折中与冲突关系。

依赖质量模型，将高层非功能目标精化为子非功能目标，描述非功能需求的层次结构关系

10获取非功能需求及建立目标模型的简单过程

1. 依赖功能需求识别获取非功能需求目标
2. 根据非功能需求层次结构，情化非功能需求目标
3. 量化底层非功能需求目标的验收标准

11系统边界定义

系统边界是系统与环境互动的界限，定义系统边界可以明确系统需要满足的与外界的交互行为，从而从宏观上界定了系统的功能概要。

系统边界是需求工程后期阶段需求分析活动的起始模型，后期的需求分析可以看成是逐一细化系统边界中的对外交互行为的活动

结构化方法中使用上下文图作为系统边界定义模型，面向对象方法中使用系统用例图作为系统边界定义模型。

问题分析：每个问题的解决方案边界，将它们合并(叠加、汇总、补充)起来就是整个系统的边界定义

目标分析：存在系统与环境的互动，它们就是系统边界定义要考虑的目标，可以称为边界目标。

12业务需求、高层次解决方案和系统特性都应该被定义到项目前景与范围文档中

13什么是涉众

所有能够影响软件系统的实现，或者会被实现后的软件系统所影响的个

人和团体

14涉众分析的主要任务

①涉众识别：寻找软件系统的涉众类别。

②涉众描述：

●描述不同涉众类别的简单特征，包括个人特征和工作特征。

●描述不同涉众类别的复杂特征，包括关注点和兴趣取向，重要性和影响力，输赢条件和受

影响程度。

③涉众评估：

●为涉众类别划分优先级。

●评估不同涉众类别的风险，化解风险。

●分析涉众浊突，实现共赢。

④涉众代表选择：从每种涉众类别中选择代表。

⑤制定涉众代表参与需求开发乃至软件系统的参与策略

15涉众识别步骤

1. 发现所有涉众类别
2. 过滤非关键涉众类别
3. 维护涉众类别

16识别涉众的方法

1. 先膨胀后收缩方法
2. 检查列表方法

在实践中人们总结出了常见的涉众类别列表，并据此指导涉众识别工作，本书将它们统称为检查列表方法。

1. 涉众网络方法

17涉众描述内容

1. 涉众的输赢条件;
2. 涉众的力量与意愿：分析涉众的力量和影响范围；分析涉众的关注点和兴趣取向
3. 涉众的个人特征与工作特征;
4. 对项目的关注点和兴趣所在，态度是反对还是赞同；
5. 对项目的期望，成为项目赢家的条件；
6. 可能受到的项目的影响，影响的具体内容及影响程度；
7. 可以对项目施加的影响，力量的施加点及其强度

18常见的涉众评估包括:优先级评估、风险评估和共赢分析

优先考虑涉众的基本特征，尤其是任务特征，使用系统更多或更重要功能、使用系统更加频繁、规模更大的用户群体可能拥有更高的优先级。

风险评估首先要分析涉众的态度，涉众的关注点和兴趣取向也是风险评估的一个重要信息内容，对具有高风险的涉众类别，要尽可能澄清各个涉众类别的角色和职责，出于化解风险的考虑，可以制定合适的项目策

共赢分析:

①列出系统的所有涉众类别，明确描述他们的兴趣和对系统的期望；

②从涉众们的兴趣和期望中发现背后涉及的共同问题(issue);

③建立涉众类别和问题的关联，如果某个涉众类别对一个issue 存在兴趣，那么该涉众类别

和这个 issue 就存在关联关系；

④对每一个 Stakeholder/Issue 关系，标明该关系上所被寄予的期望

19硬数据定义

人们在进行实际工作时会产生各种各样的表格和文档资料，这些表格和文档资料往往是用户对实际业务进行加工和抽象之后的结果，这些文档资料被称为硬数据

定量数据是指那些经过仔细设计、具有严格规范要求的格式化文档。常见的定量硬数据有以下两种。

数据收集表格，统计报表

组织中的很多文档都没有预先确定的表格，因此不是定量的。但分析它们对理解组织成员从事工作的过程还是有借鉴作用的。定性的硬数据大都是使用自然语言进行的文本描述

整个组织的描述文档，业务指导文档，业务备忘

20分析单据

1. 理清每个单据的源头
2. 理清单据的流动路径
3. 理清每个字段的前因后果
4. 注意边角上的不正规内容

21单据收集

1. 收集单据要全面
2. 一边收集一边调研
3. 只收集用过的单据
4. 每种单据需要收集多张

第七章

1用户需求获取注意事项

1. 用户需求的获取要时刻检查项目边界，在范围内的不要遗漏
2. 用户需求获取的成功依赖于合适的需求获取方法的选择与应用
3. 要准备使用多次“获取→分析”的迭代过程最终完成用户需求获取
4. 要及时将每次获取的内容组织起来，使用用例\_场景模型组织获取到的内容。

2场景的定义

将场景定义为对系统和环境行为的局部描述。

[Plihon 1998]将场景定义为对行为或者事件序列的描述，序列中的行为和事件是系统需要完成的一个任务的特殊示例。

[Jarke 1996]认为场景包含行为序列和行为发生的环境，环境描述了行为的主体、客体和上下文设置

场景具有重点描述真实世界的特征，它利用情景、行为者之间的交互、事件随时间的演化等方式来叙述性地描述系统的使用。

根据用户请求和请求时的系统条件，系统将执行不同的行为序列，每一不行为序列被称为一不场景。

3用例定义

UML将用例定义为“在系统(或者子系统或者类)和外部对象的交互中所执行的行为序列的描述，包括各种不同的序列和错误的序列，它们能够联合提供一种有价值的服务”为用例描还了在不同条件下系统对某一用户的请求所做出的响应

一个用例是多个场景的集合

4用例驱动

用例驱动的软件开发[Jacobson 2004]某种程度上就是需求驱动的软件开发。驱动项目管理、设计、构造、集成、测试、维护等后续的软件开发活动，用例/场景也能驱动它们

5场景的目的(分类)

1. 描述性场景可以用来进行需求的文档化，或者为软件开发各方的协商提供基础。
2. 探索性场景可以用来进行需求获取和需求建模与分析。
3. 解释性场景可以在需求分析时用于降低模型的复杂性，或者用于验证需求。

6用例的定位

1. 用例是静态的结构化文本描述。
2. 用例的内容可以是对当前世界的描述，对将来确定的解系统的内部行为描述以及对期待的解决方案的描述。
3. 用例可能会用于描述系统内部的交互，也可能用于描述系统和环境的交互，还可能会用于描述行为的环境和背景。
4. 用例是类型层次的事件描述，主要用来描述功能需求，可以围绕功能需求组织其他需求内容。
5. 用例的内容既包含有正常流程，又包含有异常流程
6. 用例可以用于各种目的的应用，包括描述、探索和解释。
7. 场景的各种生命周期特征、应用和处理过程都适用于用例。

7用例/场景模型开展用户需求获取

1. 依据系统用例图、目标模型建立初始用例/场景模型
2. 根据用例/场景模型指导需求获取，完善层次结构
3. 初始系统用例涉及的主题需要获取。
4. 概要用例描述中发现的新主题需要获取。
5. 具体用例中发现的模糊、不正确、不完备等细节内容需要再获取，
6. 使用用例/场景组织获取内容
7. 用新组织或修正的用例/场景完善用例/场景模型
8. 依据用例/场景模型组织需求分析模型

将系统分解为不同部分，每个部分的复杂度是可控的，这时再为这些部分分别建立需求分析模型。

1. 分析用例/场景发现仍需获取的需求内容

8用例和场景的优点

1. 以场景为单位组织用户需求（和问题域特性）
2. 很受实践者欢迎：易于接受；易于使用；用例驱动！
3. 方法多样，差异性很大：也可以用来处理业务需求和 系统级需求；还可以用来处理 设计问题、测试问题……

9用例场景描述缺点

它只考虑其他内容与功能需求之间的联系，却无法描述其他内容相互之间的联系，例如质量需求的相互依赖、界面需求的跳转、对外接口需求与质量需求的联系……

它只考虑了存在联系的事实，却无法分析联系的合理性，例如有无遗漏功能需求、数据需求及业务规则是否充分、质量需求是否可行……

第八章

1面谈过程

1. 前景和范围指导
2. 准备面谈
3. 主持面谈
4. 整理面谈报告
5. 面谈报告

2准备工作

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. 通知被会见者做准备
5. 确定问题和类型

3问题类型

开放式问题：在会见者对事实和问题的掌握比较有限，希望被会见者能够提供丰富信息时，可以使用开放式问题。

封闭式问题：在会见者对事实和问题的范围比较确定，只是希望被会见者进行选择和确认时可以使用封闭式问题。

程序性提示

探究式问题

诱导性问题

双筒问题

元问题

4问题准备

1. 保证面谈能达成目标
2. 涉众的时间往往是非常宝贵的
3. 会见者要控制面谈过程
4. 事先的准备可以减少面谈过程中的记录负担
5. 需求前期阶段的面谈中开放式问题会更多一些
6. 在需求工程的后期阶段以封闭式问题为主

5主持面谈

1. 保持有礼貌的倾听
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

6面谈的类别

在结构化面谈中，会见者会完全按照事先的问题和结构来控制面谈。

在半结构化面谈中，事先需要根据面谈内容准备面谈的问题和面谈结构。但在面谈过程中会见者可以根据实际情况采取一些灵活的策略，如调整问题的叙述方式，改变问题结构，建立场景和上下文环境等。

在非结构化面谈中没有事先预定的议程安排。在非结构化面谈中，会见者和被会见者谈话的主题可能非常广泛，而且每个主题都不会非常深入。

结构化面谈较多使用封闭式问题，所以具有封闭式问题所带来的优缺点，如易于控制、答复准确、节省时间、易于让被会见烦躁、无法获取丰富内容等。

非结构化面谈较多使用开放式问题，所以具有开放式问题所带来的优缺点，如不易控制、内容丰富等。

半结构化面谈的优缺点往往介于结构化面谈和非结构化面谈之间

7面谈的优点和局限性

面谈的优点有：

1. 面谈的开展条件较为简单，经济成本较低。
2. 能获得包括事实、问题、被会见者观点、被会见者态度和被会见者信仰等各种信息类型在内的广泛内容。
3. 通过面谈，需求工程师可以和涉众(尤其是用户)建立友好关系。
4. 通过参与面谈，被会见者会产生一种主动为项目做出贡献的感觉，提高涉众的项目参与热情。

面谈的缺点和局限性包括：

1. 面谈比较耗时，时间成本较高。
2. 在被会见者地理分散的情况下往往难以实现。
3. 面谈参与者的记忆和交流能力对结果影响较大，尤其是面谈的成功很大程度上依赖于需求工程师的人际交流能力。
4. 交谈当中常见的概念结构不同、模糊化表述、默认知识、潜在知识和态度偏见等各种问题都不可避免，进而影响面谈的效果，导致产生不充分的、不相关的或错误的数据。
5. 在会见者不了解被会见者认知结构的情况下，面谈不可能取得令人满意的效果。

8面谈相关的其他需求获取方法

调查问卷，头脑风暴

第九章

1不确定性

不确定性是指因为对未来知识了解有限而无法确定某些行为或事件的后果。

解决不确定性的方法：一种是想办法增加对未来知识的了解程度，另一种是进行风险管理

风险是指因为不确定性而可能给未来造成的损失。

2原型的定义

如果在最终的制品产生之前，一个中间制品被用来在一定广度和深度范围内表现这个最终制品，那么这个中间制品就被认为是最终制品在该广度和深度上的原型。

原型是一个系统，它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、补充或者替代。

包括书面描绘、场景叙述、故事板、幻灯演示、动画模拟、屏幕快照和程序代码

3原型的用途

原型根据用途的不同概括为：演示原型、严格意义上的原型、试验原型和引示系统原型

演示原型通过演示相应的内容展示开发者的技术能力或特定问题解决的可能性，主要用在

项目启动阶段。

严格意义上的原型主要用在需求分析阶段，用来阐明用户界面或系统功能的某些特定方面，帮助人们及时澄清问题

试验原型主要用在构建系统阶段，用来帮助开发者澄清他们所面对的一些和系统构建相关的技术问题

引示系统原型而是被用作最终系统的构建核心，那么该原型就属于引示系统原型

4使用原型法进行需求获取基本过程

1. 确定原型需求
2. 原型开发。
3. 原型评估。
4. 原型修正。

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。一个系统的所有需求之中只会有部分需求不确定，也只需要针对该部分需求建立原型，以控制原型开发的成本

开发原型时的主要注意事项有以下几点：

1. 将探索不确定功能需求的原型构建得易于修改。
2. 让探索可行性的原型收集充分的数据
3. 控制开发成本。
4. 使用脚本指导评估过程
5. 创造无偏见的评估环境
6. 引导评估者从恰当的角度进行评价
7. 观察评估者的行为
8. 收集评估者的反馈

原型修正一方面要依据评估人员的反馈，另一方面也要考虑事先的原型调整计划

5原型开发方法分类

探索式。如果开发者对用户的一些问题和需求有了一定的了解，但是了解的内容非常模糊或者不充分。这种以缺陷需求开始，继而不断调整和修正需求的原型开发方式称为探索式。

实验式的原型方法初始时就拥有清晰的用户需求，但是开发者对这些需求的实现方法、实现效果和可行性没有太大的把握

演化式的原型方法中，原型的开发并不是一个独立的活动，而是整个项目的持续开发过程中的一个部分。原型开发的初始点既有要求原型化的需求，也有项目积累下来的原型资产

探索式和实验式方法产生的原型产品又被称为抛弃式原型，以演化式原型方法产生的原型产品被称为演化式原型

6认为系统的构建是对其多个层次，并据此将原型的构建方法分成以下两种类型

水平原型方法( horizontal prototyping)。水平原型方法仅仅实现选定功能所有层次中的某些特定层次(如用户界面层),它能够处理较大范围的功能，建立的原型产品称为水平原型

垂直原型方法(vertical prototyping)。垂直原型方法会触及选定功能实现的所有层次，处理的功能范围通常较小，建立的原型产品称为垂直原型

7常见的代表性介质有纸面、幻灯动画、快速语言工具和程序语言。

静态画面表现出来的是静态的结构，如图形界面

动态程序本身就是软件系统，它能够表现所有软件可以表现的一切，所有类型的人机互动在动态程序中都能够看到。

故事板则是交互性介于静态画面和动态程序之间的一种原型表现形式，它能够表现场景式的互动

8故事板原型类型

被动(passive)故事板原型：被动故事板原型的特点是给评估者“讲”故事，

主动(active)故事板原型：主动故事板原型的特点是让评估者“看”故事，

交互(interactive)故事板原型：交互故事板原型的特点是让评估者“体验”故事

9原型方法的风险

1. 对原型开发工作投入太多，消耗了过多的时间和成本
2. 客户看到一个正在运行的原型时，难免会得出产品已经基本完成的结论，从而提出快速交付产品的不当要求
3. 用户可能会被原型所表现出来的非功能特性遮蔽了眼睛，从而忽略了他们更应该重视的功能特性。
4. 在澄清需求不确定性的同时也可能会掩盖一些用户假设，这些假设将会无从发现。

10利用原型的好处有

1. 及时、有力的响应用户需求的变化；
2. 帮助控制不完整需求所带来的风险；
3. 可以将一个大的难以处理的开发过程细分成一些更小、更容易处理的步骤；
4. 减少返工和开发成本，提高经济效益；
5. 有效的识别风险和解决需求的不确定性；
6. 提高用户在软件开发中的参与程度。

11帮助需求工程师及早解决需求的不确定性

1. 创新性产品，它们的**基本需求是潜在的**，有着很大的不确定性；
2. 产品的用户对相关类别的产品没有经验，**产品的细节需求**存在着不确定性；
3. 用户在完成工作的方式上仍然存在障碍，产品在整体方案的可行性上**存在着不确定性**；
4. 用户在**清晰说明他们的需求方面存在困难**；
5. 需求工程师在**理解用户的需求上存在困难**；
6. 需求的**可行性值得怀疑**。

第十五章

1需求规格说明活动就是将需求及软件方案进行定义和文档化，以有效将信息传递给开发人员的需求工程活动。

2编写需求规格说明文档的原因

1. 清晰、明确、结构化的文档可以将软件系统的需求信息和解决方案更好地传递给所有开发者。
2. 文档可以拓展人们的知识记忆能力。
3. 需求规格说明文档可以成为各方人员之间有关软件系统的协议基准
4. 需求规格说明文档可以成为项目开发活动的一个重要依据
5. 在需求规格说明文档的编写过程中，可以尽早发现和减少可能的需求错误，从而减少项目的返工，降低项目的工作量。
6. 需求规格说明文档可以成为有效的智力资产。

3需求规格说明文档的读者

项目管理者。设计人员和程序员。测试人员。文档编写人员。维护人员。培训人员。律师。

4需求规格说明文档的描述手段

非形式化语言，即自然语言。常利于用户的理解。但同时自然语言也具有松散、模糊、歧义、凌乱等缺点，这使得它无法被计算机所理解

半形式化语言，比自然语言具有更丰富的语义和更严格的语法，同时又没有严格到可以完全基于数学方法的语言，如数据流图、UML等图形语言。

半形式化语言具有严格的语法，定义方式比自然语言更加严格，这使得它可以避免自然语言模糊、松散、歧义、凌乱等缺。

半形式化语言具有丰富的语义，使用规则比形式化语言更复杂和多样，这使得它具有比形式化方法更强的表达能力。

它描述的信息还需要进行额外的处理才能够被计算机所理解或者准确映射为计算机行为

形式化语言，基于数学的语言，如VDM和Z语言等，具有数学的表示法特性。形式化的信息描述能够被计算机所理解，它所描述的信息内容可以准确映射为计算机行为。

大多数用户无法读懂以形式化方法描述的信息

半形式化语言的语法限制使得它用于信息描述的基本元素是有限的，这个有限性使它以限定文本或限定图形符号为描述方式成为可能。

半形式化语言追求表达语义的丰富性，而在这一点上图形符号是胜过限定文本的，所以人们倾向于选择使用图形符号的描述方式

用户倾向于使用自然语言

开发人员倾向于使用半形式语言和形式化语言

5写作的指导原则

写作是一门艺；文档化的目标是交流

6常见的写作技巧

内容的组织：所有内容位置得当；引用或强化，但不重复

表达方式：形式依赖于内容；使用系统的表达方式

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

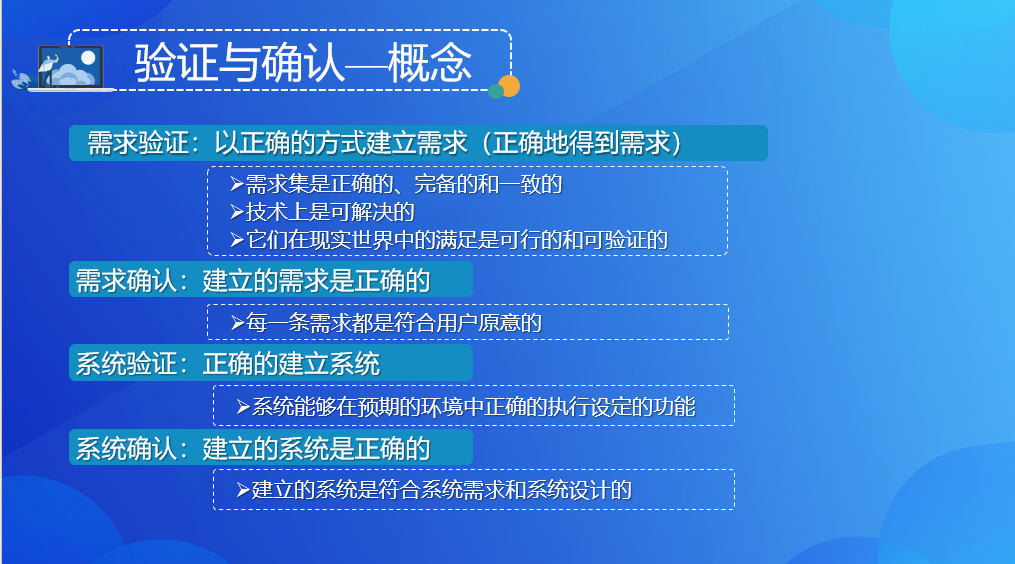
7优秀需求规格说明文档的特性

完备性；一致性；根据重要性和稳定性分级；可修改；可跟踪

第十六章

1确认与验证

一方面，它要确保正确地得到需求(需求验证),得到足以作为软件创建基础的需求；另一方面，它要确保得到正确的需求(需求确认),得到能够准确反映用户意图的需求。



2需求验证活动

1. 在需求获取中：获得的用户需求是否正确?是否充分地支持业务需求?
2. 在需求分析中：建立的分析模型是否正确反映了问题域特性和需求?细化的系统级需求是否充分和正确的支持用户需求?
3. 需求规格说明：需求规格说明文档是否组织良好、书写正确?需求规格说明文档内的需求是否充分和正确地反映了涉众的意图?需求规格说明文档是否可以作为后续开发工作(设计、实现、测试等等)的基础?

3需求验证的方法

1. 需求评审

指由作者之外的其他人来检查产品问题。是主要的静态分析手段, 原则上，每一条需求都应该进行评审。

评审的过程:

在规划阶段(planning),作者和仲裁者共同制定评审计划，决定评审会议的次数，安排每次评审会议的时间、地点、参与人员和评审内容等。

在总体部署阶段(overview),作者和仲裁者向所有参与评审会议的人员描述待评审材料

的内容、评审的目标以及一些假设，并分发文档。

在准备阶段(preparation),评审人员各自独立执行检查任务。在检查的过程当中，他们可能会被要求使用检查清单、场景等检查方法。检查中发现的问题会被记录下来，以准备开会讨论或者提交给收集人员。

在评审会议阶段(inspection meeting),通过会议讨论，识别、确认、分类发现的错误。在评审会议结束时，还可以根据评审发现的问题严重度来确定软件需求规格说明文档是可以在修正后接受，还是需要在修正后再次进行评审。

在返工阶段(rework),作者修改发现的缺陷。

在跟踪阶段(follow-up),仲裁者要确认所有发现的问题都得到了解决，所有的错误都得到了修正。仲裁者还要判断修正后的文档是否已满足评审的结束标准，如果不满足就需要再次 进行评审

评审的检查方法:自由方法,检查清单,缺陷,功能点,视角,场景,逐步提升

评审的类型: 审查, 小组评审, 走查, 轮查, 临时评审

1. 原型与模拟: 当有些需求涉及复杂的动态行为时，它可能就需要使用原型或模拟方法来加以验证
2. 开发测试用例: 如果无法为某条需求定义完备的测试用例，那么它可能就存在着模糊、信息遗漏、不正确等缺陷。除了排斥性需求（这种需求要求特定的行为绝对不会发生）, 非功能需求，如可靠性、可用性等。
3. 用户手册编制
4. 利用跟踪关系

如果不能依据跟踪关系找到一条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。

如果业务需求和用户需求没有得到后项需求(用户需求和系统级需求)的充分支持，那么软件需求规格说明文档就存在不完备的缺陷。

1. 自动化分析

4问题的修正过程

1. 需求澄清
2. 理解当中出现了偏差。这时需要需求工程师重新进行分析工作
3. 需求信息没有被纳入需求分析或者文档化工作。这要求需求工程师重新分析和文档化这部分信息。
4. 在文档化的过程当中使用了不恰当的表达方式
5. 发现缺失需求
6. 解决需求冲突
7. 修正不切实际的期望

第十七章

1需求管理的作用有

1. 增强了项目涉众对复杂产品特征在细节和相互依赖关系的理解。
2. 增进了项目涉众之间的交流。
3. 减少了工作量的浪费，提高了生产力。
4. 准确反映项目的状态，帮助进行更好的项目决策
5. 改变项目文化，使需求的作用得到重视和有效发挥

2需求管理的重要任务

1. 交流涉众需要什么。
2. 将需求应用、实施到解决方案。
3. 驱动设计和实现工作。
4. 控制变更。
5. 将需求分配到子系统。
6. 测试和验证最终产品。
7. 控制迭代式开发中的变化。
8. 辅助项目管理。

3需求基线定义

需求基线就是被明确和固定下来的需求集合，是项目团队需要在某一特定产品版本中实现的特征和需求集合

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，并且只有通过正式的变更控制过程才能修改它

4需求的状态类别

已提议，已批准，已实现，已验证，已删除，已否决

5前向跟踪

前向跟踪是指被定义到软件需求规格说明文档之前的需求演化过程

向前跟踪到需求：说明涉众的需要和目标产生了哪些软件需求。

从需求向后回溯：说明软件需求来源于哪些涉众的需要和目标。

6后向跟踪

后向跟踪是指被定义到软件需求规格说明文档之后的需求演化过程。

从需求向前跟踪：说明软件需求是如何被后续的开发物件支持和实现的。

回溯到需求的跟踪：说明各种系统开发的物件是因为什么原因(软件需求)而被开发出来的。

7需求跟踪的用途

①需求的后向跟踪可以帮助项目管理者：

●评估需求变更的影响。

●尽早发现需求之间的冲突，避免未预料的产品延期。

● 可以收集没有被实现的需求，并估算这些需求需要的工作量。

②需求的后向跟踪可以帮助客户和用户：

●评价针对用户需求的产品的质量。

●可以确认成本上没有(昂贵的)镀金浪费

●确信开发者的关注点始终保持在需求的实现上

③需求跟踪中针对具体需求的设计方案选择、设计假设条件以及设计结果等信息可以帮助 设计人员：

●验证设计方案正确的满足了需求。

●评估需求变更对设计的影响。

●在设计完成很久之后仍然可以理解设计的原始思路。

④需求跟踪信息还可以帮助维护人员：

●评估某一个需求变化时对其他需求的影响。

●评估需求变化时对实现的影响。

● 评估未变化需求对实现变更的允许度

8需求跟踪过程的建立

1. 认识到需求跟踪的重要性，明确需求跟踪需要解决的问题。
2. 说明需求跟踪过程的目标。
3. 明确需要捕获的跟踪联系。
4. 组织提供资源支持和技术支持。
5. 制定有效的过程策略。
6. 便利需求跟踪信息的使用。

9需求依赖

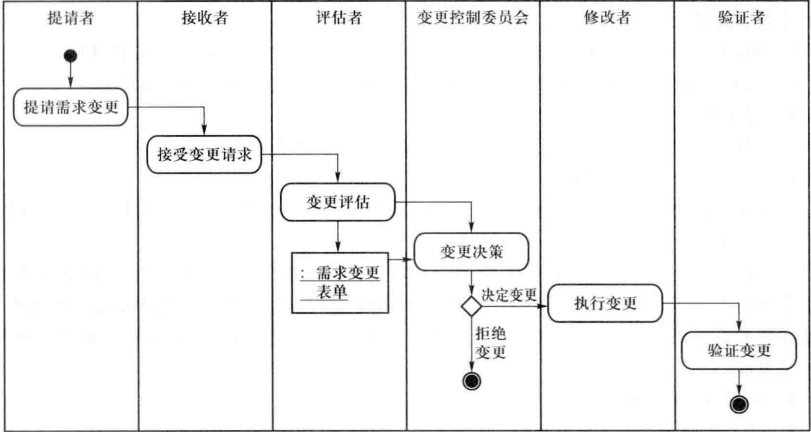
需求依赖联系的特殊性并不在于它的重要性，而在于它是难以发现、建立和维护的。

对需求之间依赖联系的处理被称为需求交互作用管理

10需求变化的原因

1. 问题发生了改变
2. 环境发生了改变
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. 相关产品的出现

11变更控制过程



12变更控制中的注意事项

1. 认识到变更的必要性，并为之制定计划
2. 维护需求基线，审计变更记录
3. 管理范围蔓延
4. 灵活应对变更请求
5. 使用辅助工具

13组建需求团队

1. 团队成员的技能分布要完备
2. 团队成员应该尽职尽责，能够耐心和持之以恒的处理各种复杂任务。
3. 团队成员应该具备出色的交流能力和沟通技巧
4. 团队成员应该能够互相信任，要将固执己见的人员排出于团队之外。
5. 团队成员的数量要适中
6. 尽量纳入在客户方有决策权的人
7. 对需求团队进行必要的培训，尤其是那些初次参与需求工程活动的成员。
8. 核心的需求工程师要足够称职，要具备很多的知识和技能。

14维持需求团队内部的有效沟通

1. **建立一致的目标**

**每个团队成员可以拥有自己的目标，不能和共同目标相冲突**

**项目前景和范围**

1. **建立有效的沟通机制**

**定期的会议，需求管理**

1. **利用有效的沟通技巧**

**提高积极性：自备食品的聚餐，小的集体娱乐活动，等等。**

1. **利用辅助的工具和技术**
2. **方便协同工作的工具和技术**

15风险产生的原因是对未来不利事件的不确定性

风险管理就是管理风险的活动，它关注软件开发活动和任务的风险和不确定性，并采取行动减少其中的不确定性或者降低风险的影响范围

