第一讲：绪论—需求工程与软件需求

1. 软需缺陷纠正的成本

重新进行规格说明；重新设计；重新编码；重新测试；

版本升级；纠正活动；报废；回收成本；

保修成本；产品赔偿；服务成本；建档成本；

2. 软件开发和维护方法的不正确性主要体现在：

忽视软件开发前期的需求分析，包括缺乏先进的需求开发技术和规范的需求管理过程等；

开发过程缺乏统一的、规范化的方法论的指导；

文档资料不齐全或不准确；

忽视与用户之间、开发组员之间的交流；

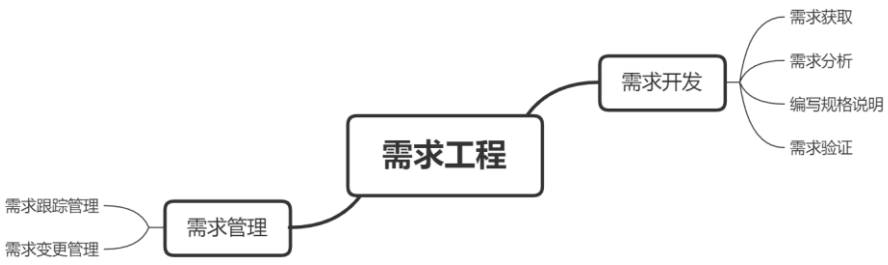
忽视测试的重要性；

不重视维护或由于上述原因造成维护工作的困难。

3. 需求工程的概念

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。RE 可分为系统需求工程和软件需求工程。

需求工程是一个不断反复的需求定义、文档记录、需求演进的过程，并最终在验证的基础上冻结需求。



4. 软件需求工程的概念

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。

5. HerbKrasner 定义的需求工程的五阶段生命周期：

需求定义和分析、需求决策、形成需求规格、需求实现与验证、需求演进管理。

6. Matthias Jarke 和 Klaus Pohl 提出的三阶段周期的说法：

获取、表示和验证。

7. 综合几种观点对需求工程活动的划分:

需求获取、需求建模、形成需求规格、需求验证、需求管理

8. IEEE 软件工程标准词汇表(1997 年)中定义需求为：

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
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。

9. 本课程对软件需求的定义

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。通过对应问题及其环境的理解与分析，为问题涉及的信息、功能及系统行为建立模型，将用户需求精确化、完全化，最终形成需求规格说明，这一系列的活动即构成软件开发生命周期的需求分析阶段。

补充 9. 软件需求在整个软件开发过程中的作用

对于整个开发过程来说，需求分析是介于系统分析和软件设计阶段之间的桥梁。

一方面，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。

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。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。

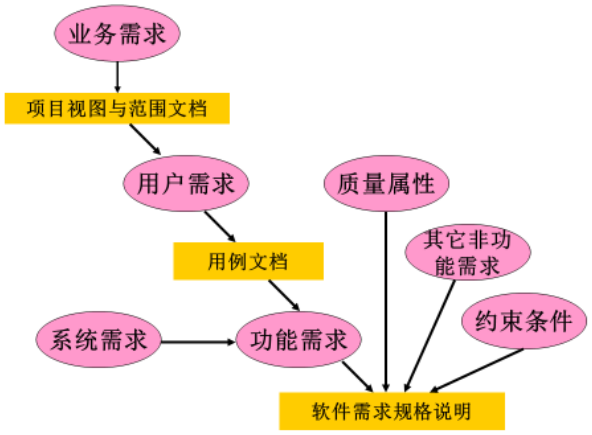
10. 软件需求的层次划分

软件需求包括三个不同的层次:

- 业务需求 (business requirement)：反映了组织机构或客户对系统、产品高层次的目标要求，它们在项目视图与范围文档中予以说明。
- 用户需求(user requirement)：描述的是用户的目标，或用户要求系统必须要完成的任务。用例(use case)文档、场景描述(scenario)和事件—响应表均用于表达用户需求。
- 功能需求(functional requirement)：定义了开发人员必须实现的软件功能，使得用户能利用这些功能完成他们的任务，从而满足了业务需求。

作为功能需求的补充，软件需求规格说明还应包括非功能需求，它描述了系统展现给用户的行为和执行的操作等。

软件需求各组成部分之间的关系如图所示。



在项目中它们在不同的时间来自不同的来源，也有着不同的目标和对象，并需以不同的方式编写成文档。

- 业务需求(或产品视图和范围)不应包括用户需求(或使用实例)。
- 而所有的功能需求都应该源于用户需求。
- 同时也需要获取非功能需求，如质量属性 。

11. 非功能性需求：

作为功能需求的补充，它描述了系统展现给用户的行为和执行的操作等。包括外部界面细节、性能要求及质量属性。

补充 11. 功能需求是什么：

功能需求定义了开发人员必须实现的软件功能，使得用户能利用这些功能完成他们的任务，从而满足了业务需求。

例子， 如某 web 项目的功能需求：

1）用户注册模块：包括注册、登录及修改注册信息功能。

2）课程视频模块：包括上传视频、下载视频等功能

12. McCall 提出的 11 个质量特性：

运行：正确性，可靠性，效率，完整性，易用性；

修正：可维护性，测试性，灵活性；

转移：可移植性，可重用性，互操作性

13. 设计约束：

所谓约束是指对开发人员在软件产品设计和构造上的限制。

补充 13.需求工程中需要考虑到哪些约束问题？

约束条件制约着开发人员对设计或者实现的选择。

约束可以由外部的相关人员提出，也可以产生自其他系统(这与正在创建或者维护的系统有交互)或者是来自系统生命周期中的一些活动(比如事务和维护）。其他的约束来自于已有的约定、管理决策和技术选择（ISO/IEC/IEEE 2011）。约束来源包括以下几类。

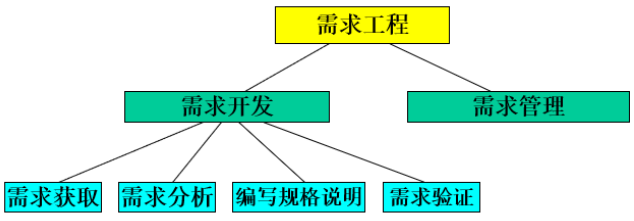
- 必须要用或者要避免的特定技术、工具、语言和数据库。
- 因产品操作环境或者平台而产生的约束,比如要用的网页浏览器或者操作系统的类型和版本。

- 需要遵循的开发约定或者标准。(比如，如果客户所在的企业要维护软件,可能就会指定子承包商必须遵循的设计表示方法和编码标准。)早期产品的向后兼容与潜在的向前兼容能力，比如，为了创建某个特定的数据文件,需要知道正在使用哪一个软件版本。
- 受法律规章或者其他业务规则决定的限制性或者合规性需求。
- 硬性限制，比如时间需求，内存或者处理器限制，大小，重量，材料或者成本。
- 由操作环境或由用户特性和局限所造成的物理性约束。对现有产品进行优化时要遵循的现有接口约定。
- 与其他现有系统的接口,比如数据格式和通信协议。
- 显示屏幕尺寸的限制，比如需要在平板电脑或者手机上运行时。使用的标准数据交换格式，比如电子商务使用的 XML 或者 RosettaNet 。

约束也可能是在不经意中提出的。比较普遍的是用户描述“需求”时，他们描述的实际上是为满足想象中的某个需要而讲述的某个特殊解决方案。作为业务分析师，必须从包括类似解决方案的需求中将其识别出来,并分离出由此方案产生的与约束条件相关的需求。用户想象的解决方案实际上可能是解决问题的理想方法，这种情况下的约束条件非常正当合理。更常见的情况是实际需求是隐藏的需求，业务分析师必须与用户一起协作，让其描绘出导致提议方案的实际想法。多问几个“为什么”，挖掘出真实的需求。

14. 需求工程的构成

需求工程是由一系列与软件需求有关的活动组成，包括软件需求开发活动和需求管理活动两部分，如图所示（书中图 2.1）：



需求工程内容及层次分解

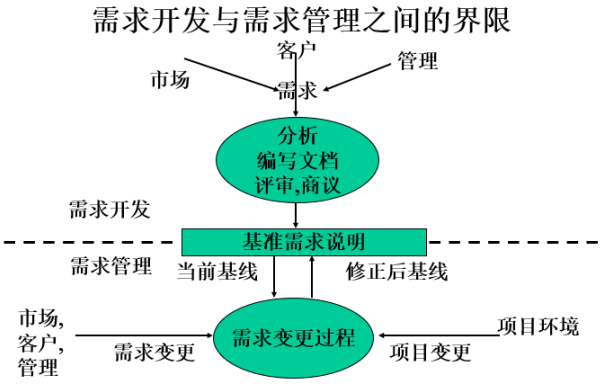
（1）需求开发中的问题获取(elicitation)、需求分析(analysis)、编写规格说明(specification)和需求验证(verification)四个阶段包括了软件产品中需求收集、评价、编写文档等所有活动。需求开发活动包括以下几个方面：

- 确定产品所期望的用户类。
- 获取每个用户类的需求。
- 了解实际用户任务和目标以及这些任务所支持的业务需求。

- 分析源于用户的信息以区别用户任务需求、功能需求、业务规则、质量属性、建议解决、方法和附加信息。

（2）需求管理包括需求跟踪、变更需求和基线管理。需求管理的任务是“与客户就软件项目的需求达成并保持一致”。

掌握：



15. 需求开发过程包括：

确定产品所期望的用户类；

获取每个用户类的需求；

了解实际用户任务和目标以及这些任务所支持的业务需求；

分析源于用户的信息以区别用户任务需求、功能需求、业务规则、质量属性、建议解决、方法和附加信息。

16. 需求获取

（1）需求获取包括的主要活动

确定需求开发过程；

编写项目视图和范围文档；

用户群分类；

选择产品代表；

建立核心队伍；

确定用例；

召开应用程序开发联系会议；

分析用户工作流程；

确定质量属性和其他非功能性需求；

检查问题报告；

需求重用；

……

（2）需求获取的方法和技能

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

（3）主要任务

收集材料、定义项目视图和范围、选择信息来源、选择获取方法并执行获取、记录获取结果。

（4）可能出现的主要问题

- 捕获范围不足
- 缺乏计划性
- 缺乏科学性
- 获取对象不明确

17. 需求分析的主要任务

背景分析、确定系统边界、

需求建模（数据流图、E-R 图、状态转换图、类图等描述需求）、

需求细化、确定优先级、需求协商、

绘制关联图、原型开发、数据字典创建、

子系统建立（建立系统结构，将需求分配到各个子系统和模块中）

需求分析和需求获取交替进行。

18. 编写需求规格说明

就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。

主要任务是：

- 定制文档模板

- 编写文档

主要方法包括：采用软件需求规格说明模板、为每项目标注标号、记录业务范围、创建需求能力跟踪能力矩阵、设计数据字典、数据流图、状态转化图、对话图和类图等。

需求规格说明书应该具有：

- 共享性：可获得；可获知
- 更新性

19. 需求验证

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。验证具体包括：

- 审查需求文档（关键手段）
- 依据需求编写测试用例
- 编写用户手册
- 确定合格标准

需要验证需求规格文档至少符合以下标准：

- 文档内每条需求都准确地反映了用户的意图；
- 文档记录的需求集在整体上具有完整性和一致性；
- 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

需求验证阶段的任务：

- 执行验证：同级评审
- 问题修正：发现问题及时修正

20. 需求管理

需求管理包括需求跟踪、变更需求和基线管理。

需求管理阶段任务和活动包括：

- 建立和维护需求基线。
- 建立需求跟踪信息。
- 后向跟踪和前向跟踪：让每项需求都能与其源头、对应的设计、源代码和测试用例联系起来以实现跟踪。

21. 导致发生不合格需求说明的情况：

无足够用户参与；

用户需求的不断增加；

模棱两可的需求；

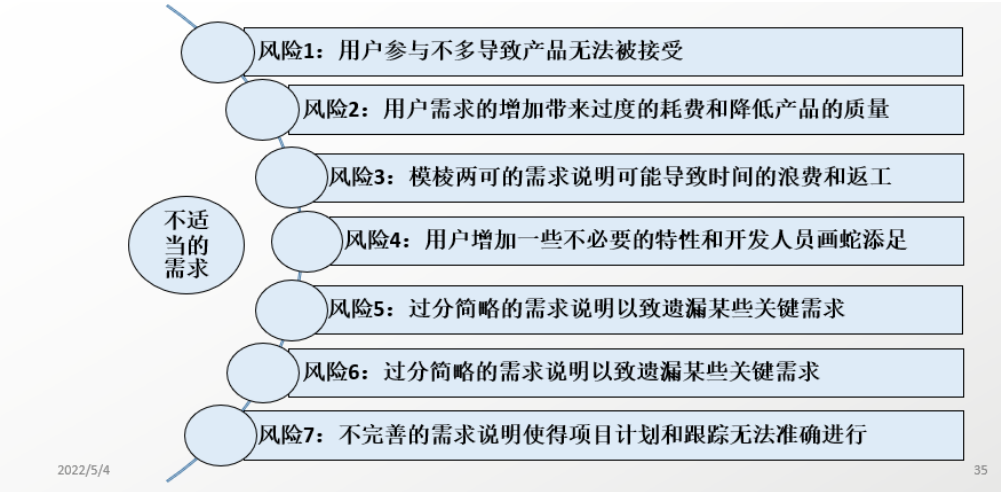
不必要的特性；

过于精简的规格说明；

忽略了用户分类；

不准确的计划

补充 21.不合格的需求会派生的问题



22. 良好需求具有的特性

（1）每一项需求都应该具备下列特性：

完整性、正确性、可行性、必要性、

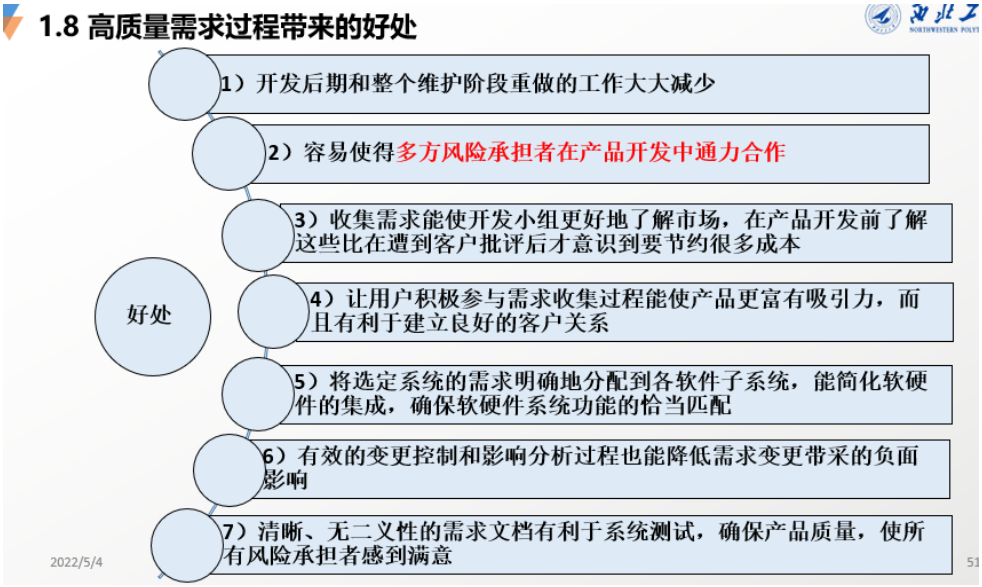
划分优先级、无二义性、可验证性、

一致性、可修改性、可跟踪性

（2）整个需求规格说明必须具备的特性

一致性、可修改性、可跟踪性

补充 22. 成功的需求的好处



第二讲：需求获取

以下重点探讨的内容是：

- 确定需求开发过程（省略）。
- 编写项目视图和范围文档。
- 将用户群分类并归纳其特点，为每个用户类选择产品代表(product champion)。
- 确定用例（用户代表）。
- 确定质量属性和其它非功能需求。

23. 建立项目视图与范围

- 在项目业务需求、用户需求和功能需求（含非功能需求）等三个层次构成的需求链中，业务需求代表了最高层的抽象：他们为软件系统定义了项目视图(Vision)和范围(scope)。
- 一个明确定义了项目视图和范围的文档可以为需求变更的决策提供参考。

（1）通过业务需求确定项目视图

- 项目视图可以把项目参与者定位到一个共同和明确的方向上。项目视图描述了产品所涉及的各个方面和在一个确定环境中最终所具有的功能。而范围描述了产品应包括的部分和不应包括的部分。范围的说明在包括与不包括之间划清了界线，当然，它还确定了项目的局限性。
- 项目的业务需求在视图上和范围上形成文档，这些必须在创建项目之前起草。
- 业务需求是从各个不同的人那里收集来的，这些人对于为什么要从事该项目和该项目最终能为业务和客户提供哪些价值有较清楚的了解。

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。

（2）项目视图和范围文档

包括了业务机遇的描述、项目的视图和目标、产品适用范围和局限性的陈述、客户的特点、项目优先级别和项目成功因素的描述。

24. 需求的来源

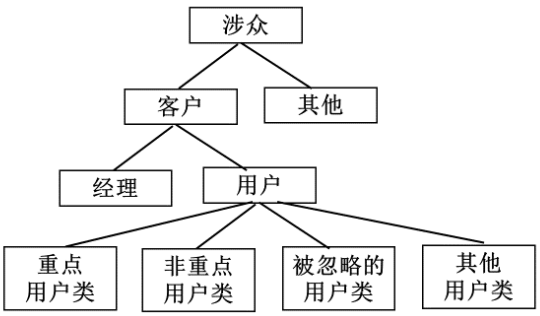
与潜在用户进行交谈和讨论、
把对现有产品或竞争产品的描述写成文档、
系统需求规格说明、
现有系统的问题报告和改进要求、
市场调查和用户问卷调查、
观察用户如何工作、
用户工作的情景和内容分析、
事件和响应

25. 用户类

每一个用户类都将有自己的一系列功能和非功能要求。
有一些受产品影响的人并不一定是产品的直接使用者，而是通过报表或其它应用程序访问产品的数据和服务。这些非直接的或次级(secondary)的用户也有他们的需求，于是他们组成了附加的用户类。

用户类不一定都指人，有时应把其它应用程序或系统接口所用的硬件组件也看成是附加用户类的成员。

26. 涉众、客户、用户、用户类的关系



用户类是产品用户的子集，产品用户则是产品客户的子集，而产品客户又是产品涉众的子集。

27. 用户代表

（1）寻找用户（产品）代表

- 每种类型项目——包括企业信息系统、商业应用程序、集成系统、软件嵌入式产品、Web 开发程序和签约合同的软件，在获取(elicitation)用户需求时需要挑选合适的用户代表来反映各类用户的需求。
- 用户代表必须参加整个软件开发生存周期，而不仅仅是只参加开始的需求阶段。

（2）用户（产品）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)。

通过产品代表这一途径，可以提供一个有效的方法来使客户——开发者之间的伙伴关系结构化和形式化。

每一个产品代表代表了一个特定的用户类，并在那个用户类和开发者之间充当主要的接口。

产品代表者必须是真正的用户，而并不是用户的代理人，如主办者、产品客户、市场人员或者软件组成员充当用户。

产品代表从他们所代表的用户类中收集需求信息。

每个产品代表都负责协调他们所代表的用户在需求表达上的不一致性和不兼容性。目的就是由每个产品代表者与分析员合作，为那个用户类整理出统一的需求意见。

（3）对产品代表的要求

计划-需求-验证和确认-用户帮助-变更管理

28. 用例化方法：什么是用例化方法，用例的模型

- 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。

基于“用例”方法进行需求获取的目的在于：

- 描述用户需要使用系统完成的所有任务。
- 在理论上，用例的结果集将包括所有合理的系统功能。
- 应用基于用例的方法进行需求获取，可以为用户和开发者带来更好的效果。

（1）用例和用法说明

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。因此，一个用例是相关的用法说明的集合，并且一个说明(scenario)是用例的例子。

在用例中，一个说明被视为事件的普通过程(normal course)，也叫作主过程、基本过程，普通流。

在描述普通过程时列出执行者和系统之间相互交互或对话的顺序。

当这种对话结束时，执行者也达到了预期的目的。

- 用例图提供了一个高级别的用户需求的可视化表示。

（2）确定用例并编写用例文档

可以使用多种方法来确定用例：

- 首先明确执行者和他们的角色，然后确定业务过程，在这一过程中每一个参与者都在为确定用例而努力。
- 确定系统所能反映的外部事件，然后把这些事件与参与的执行者和特定的用例联系起来。
- 以特定的说明形式表达业务过程或日常行为，从这些说明中获得用例，并确定参与到用例中的执行者。
- 有可能从现在的功能需求说明中获得用例。如果有些需求与用例不一致，就应考虑是否真的需要它们。

（3）用例和功能需求

- 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。

（4）用例的益处

用例方法给需求获取带来的好处来自于该方法是以任务为中心和以用户为中心的观点。

- 比起使用以功能为中心的方法，用例方法可以使用户更清楚地认识到新系统允许他们做什么。
- 用例有助于分析者和开发者理解用户的业务和应用领域。认真思考执行者——系统对话的顺序，使其可以在开发过程早期发现模糊性，也有助于从用例中生成测试用例。

有了用例，所得到的功能需求明确规定了用户执行的特定任务。

在技术方面，用例的方法也带来了好处。开发者运用面向对象的设计方法可以把用例转化为对象模型。

如果跟踪功能需求、设计、编码和测试以至到它们父类的用例——用户意见，那么这就很容易看出整个系统中业务过程的级联变化。

（5）在用例的方法中应注意如下的陷阱：

太多的用例；
用例的冗余；
用例中的用户界面的设计；
用例中包括数据定义；
试图把每一个需求与一个用例相联系
用户无法理解用例
滥用包涵和扩充关系

29. 质量属性

- 用户总是强调确定他们的功能、行为或需求——软件让他们做的事情。除此之外，用户对产品如何良好地运转抱有许多期望。这些特性包括：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（quality attribute）或质量因素（quality factor），是系统非功能(也叫非行为)需求的一部分。

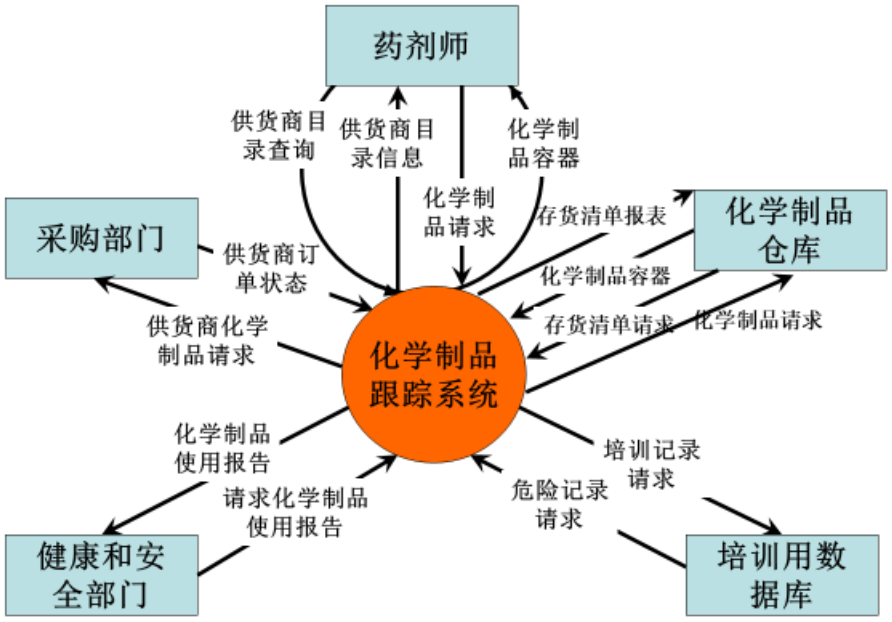
第三讲：需求分析

30. 需求分析中的主要任务包括：

- 绘制系统关联图。
- 建立数据字典。
- 建立用户界面（接口）原型。
- 为需求建立模型。
- 确定需求优先级。

31. 系统关联图

- 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。



化学制品跟踪系统关联图

8

32. 原型法：构成，在需求开发过程中原型法的种类划分，有哪些风险，什么是原型，原型化的作用

（1）什么是原型？

软件原型是所提出的新产品的部分实现或可能的实现。

（2）为什么要建立原型？

作为一种需求工具：明确并完善需求；

作为一种设计工具：探索设计选择方案；

作为一种构造工具：发展为最终的产品原型。

- 建立原型的主要原因是为了解决在产品开发的早期阶段不确定的问题。

（3）在需求开发过程中原型法的种类划分

①水平原型（行为原型/演示模型）

这种原型中所提出的功能经常并没有真正地实现。

它仅包含少量的功能并没有真正实现所有的功能。

②垂直原型(vertical prototype)（“结构化原型”/“概念的证明”）

垂直原型实现了一部分应用功能。

当不能确信所提出的构造软件的方法是否完善或者当需要优化算法，评价一个数据库的图表或测试临界时间需求时，就要开发一个垂直原型。

③抛弃型原型

抛弃型原型或探索型原型在其达到预期目的以后将会被抛弃，所以应以花最小的代价尽快地建立该原型。

④进化型原型

在已经清楚地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。

⑤书面原型和电子原型

表3.1 软件原型的典型应用

	抛弃型	演化型
水平原型	●澄清并细化使用实例和功能需求 ●识别遗漏的功能 ●研究用户界面方法	●实现核心的使用实例 ●根据优先级实现附加的使用实例 ●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	●实现并扩充核心客户端／服务器功能层和通信层 ●实现并优化核心算法 ●测试并调整性能

表3.1总结了抛弃式、演化式、水平和垂直原型的一些典型应用。

（4）原型化的作用

①减少客户对产品不满意的风险。

②原型可以使新产品实在化，为用例带来生机，消除在需求理解上的差异。

（5）原型法的风险

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。

33. （原整理内容）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。原型可以使新产品实在化，为用例带来生机，并消除在需求理解上的差异。一个软件原型通常仅仅是真实系统的一部分或一个模型，并且它可能根本不能完成任何有用的事。

建立原型的目的：

作为一种需求工具：明确并完善需求；

作为一种设计工具：探索设计选择方案；

作为一种构造工具：发展为最终的产品原型。

建立原型的主要原因是为了解决在产品开发的早期阶段不确定的问题。

水平原型也叫做行为原型或演示模型。它可以让你探索预期系统的一些特定行为，并达到细化需求的目的。

垂直原型也叫做结构化原型或概念的证明，实现了一部分应用功能。可以使用户评估所提出体系结构的通信组件、性能和可靠性还有核心算法的有效性等。

抛弃型原型或探索型原型在其达到预期目的以后将会被抛弃，所以应以花最小的代价尽快地建立该原型。用来解决不确定性并提高需求质量。当遇到需求中的不确定性、二义性、不完整性或含糊性时适应运用

进化型原型在已经清楚地定义了需求的情况下，为开发渐增式产品提供了坚实的构造基础，是螺旋式软件开发生存周期模型的一部分，一开始就必须具有健壮性和产品质量级的代码，比建立抛弃型原型所花的时间要多。

34. 原型法成功的因素：

项目计划中应包括原型风险。

安排好开发、评价和可能的修改原型的时间；

计划开发多个原型，因为很少能一次成功；

尽快并且廉价地建立抛弃型原型；

在抛弃型原型中不应含有代码注释、输入数据有效性检查、保护性编码技术，或者错误处理的代码；

对于已经理解的需求不要建立原型；

不能随意地增加功能；

不要从水平原型的性能推测最终产品的性能；

在原型屏幕显示和报表中使用合理的模拟数据；

不要期望原型可以代替需求文档

35. 需求建模

- 需求的图形化表示的模型包括：
- 数据流图(DFD)、实体关系图(ERD)、状态转化图(STD)、对话图和类图。

36. 数据流图

- 数据流图(data flow diagram，DFD)是结构化系统分析的基本工具。

- **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。**
- 数据流图描述了软件需求规格说明中的功能需求怎样结合在一起使用户可以执行指定的任务。

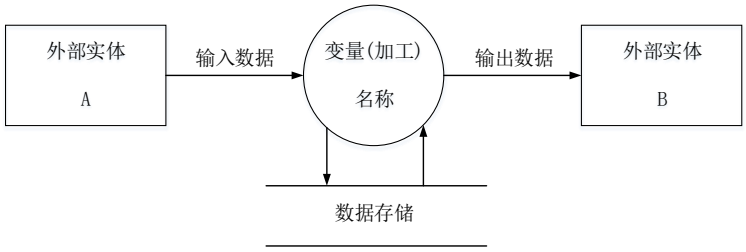


图 1　数据流图的基本形式

关联图是数据流图最高层的抽象。关联图把整个系统表示成一个简单的黑匣子的过程，并用一个圆圈表示。关联图还表示出外部实体或与系统有关的端点，以及在系统与端点之间的数据和物质流。在关联图中，端点用矩形框表示。关联图中所有的数据流也出现在 0 层数据流图上。此外，0 层数据流图包含了许多数据存储(datastore) ，它是用一对水平的平行线表示， 由于数据存储在系统内部，因此它们并不出现在关联图中。从圆圈到数据存储的流表示数据放入数据存储器，从数据存储器出来的流表示一个读操作，而数据存储器和圆圈之间的双向箭头则表示一个更新操作。

37. 实体-关系图

实体-关系图(Entity—Relationship Diagram，ERD)描绘了系统的数据关系。

利用实体-关系图作为需求分析的工具，表示来自于问题域及其联系的逻辑信息组。

分析实体-关系图有助于对业务或系统数据组成的理解和交互，并暗示产品将有必要包含一个数据库。

实体(entity)是物理数据项(包括人)或者数据项的集合，这对所分析的业务或所要构造的系统是很重要的。

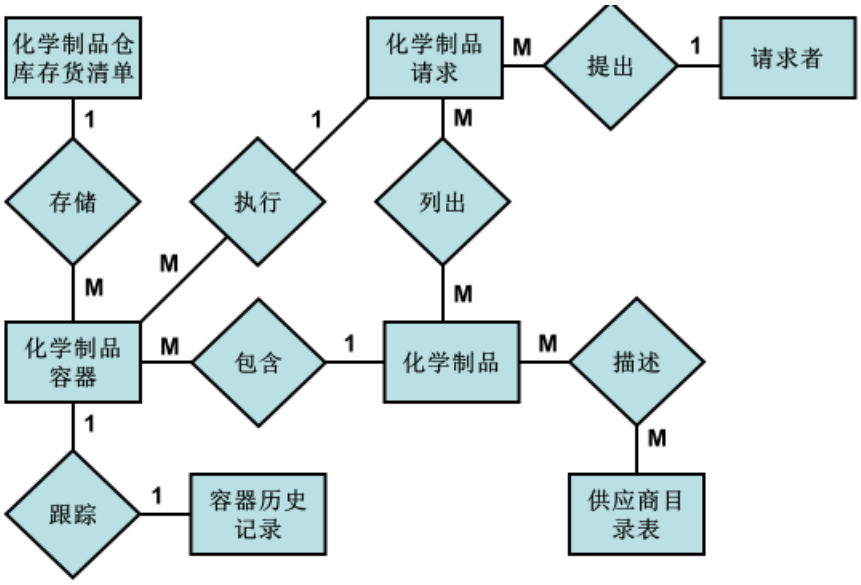


图3.5“化学制品跟踪系统”的实体联系图”

每一实体用几个属性(attribute)来描述；每一实体的单个实例将具有不同的属性值。

实体-关系图中的菱形框代表关系(relationship)，它确定了实体对之间逻辑上和数量上的联系。

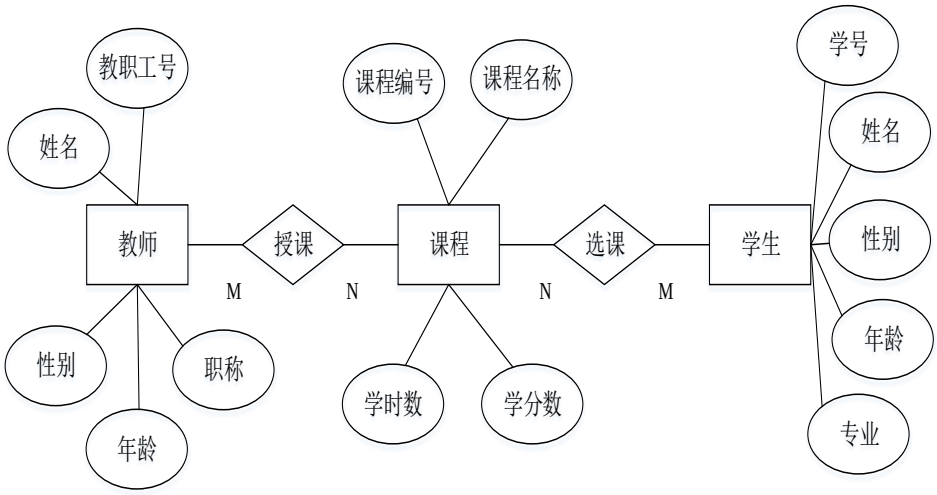


图 2　本科生应用系统中教师、学生和课程之间的实体-关系

38. 状态转换图

- **所有软件系统都包含功能行为、数据操作和状态转变。**

用自然语言描述一个复杂的有限状态机可能会忽略一个允许的状态改变或者引起一个不允许的改变。

与状态机的行为有关的需求可能将多次出现在软件需求规格说明中，它取决于软件需求规格说明是如何组织的。这对综合理解系统行为造成困难。

状态转换图(State Transition Diagram，STD)为有限状态机提供了一个简洁、完整、无二义性的表示。

状态转换图包括如下 3 种元素：

- 用矩形框表示可能的系统状态。
- 用箭头连接一对矩形框表示允许的状态改变或转换。
- 用每个转换箭头上的文本标签表示引起每个转换的事件或条件。
- 标签可能既标识事件也标识相应的系统响应或输出。

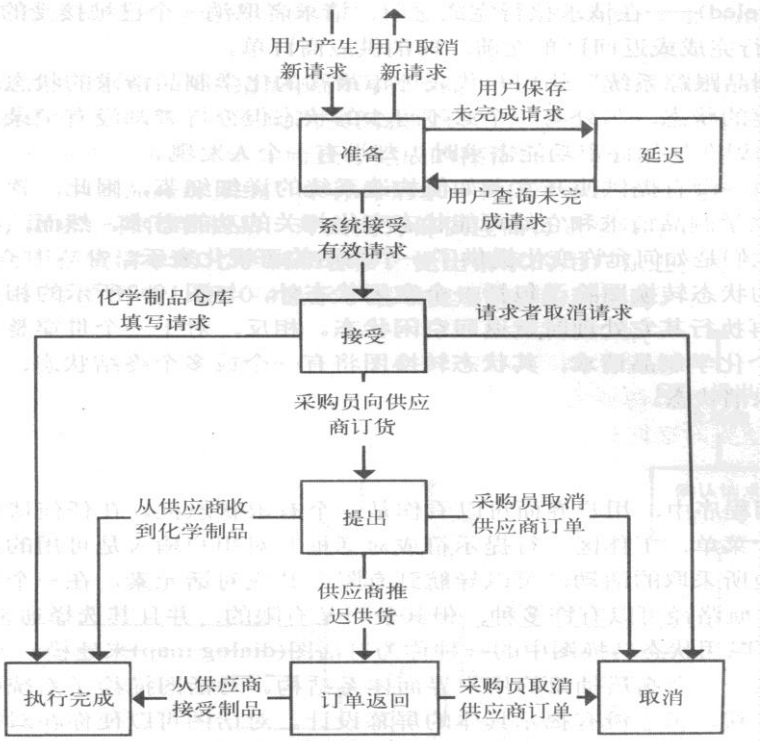


图10-3 “化学制品跟踪系统”中化学制品请求的状态转换图

状态转换图并没有提供使开发者如何构造系统的详细细节。因此，软件需求规格说明必须包括与处理化学制品请求和它的可能状态变化相关的功能需求。

- 状态转换图没有表示出系统所执行的处理细节，只表示了处理结果可能的状态转换。
- 状态转换图有助于开发者理解系统的预期行为，它对于检查所要求的状态和转换是否已全部正确地写入功能需求中也是一种好方法。
- 测试者可以从覆盖所有转换路径的状态转换图中获得测试用例。

39. 对话图

在许多应用程序中，用户界面可以看作是一个有限状态机。许多用户界面可以用状态转换图中的一种称为对话图(dialogmap)来建模。

- 对话图代表了一个高层抽象的用户界面体系结构。

40. 类图

类描述包含了属性(数据)和在属性上执行的操作。

类图(class diagram)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。

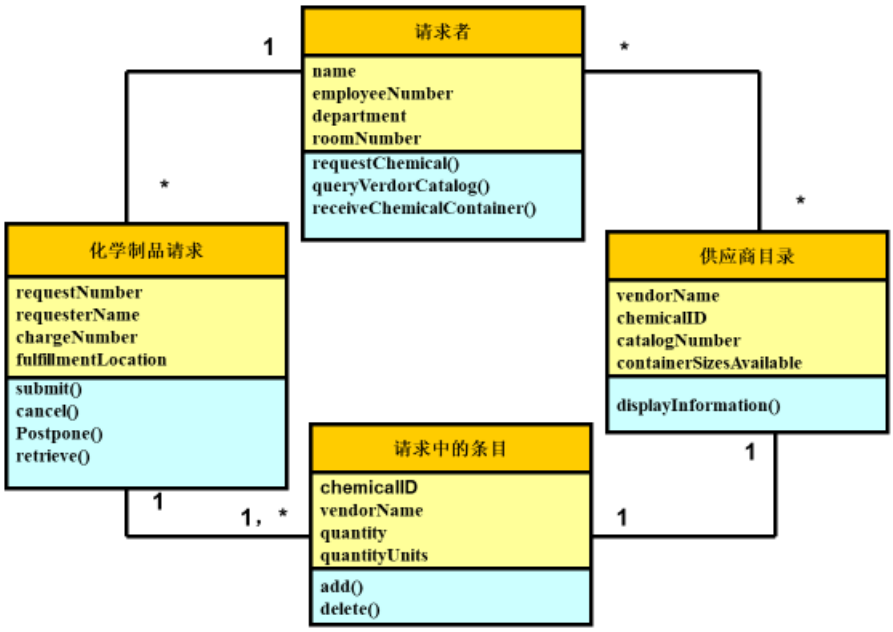


图3.8 化学制品跟踪系统中的部分类图 48

- **不要把用户类和对象类相混淆，虽然它们名字相似，但并不一定存在特定的联系。**

41. 数据字典

数据字典是一个对系统用到的所有数据项和结构定义的共享仓库，它可以确保开发人员使用统一的数据定义，该定义中包括所有数据元素和结构的含义、类型、数据大小、格式、度量单位、精度以及允许取值范围。

第四讲：编写需求文档：需求规格说明

需求开发的最终成果反映在客户和对所开发产品达成共识后所编写的文档中。

- 业务需求要写成项目视图和范围文档；
- 用户需求经常是以用例的形式获得的；
- 产品的详细功能性需求和非功能性需求则记录在软件需求规格说明(Software Requirement Specification,SRS)中。

只有把这些需求组织起来，编写成易于理解的文档，并由项目的主要风险承担者对其进行评审后才能确定。

- 需求开发的最终成果是：客户和开发小组对将要开发的产品达成一致协议。这一协议综合了业务需求、用户需求和软件功能和非功能需求等。
- 项目视图和范围文档包含了业务需求，而用例文档则包含了用户需求。
- 编写从用例派生出的功能需求文档，还要编写产品的非功能需求文档，包括质量属性和外部接口需求。
- 只有以结构化和可读性方式编写这些文档，并由项目的风险承担者评审通过后，各方面人员才能确信他们所赞同的需求是可靠的。

42. 可以用三种方法编写软件需求规格说明：

- 文档：用结构合理的自然语言来精心编写需求文本型文档。
- 建立图形化模型，这些模型可以描绘转换过程、系统状态和它们之间的变化、数据关系、逻辑流或对象类和它们的关系。

  图形化分析模型通过提供另一种需求视图，增强了软件需求规格说明。
- 编写形式化规格说明，这可以通过使用数学上精确的形式化逻辑语言来定义需求。

43. 软件需求规格说明

软件需求规格说明精确地阐述一个软件系统必须提供的功能和性能以及它所要考虑的限制条件或约束。

- 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。

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。

- SRS 作为产品需求的最终成果必须具有综合性,必须包括所有的需求。开发者和客户不能作任何假设。
- 如果任何所期望的功能或非功能需求未写入软件需求规格说明，那么它将不能作为协议的一部分并且不能在产品中出现。
- 必须在开始设计和构造之前编写出整个产品/或每次增量部分的软件需求规格说明。

每次实现的需求文档必须纳入基线（Baseline)。基线(baseline)是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。

- 必须通过项目中所定义的变更控制过程来更改基线软件需求规格说明。

- 所有的参与者必须根据已通过评审的需求来安排工作以避免不必要的返工和误解。
- 测试中要注意使用最新版本的软件需求规格说明。否则，测试工作将是徒劳的。

44. 不同读者使用 SRS 来达到不同的目的：

客户和营销部门依赖 SRS 来了解他们所能提供的产品；

项目经理根据包含在 SRS 中描述的产品来制定规划，并预测进度安排、工作量和所需资源；

软件开发团队依赖 SRS 来理解他们将要开发的产品；

测试小组使用 SRS 中对产品行为的描述制定测试计划、设计测试用例和执行测试过程；

软件维护和支持人员根据 SRS 了解产品的每一部分的功能是什么；

文档编写人员根据 SRS 和用户界面设计文档来编写用户手册和帮助文档；

产品发布组在 SRS 和用户界面设计的基础上编写客户文档，如用户手册和帮助屏幕等；

培训人员根据 SRS 和用户文档编写培训材料；

公司法律顾问要确保该需求遵守相应的法律法规；

分包商根据 SRS 来进行相应的开发工作。

45. 把用户界面的设计编入软件需求规格说明既有好处也有坏处

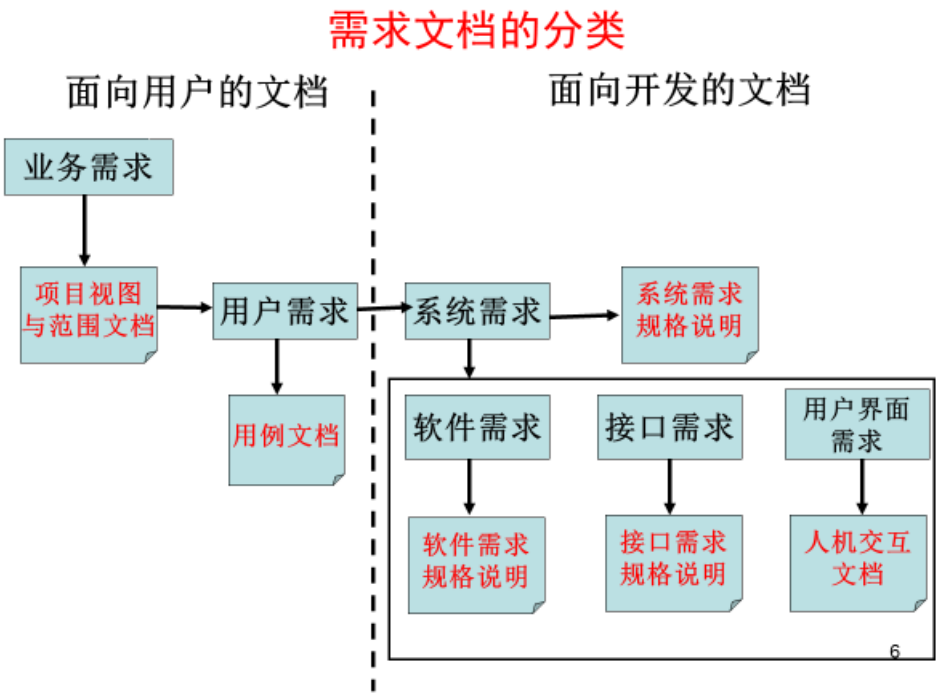
消极方面：

- 屏幕图像和用户界面描述是解决方案(设计)，而不是需求。
- 用户界面的布局不能替代用户需求和功能需求。

积极方面：

- 对可能的用户界面（如，工作原型）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。
- 直观的用户界面更有助于项目计划的制定和预估。

第五讲：需求验证



46. 需求验证的主要方法

1)评审（审查）需求文档

2)以需求为依据编写测试用例

3)编写用户手册

4)确定合格的标准

47. 需求评审

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。

48. 评审员检查的内容有：

系统定义的目标是否与用户的要求一致；

提供的文档资料是否齐全；

文档中的描述是否完整、清晰、准确的反映了用户要求；

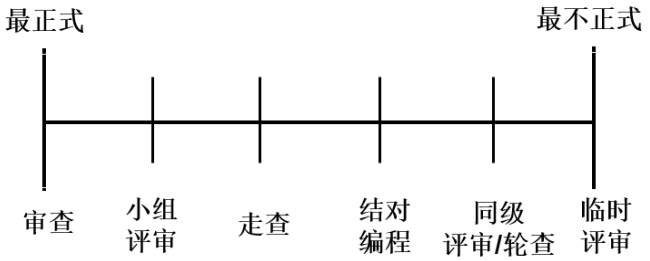
主要功能是否已包括在规定的软件范围之内，是否都已充分说明；

被开发项目的数据流与数据结构是否确定且充分；

设计的约束条件或者限制条件是否符合实际；

是否详细制定了检验标准，能否对系统是否成功进行确认

49. 6 种评审方法



通常，总是由一些非软件开发人员进行产品检查以发现产品所存在的问题，这就是技术评审。

- 需求文档的评审是一项精益求精的技术，它可以发现那些二义性的或不确定的需求、那些由于定义不清而不能作为设计基础的需求，还有那些实际上是设计规格说明的所谓的“需求”。
- 需求评审也为风险承担者们提供了在特定问题上达成共识的方法。

不同种类的技术评审具有不同的称谓。

- 非正式评审的方法包括：把工作产品分发给许多其它的开发人员粗略看一看和走过场似地检查一遍(walk-through)。同时执行者描述产品，且征求意见。
- 正式评审则遵循预先定义好的一系列步骤过程。
- 正式技术评审的最好类型叫作审查(inspection)。

50. 审查过程

（1）参与者

对于一个软件需求规格说明，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，他们的工作基础都是软件需求规格说明。

（2）审查中每个成员扮演的角色

1)作者

作者在审查中却起着被动的作用，不应该充当下列任一角色：调解者、读者或记录员。

2)调解者

3)读者

4)记录员

（3）审查阶段

正如图5-4所示，审查是一个多步骤事件。

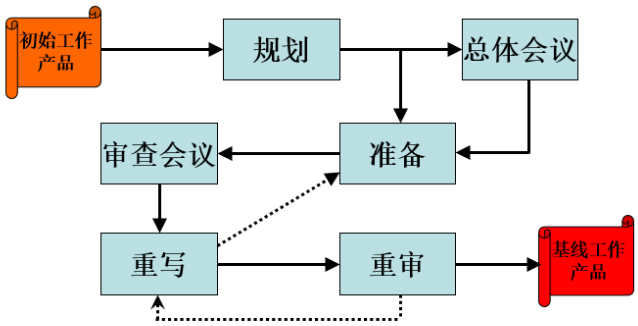


图5-4 审查过程阶段

1)规划(planning)

作者和调解者协同对审查进行规划，以决定谁该参加审查，审查员在召开审查会之前应收到什么材料并且需要召开几次审查会。

审查速度对能发现多少错误影响甚大 。

2)总体会议(overview meeting)

总体会议可以为审查员提供了解会议的信息，包括他们要审查的材料的背景，作者所作的假设和作者的特定审查目标。

3) 准备(preparation)

在正式审查的准备阶段，每个审查员以典型缺陷(defect)清单为指导，检查产品可能出现的错误，并提出问题。审查员所发现的错误中有高达 75%的错误是在准备阶段发现的，所以这一步骤不能省略。

4)审查会议(inspection meeting)

在审查会进行过程中，读者通过软件需求规格说明指导审查小组，一次解释一个需求。当审查员提出可能的错误或其它问题时，记录员就记录这些内容，其形式可以成为需求作者的工作项列表。会议的目的是尽可能多地发现需求规格说明中的重大缺陷。

5)重写(rework)

作者必须在审查会之后，安排一段时间用于重写文档，解决需求中的二义性、消除模糊性，并且为成功开发一个项目打下坚实的基础。

6)重审(follow-up)

这是审查工作的最后一步，调解者或指派人单独重审由作者重写的需求规格说明。重审确保了所有提出的问题都能得到解决，并且正确修改了需求的错误。重审结束了审查的全过程并且可以使调解者做出判断：是否已满足审查的退出标准。

（4）进入和退出审查的标准

①需求文档的进入审查的标准：

- 文档符合标准模板。
- 文档已经做过拼写检查和语法检查。
- 作者已经检查了文档在版面安排上所存在的错误。
- 已经获得了审查员所需要的先前或参考文档，例如系统需求规格说明。
- 在文档中打印了行序号以方便在审查中对特定位置的查阅。
- 所有未解决的问题都被标记为 TBD(待确定)。
- 包括了文档中使用到的术语词汇表。

②需求文档的退出标准：

- 已经明确阐述了审查员提出的所有问题。
- 已经正确修改了文档。
- 修订过的文档已经进行了拼写检查和语法检查。
- 所有 TBD 的问题已经全部解决，或者已经记录下每个待确定问题的解决过程，目标日期和提出问题的人。
- 文档已经登记入项目的配置管理系统。
- 检查是否已将审查过的资料送到有关收集处。

（5）需求评审的困难

1)大型的需求文档

- 为了避免使审查小组感到不安，只在把软件需求规格说明作为基线时才进行审查，在审查全部的文档之前，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。

2)庞大的审查小组

- 把审查组分成若干小组并行地审查软件需求规格说明，并把他们发现的错误集中起来，剔除重复的部分。
- 研究表明：多个审查小组比起单一的大组而言，可以发现需求文档中更多的错误。审查小组总是发现错误的不同子集，所以并行审查的结果是追加的，而不是冗余的。

3)审查员在地域上的分散

- 视频会议是一种有效的解决方案。

第六讲：需求管理

必须使用版本控制技术和配置管理技术来管理需求文档，以达到有效的变更管理。

需求管理包括在工程进展过程中维持需求约定集成性和精确性的所有活动，如图6.1所示。

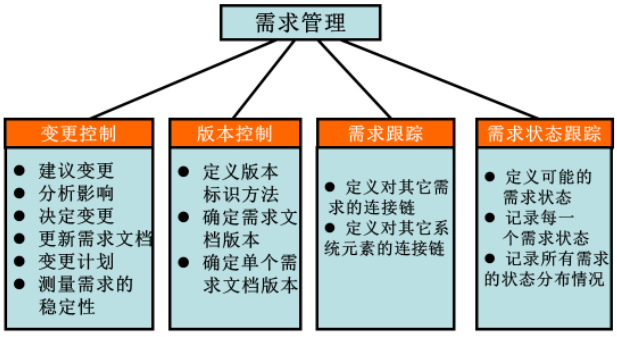


图6.1 需求管理的主要活动

开发团队接受了所建议的需求变更时，有可能无法履行原定的进度安排和质量要求。在这种情况下，必须就约定的变更与所涉及的经理、开发者以及其它相关组织进行协商。

51. 什么是基线？什么是需求基线？

基线(baseline)是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。需求基线（Requirement Baseline)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。

52. 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。通过评估实际的需求管理的工作量，跟踪和了解管理效果，能使我们核实是否确实执行了管理需求所需开展的有关活动，以降低项目的风险。

53. 配置管理

- 配置管理(Configuration Management，CM)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。
- 配置管理的目标是记录产品的演化过程，确保产品开发者在产品生命周期中各个阶段都能得到精确的产品配置。
- 软件配置管理（Software Configuration Management，SCM），又称软件形态管理、或软件建构管理，是一种标识、组织和控制修改的技术。软件配置管理应用于整个软件工程过程。界定软件的组成项目，对每个项目的变更进行管控（版本控制），并维护不同项目之间的版本关联，以使软件在开发过程中任一时间的内容都可以被追溯。

- 软件配置管理可以提炼为三个方面的内容：

1. Version Control-版本控制
2. Change Control-变更控制
3. Process Support-过程支持

54. 需求蔓延

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。

- 管理范围蔓延的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。
- 控制需求蔓延的另一个有效的技术是原型法，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。
- 而控制范围蔓延最有效的方法是要敢于说“不”。

55. 变更控制过程

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。

通过变更控制过程来跟踪已建议变更的状态，确保不会丢失或疏忽已建议的变更。

一旦确定了一个需求集合的基线，应该对所有已建议的变更都遵循变更控制过程。

- 变更控制过程并不是给变更设置障碍。
- 变更过程应该做成文档，尽可能简单，当然首要的是有效性。

只要变更涉及两个人或两个人以上都应该通过变更控制过程来处理。

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

56. **变更控制委员会**(change control board, CCB)

CCB 是由人组成的团体，可以由一个小组担任，也可由多个不同的小组担任，负责做出决定究竟将哪一些已建议的需求变更或新产品特性付诸应用。CCB 同样也决定在哪一些版本中，什么时候纠正哪一些错误。

- 广义上，变更控制委员会对项目中任何基线工作产品的变更都可做出决定，需求变更文档仅是其中之一。

57. CCB 的职责

（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。

58. 影响分析

影响分析是需求管理的一个重要组成部分。影响分析可以提供对建议的变更的准确理解，帮助做出信息量充分的变更批准决策。通过对变更内容的检验，确定对现有的系统做出是修改或抛弃的决定，或者创建新系统以及评估每个任务的工作量。进行影响分析的能力依赖于跟踪能力数据的质量和完整性。

变更只能在项目时间、预算、资源的限制内进行协商。

59. 影响分析的过程，怎么做影响分析

建议的变更涉及的问题核对表（简称表 A），变更影响的软件元素核对表（简称表 B）

过程：

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 6.4 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。

60. 需求跟踪

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。

（1）需求的链接链

跟踪能力(联系)链(traceability link)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。

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。

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。

跟踪联系链记录了单个需求之间的父层、相互连接、依赖的关系。当某个需求变更(被删除或修改)后，这种信息能够确保正确的变更传播，并将相应的任务作出正确的调整。

一个项目不必拥有所有种类的跟踪联系链，要根据具体的情况调整。

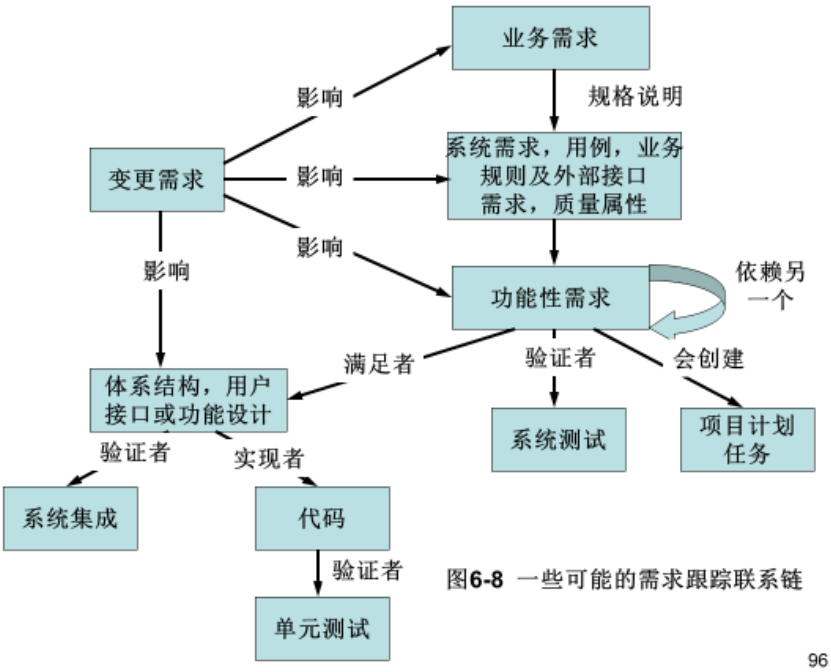


图6-8 一些可能的需求跟踪联系链

96

（2）跟踪联系链可能的信息源

关于不同种类源和目标对象间的联系链：

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

（3）什么是需求跟踪矩阵

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。

61. 使用需求管理工具的益处

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

第七讲：需求风险管理及超越需求开发

答案

一、填空题

1）软件需求工程

2）软件原型

3）设计和实现上　不应该

4）（1）条件或能力　　（3）(1)或(2)

5）需求基线

二、单选题

1）D　　2）C　　3）C　　4）A　　5）B

三、多项选择题

1）ACDE

解析：

可以把需求工程的活动划分为以下 5 个独立的阶段：

（1）需求获取：通过与用户的交流，对现有系统的观察及对任务进行分析，从而开发、捕获和修订用户的需求；

（2）需求建模：为最终用户所看到的系统建立一个概念模型，作为对需求的抽象描述，并儘可能多的捕获现实世界的语义；

（3）形成需求规格：生成需求模型构件的精确的形式化的描述，作为用户和开发者之间的一个协约；

（4）需求验证：以需求规格说明为输入，通过符号执行、模拟或快速原型等途径，分析需求规格的正确性和可行性；

（5）需求管理：支持系统的需求演进，如需求变化和可跟踪性问题。

软件需求工程的活动可以划分为5个独立的阶段：需求获取、需求建模、形成需求规格、需求验证和需求管理。需求建模是_____。

A．分析需求的正确性和可行性的过程
B．对需求的抽象描述
C．对生成需求模型构件的精确的形式化的描述
D．开发、捕获和修订用户的需求

正确答案

B

解析

[解析] 需求获取是通过与用户的交流，对现有系统的观察及对任务进行分析，从而开发、捕获和修订用户的需求过程；需求建模是为最终用户所看到的系统建立一个概念模型，作为对需求的抽象描述，并尽可能多地捕获现实世界的语义；形成需求规格是生成需求模型构件的精确的形式化的描述，作为用户和开发者之间的一个协约；需求验证是以需求规格说明为输入，通过符号执行、模拟或快速原型等途径，分析需求规格的正确性和可行性；需求管理是支持系统的需求演进，如需求变化和可跟踪性问题。

2）ACE 不确定（题目问的是“整个”）

解析：

[解析] 软件需求规格说明书是确保软件质量的有力措施，衡量软件需求规格说明书质量好坏的标准、标准的优先级及标准的内涵是：
①正确性。体现待开发系统的真实要求。
②无歧义性。对每一个需求只有一种解释，其陈述具有唯一性。
③完整性。包括全部有意义的需求，功能的、设计的、性能的、约束的属性或外部接口等方面的需求。
④可验证性。描述的每一个需求都是可以验证的，即存在有限代价的有效过程验证确认。
⑤一致性。各个需求的描述不矛盾。
⑥可理解性。需求说明书必须简明易懂，尽量少包含计算机的概念和术语，以便用户和软件人员都能接受它。
⑦可修改性。SRS的结构风格在需求有必要改变时是易于实现的。
③可追踪性。
其中最重要的、放在第一位的就是正确性。

3）ABCE

4）BCE

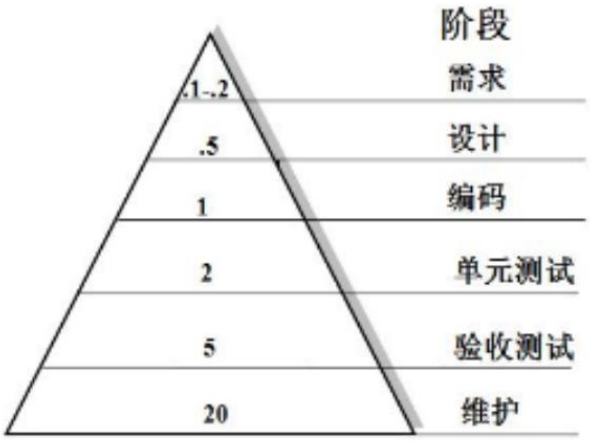
5）AC

6）ABC

四、为什么在软件开发项目中维护阶段发现错误的修复成本是需求阶段发现错误修复成本的 100 倍到 200 倍（3-5）？详细说明这些成本的主要构成（10-12）？

答：

1. 因为维护是建立在需求、设计、编码等的基础之上的，如果在维护阶段发现错误，那么要修复，或许就要从编码、设计、需求等阶段开始修复，随之伴随而来的，可能就是要重新进行规格说明，重新进行设计，重新进行编码等，这就成倍的增加了修复的成本。如下图所示，该图是许多公司项目生命周期各阶段修复成本的度量和计算，由图可得，如果把编码阶段发现和修复一个错误所需要的努力用“1”个成本单元表示的话，那么，需求阶段的错误修复成本是它的 5—10，而在维护阶段发现和修复一个错误的成本超过 20 倍，因此，软件开发项目中维护阶段发现错误的修复成本是需求阶段发现错误修复成本的 100 倍到 200 倍。



2. 这些成本由以下方面构成：

（1）重新进行规格说明:

（2）重新设计；

（3）重新编码；

（4）重新测试；

（5）版本升级：用一个修正后的版本来替代有缺陷的版本；

（6）纠正活动：消除由于不正确的系统错误造成的一切危害，这可能涉及到偿还不满用户的经济损失，以及重新运行系统所付出的代价等；

（7）报废：包括以最好的意图完成的代码、设计和测试用例，当发现它们是依据于不正确的需求时则不得不全部丢弃！

（8）收回有缺陷的软件版本以及相关的用户手册。 有时软件可能会已经嵌入到数字手表、微波炉或汽车等产品中，这时所收回的内容也包括有形的产品和嵌入该系统的软件；

（9）保修成本；

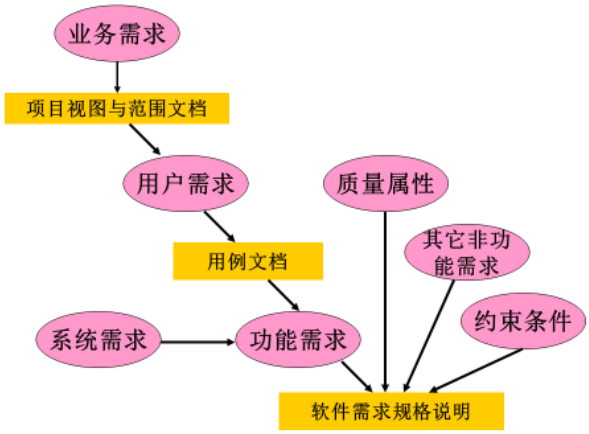
（10）产品赔偿：客户可能要求对有缺陷软件造成的损害进行赔偿；

（11）公司代表到客户那里重新安装软件所必须支付的服务成本；

（12）建档成本。

五、图示并论述软件需求的组成层次及其相互关系。

软件需求各组成部分之间的关系如图所示。



软件需求包括三个不同的层次:

（1） 业务需求 (business requirement)：反映了组织机构或客户对系统、产品高层次的目标要求，它们在项目视图与范围文档中予以说明。

（2） 用户需求(user requirement)：文档描述了用户使用产品必须要完成的任务，这在使用实例(use case，简称用例)文档或方案脚本(scenario)说明中予以说明。

（3） 功能需求(functional requirement)：定义了开发人员必须实现的软件功能，使得用户能完成他们的任务，从而满足了业务需求。

此外，还包括系统需求和其他需求，其他需求分为质量属性或其他非功能需求和设计约束等。

六、简述软件需求的几种典型来源

与潜在用户进行交谈和讨论、

把对现有产品或竞争产品的描述写成文档、

系统需求规格说明、

现有系统的问题报告和改进要求、

市场调查和用户问卷调查、

观察用户如何工作、

用户工作的情景和内容分析、

事件和响应

七、分别说明每项需求和整个需求规格说明书应具有哪些主要特征？图示并论述需求审查的过程，并说明需求规格说明书进入和退出审查的标准。

答：

1. 主要特性: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。

2. 需求评审要经历如下过程：

（1）规划。作者和调解者协同对审查进行规划，以决定谁该参加审查，审查员在召开审查会之前应收到什么材料并且需要召开几次审查会。

（2）总体会议。总体会议可以为审查员提供了解会议的信息，包括他们要审查的材料的背景，作者所作的假设和作者的特定审查目标。如果所有的审查员对要审查的项目都很熟悉，那么就可以省略本次总体会议。

（3）准备。在正式审查的准备阶段，每个审查员以典型缺陷清单为指导，检查产品可能出现的错误，并提出问题。

（4）审查会议。在审查会进行过程中，读者通过软件需求规格说明指导审查小组，一次解释一个需求。当审查员提出可能的错误或其它问题时，记录员就记录这些内容，其形式可以成为需求作者的工作项列表。会议的目的是尽可能多地发现需求规格说明中的重大缺陷。另外，审查会不应该超过两个小时，如果需要更多的时间，就另外再安排一次会议。

（5）重写。几乎每一个质量控制活动都可能发现一些需求缺陷。因此，作者必须在审查会之后，安排一段时间用于重写文档，解决需求中的二义性、消除模糊性，并且为成功开发一个项目打下坚实的基础。

（6）重审。这是审查工作的最后一步，调解者或指派人单独重审由作者重写的需求规格说明。重审确保了所有提出的问题都能得到解决，并且正确修改了需求的错误。重审结束了审查的全过程并且可以使调解者做出判断：是否已满足审查的退出标准。具体流程如下图：

正如图5-4所示，审查是一个多步骤事件。

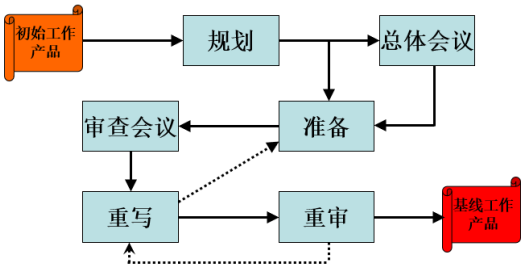


图5-4 审查过程阶段

3. 需求规格说明书进入审查的标准：

（1） 文档符合标准模板；

（2） 文档已经做过拼写检查和语法检查；

（3） 作者已经检查了文档在版面安排上所存在的错误；

（4） 已经获得了审查员所需要的先前或参考文档，例如系统需求规格说明；

（5） 在文档中打印了行序号以方便在审查中对特定位置的查阅；

（6） 所有未解决的问题都被标记为 TBD(待确定)；

（7） 包括了文档中使用到的术语词汇表。

4. 需求规格说明书退出审查的标准：

（1） 已经明确阐述了审查员提出的所有问题；

（2） 已经正确修改了文档；

（3） 修订过的文档已经进行了拼写检查和语法检查；

（4） 所有 TBD 的问题已经全部解决，或者已经记录下每个待确定问题的解决过程，目标日期和提出问题的人；

（5） 文档已经登记入项目的配置管理系统；

（6） 检查是否已将审查过的资料送到有关收集处。

1. 每项需求和整个需求规格说明书应具有哪些主要特征？

需求验证确保需求符合需求陈述(requirement statement)的良好特征(完整的、正确的、可行的、必要的、具有优先级的、无二义性及可验证的)，并且符合需求规格说明的良好特性(完整的、一致的、易修改的、可跟踪的)。

2. 图示并论述需求审查的过程

正如图**5-4**所示，审查是一个多步骤事件。

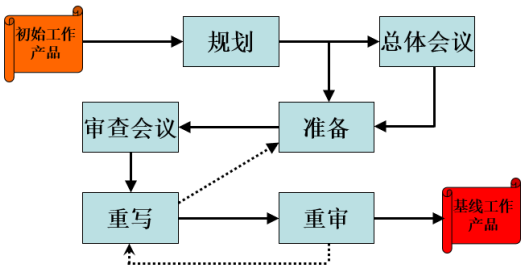


图**5-4** 审查过程阶段

1)规划(planning)

作者和调解者协同对审查进行规划，以决定谁该参加审查，审查员在召开审查会之前应收到什么材料并且需要召开几次审查会。

2)总体会议(overview meeting)

总体会议可以为审查员提供了解会议的信息。

3) 准备(preparation)

在正式审查的准备阶段，每个审查员以典型缺陷(defect)清单为指导，检查产品可能出现的错误，并提出问题。

4)审查会议(inspection meeting)

会议的目的是尽可能多地发现需求规格说明中的重大缺陷。

5)重写(rework)

作者必须在审查会之后，安排一段时间用于重写文档，解决需求中的二义性、消除模糊性，并且为成功开发一个项目打下坚实的基础。

6)重审(follow-up)

调解者或指派人单独重审由作者重写的需求规格说明。重审确保了所有提出的问题都能得到解决，并且正确修改了需求的错误。

3. 需求规格说明书进入和退出审查的标准

(1) 需求文档的进入审查的标准

- 文档符合标准模板。
- 文档已经做过拼写检查和语法检查。
- 作者已经检查了文档在版面安排上所存在的错误。

- 已经获得了审查员所需要的先前或参考文档，例如系统需求规格说明。
- 在文档中打印了行序号以方便在审查中对特定位置的查阅。
- 所有未解决的问题都被标记为 TBD(待确定)。
- 包括了文档中使用到的术语词汇表。

(2) 需求文档的退出标准

- 已经明确阐述了审查员提出的所有问题。
- 已经正确修改了文档。
- 修订过的文档已经进行了拼写检查和语法检查。
- 所有 TBD 的问题已经全部解决，或者已经记录下每个待确定问题的解决过程，目标日期和提出问题的人。
- 文档已经登记入项目的配置管理系统。
- 检查是否已将审查过的资料送到有关收集处。

八、论述变更管理中的主要活动

1. 建立基线、变更控制系统和变更控制流程
2. 识别变更，并以书面格式提出变更请求
3. CCB 审查变更请求
4. CCB 批准或者否决变更
5. 将批准的变更纳入项目管理计划中
6. 实施被批准的变更
7. 监控被批准变更的实施情况
8. 变更验证
9. 沟通存档