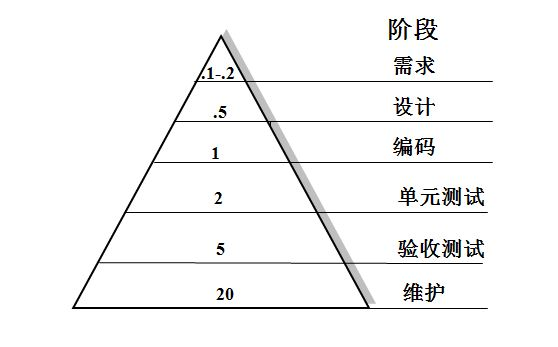
**1.（15分）为什么在软件开发项目中维护阶段发现错误的修复成本是需求阶段发现错误修复成本的100倍到200倍（3-5）？详细说明这些成本的主要构成（10-12）？**

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。

1. **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**。

***需求工程5阶段：***

**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；

**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，并尽可能多的捕获现实世界的语义；

**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；

**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；

**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。

1. 需求开发中的**问题获取、需求分析、编写规格说明和需求验证**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。
2. 需求工程中的这些活动一般是多次循环的形式，循环的过程如图所示.

**主要活动:**

**1）需求获取：**

编写项目视图和范围（vision and scope)文档

确定用例（Use Cases)

确定质量属性

1. **需求分析:**

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）

**3）编写需求规格说明书**

定制文档模板

编写文档

1. **需求验证**

执行验证：同级评审

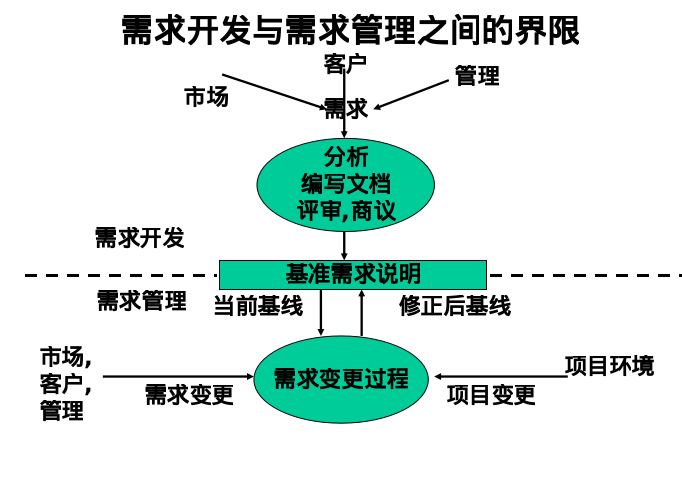
问题修正：发现问题及时修正

**5）需求管理**

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。

****

**不适当的需求过程所引起的一些风险:**

* 用户参与不多导致产品无法被接受。
* 用户需求的增加带来过度的耗费和降低产品的质量。
* 模棱两可的需求说明可能导致时间的浪费和返工。
* 用户增加一些不必要的特性和开发人员画蛇添足。
* 过分简略的需求说明以致遗漏某些关键需求。
* 忽略某类用户的需求将导致众多客户的不满。
* 不完善的需求说明使得项目计划和跟踪无法准确进行。

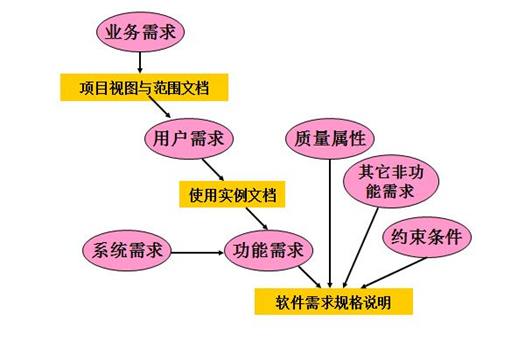
**2.（12分）什么是软件需求（5）？说明软件需求的层次并描述其相互关系（7）。**

答：1、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权能(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权能。（3）一种反映上面(1)或(2)所描述的条件或权能的文档说明。**

**或答：**

**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。

1. 软件需求的三个不同层次之间的关系可用下图表示**（图正确得4分）**

 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。**
4. **非功能需求：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**
5. **设计约束:指对开发人员在软件产品设计和构造上的限制，包括：非技术因素的技术选型, 预期的软硬件环境**

**3.（15分）选定一不少于四种用户类的简单项目，论述该项目的视图陈述（4），确定并分析项目的用户类及特征（4），给出系统用例模型（4），并绘制系统关联图（3）。**

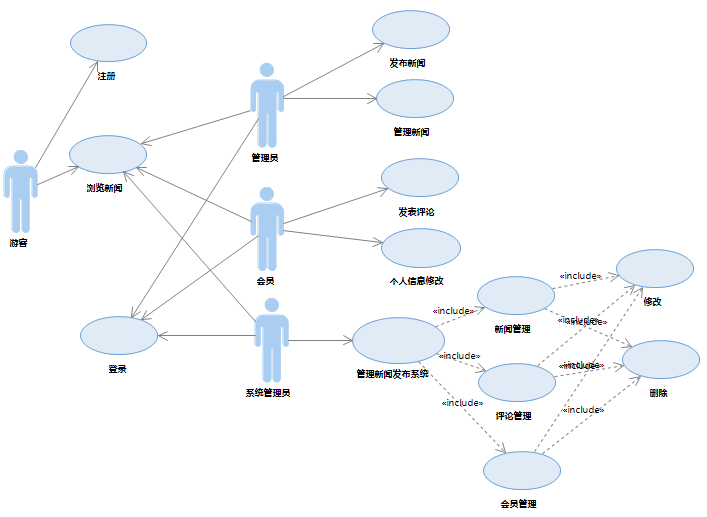
答：新闻发布系统

1、项目陈述如下：

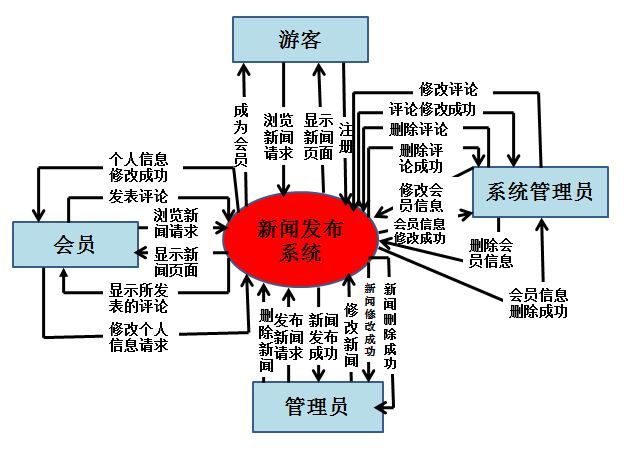
“新闻发布系统”可使任何人方便的对新闻内容进行浏览，任何人可以通过注册成为会员，会员可以享有对新闻和评论进行评论的权限，同时会员也可以对自己的个人信息进行修改，管理员登录系统后，可以在后台发布并管理新闻，后台的系统管理员可以管理新闻、评论和会员信息。系统可以对新闻进行有效的管理，包括新闻的各种内容、属性还有评论和会员信息等，通过不同用户所拥有的管理权限，方便对新闻等信息进行删除更改，同时用户通过登录功能可以帮助用户随时了解新闻状态，保持新闻的时效性和正确性，同时扩大新闻的阅读量和传播率，避免新闻发布可能产生的管理混乱，严格用户职责，做到责任追溯，评论追溯等科学化管理。

2、用户类及特征分析（略）

3、用例模型（参考）：



4、系统关联图：



**4（12）什么是软件原型（3）？使用原型的目的有哪些（3）？说明软件原型的种类和使用原型技术应遵守的主要原则（6）**。

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。一个**软件原型是所提出的新产品的部分实现**，通过使用原型可以使开发小组正确理解需求，发现和解决在产品开发的早期阶段不确定的问题以及需求中的二义性和不完整性问题，最终明确如何最好地实现这些需求并最终明确并完善需求、探索设计选择方案、发展为最终的产品。同时用户、经理和其他非技术项目风险承担者发现在确定和开发产品时，原型可以使他们的想象更具体化。

使用原型有三个主要目的：

**明确并完善需求:**  原型作为一种需求工具，它是对部分系统的初步实现。用户对原型的评价可以指出需求中存在的问题，在开发真正产品之前，可以最低的费用来解决这些问题。

**探索设计选择方案:** 原型作为一种设计工具，用它可以探索不同的用户界面技术，使系统达到最佳的易用性，并且可以评价可能的技术方案。

**发展为最终的产品原型:** 作为一种构造工具，是产品最初子集的完整功能实现，通过一系列小规模的开发循环，你可以完成整个产品的开发。

**软件原型的种类：水平原型和垂直原型、抛弃型原型和进化型原型、电子原型和书面原型。**

通过水平和垂直原型让用户体验或者验证需求实现的具体行为（或操作）以及部分确定性的功能，而抛弃型和进化型原型则针对不确定性的问题通过原型进行探讨和研究最终剔除掉需求的不确定性。

为了帮助开发者在需求开发过程中建立有效的原型，**请遵循如下原则：**

* **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。**用最少的投资开发那些用于回答问题和解决需求的不确定性的原型。不要努力去完善一个抛弃型原型的用户界面。
* **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**。当一个简单的抛弃型原型达到原型目的时，就不应该随便扩充它的功能。
* **不要从水平原型的性能推测最终产品的性能**。原型可能没有运行在最终产品所处的特定环境中，并且你开发原型的工具与开发产品的工具在效率上是存在差异的。
* **在原型屏幕显示和报表中使用合理的模拟数据。**那些评价原型的用户会受不现实数据的影响而不能把原型看成真正产品的模型。
* **不要期望原型可以代替需求文档。**原型只是暗示了许多后台功能，因此必须把这些功能写入软件需求规格说明，使之完善、详细并且可以有案可稽。

**5.（12）简述软件需求的几种典型来源。**

典型的软件需求来源:

* **与潜在用户进行交谈和讨论**
* **描述现有产品或竞争产品的文档**
* **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**
* **用户工作的情景分析**
* **事件和响应**

**并做适当的解释。**

**6 （12分）本课程中涉及的主要图形化分析方法有哪些（5）？绘制系统数据流图应遵循哪些原则（7）?**

答：1、本课程中涉及的主要的图形化分析方法有：

**用例图，数据流图，实体联系图，状态转换图，对话图，类图。**

1. 绘制系统数据流程图应遵循的原则有：
2. **把数据存储放在0层数据流图或更低层子图上，不要放在关联图上**；
3. **过程是通过数据存储进行通讯，而不是从一个过程直接流到另一过程。**类似地，数据不能直接由一个数据存储直接流到另一个数据存储，它必须通过一个过程圆圈；
4. **使用数据流图时，不要试图让数据流图反映处理的顺序；**
5. **用一个简明的动作命名过程：动词+对象。**数据流图中所用的名字应对客户有意义，并且与业务或问题域相关；
6. **对过程的编号要唯一且具有层次性**。在0层图上，每个过程的编号用整数表示。如果你为过程3创建子图，则子图中的过程编号应表示为3.1，3.2等等；（6）**不要在一个图中绘制多达7-10个以上的过程**，否则就很难绘制、更改和理解；
7. **不要使某些圆圈只有输入或只有输出**。数据流图中圆圈所代表的处理过程通常要求既有输入又有输出。

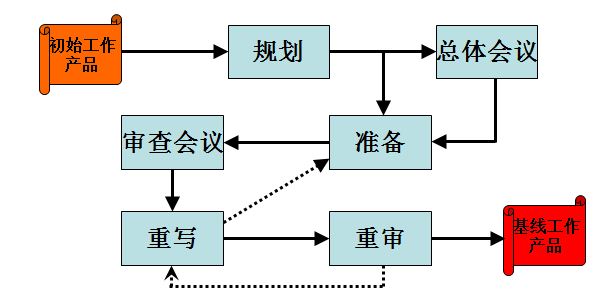
**7.（12分）优秀需求及需求规格说明应具有哪些主要特性（5）？图示并论述需求审查的过程(4)，并说明需求规格说明书进入和退出审查的标准（3）。**

**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。**

**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：



**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。

**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。

**8.（10）需求管理的主要活动有哪些（6），给出需求变更控制过程描述（4）。**

答: 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下:

**变更控制**

1. **需求跟踪**

* 定义对其它需求的连接链
* 定义对其它系统元素的连接链

1. **需求状态跟踪**

* 定义可能的需求状态
* 记录每一个需求状态
* 记录所有需求的状态分布情况
* 建议变更
* 分析影响
* 决定变更
* 更新需求文档
* 变更计划
* 测量需求的稳定性

**版本控制**

* 定义版本标识方法
* 确定需求文档版本
* 确定单个需求文档版本

**需求变更控制过程描述如下（加必要解释）：**

1．概述

1.1 目的

1.2 范围

1.3 定义

2．角色和职责

3．变更请求状态

4．开始条件

5．任务

5.1 评估请求

5.2 做出决策

5.3 执行变更

5.4 通知受变更影响的各方

6．验证