系统设计与分析考试复习范围-自整理

**历年试卷**

//2005A 2006A

1、UML中Association(关联), Aggregation(聚合), Composition(组合)和Dependency(依赖)辨析

2、UML三种拓展机制

Stereo type(构造型)、Tagged Value（标记值）、Constraint（约束）

3、约束的概念

约束是用文字表达式来表示元素、依赖关系、注释上的语义限制。约束用大括号内的字符串表达式表示{XXXX}

4、UML接口

用于声明对象类所需要的服务

5、活动图

定义：用来描述满足用例要求所要进行的活动以及活动间的约束关系（描述用例的执行）

优缺点：（1）活动图擅长列举对象协作以完成任务（2）很难准确地识别如何将活动图操作映射到对象模型的具体细节。（3）由于活动图允许并行行为建模，所以它们有利于工作流建模。

6、场景中的消息

序列图---消息/方法名字放置在带箭头的线上面。正在被传递给接收对象的消息，表示接收对象的类 实现的一个操作/方法（被调用对象的操作）

7、动态模型描述系统随时间的变化

状态机视图、交互视图、活动视图

8、Rose的四种视图模型

用例视图、逻辑视图、组件视图、部署视图

9、可以表示并发处理的UML图

活动图、交互图

10、UML体系结构是一种四层元模型体系结构.

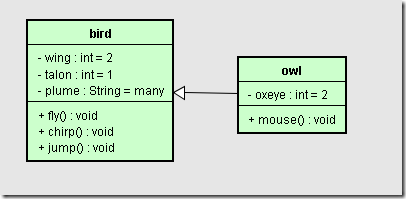
元元模型层、元模型层、模型层、用户模型层

2、//解释概念并举例

（1）协作图

是一种交互图，协作图显示了一系列的对象和在这些对象之间的联系以及对象间发送和接收的消息。

（2）UML类图关系（类与类之间的关系-泛化、依赖、关联、聚合、组合）

泛化（Generalization）-继承关系，指的是一个类（称为子类、子接口）继承另外的一个类（称为父类、父接口）的功能，并可以增加它自己的新功能的能力。//猫头鹰是鸟的一种，即有鸟的共性也有猫头鹰的特性

（3）Transition //状态的转移？移交？

（4）封装（Encapsulation）

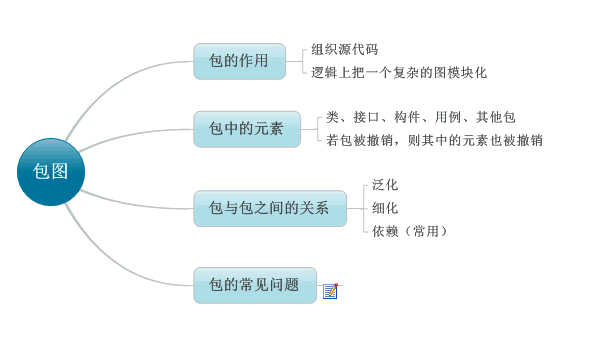
3、按场景绘制类图

4、面向对象建模—面向对象分析及UML建模

//2005B 2006B

1、在实现过程中，开发团队决定使用第三方实用程序来提供一些所需的功能。什么OOAD工件最可能反映这一变化：包图、相互作用图

包图是一种维护和描述系统总体结构的模型的重要建模工具，通过对包中各个包以及包之间关系的描述，展现出系统的模块与模块之间的依赖关系。

包图的作用：包图可以描述需求，设计的高阶概况；包图通过合理规划自身功能反应系统的高层架构，在逻辑上将系统进行模块化分解；包图最终是组织源码的方式

组件图是个UML框图，用来显示系统中的组件及其相互依赖性。组件图中，可以了解各软件组件（如源代码文件或动态链接库）之间的编译器和运行时依赖关系。

2、创建与场景相关的用例

3、确定用于调用操作的公共方法名称的最佳来源的OOAD工件：交互图、类图

4、部署图

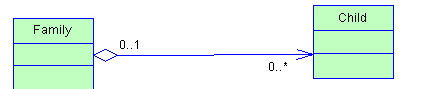
用来显示系统中软件和硬件的物理架构。部署组件之间的依赖关系往往与包依赖关系相同；部署图可以用来显示代码的物理模块；部署图可用于显示系统在计算机之间的物理分布

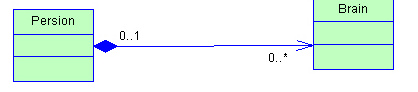
5、活动图中最大并发线程

并发路径-对活动建模时，往往要将一个转移划分成两个单独的同时（并发）执行的路径，而后他们再合并到一起。可用一个与转换线垂直的黑色粗实线条表示，并发的路径从这个实线条引出。

2、（1）哪个UML图用来描述单个类的行为

(2) 组合和聚合之间的区别

聚合是一种强的关联关系。是整体和个体之间的关系。例如，汽车类与引擎类，轮胎类之间的关系就是**整体与个体**之间的关系

组合也是关联关系的一种，一种比聚合关系强的关系。组合关系中的部分类不能独立于整体类存在。

4、（1）面向对象分析OOA与传统结构化分析的不同之处

5、根据场景-设计-UML绘图

//2007A

1. 面向对象技术优点

增加封装；模块之间的耦合减少；高级抽象

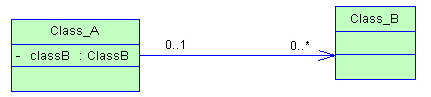
1. 用例图相关概念

用例图显示用例与参与者之间的交互；用例图显示哪些参与者启动用例；用例图说明了参与者从用例中接收信息。

2、//解释概念并举例子

（2）用例

（3）关联关系

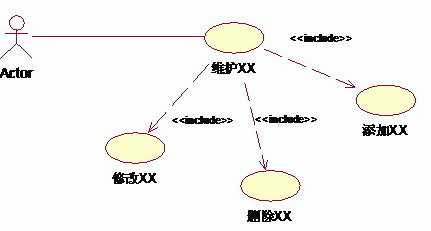
体现的是两个类、或者类与接口之间语义级别的一种强依赖关系。被关联类B以类属性的形式出现在关联类A中，也可能是关联类A引用了一个类型为被关联类B的全局变量

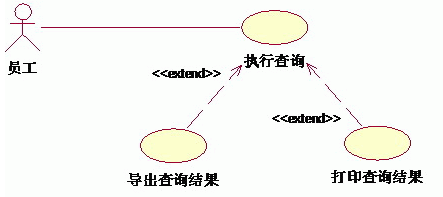
//2007B

1、用例图、用例、用例关系等相关概念

·包含关系：使用包含（Inclusion）用例来封装一组跨越多个用例的相似动作

·扩展关系：将基用例中一段相对独立并且可选的动作，用扩展（Extension）用例加以封装，再让它从基用例中声明的扩展点（Extension Point）上进行扩展。（扩展用例为基用例添加新的行为。扩展用例可以访问基用例的属性，因此它能根据基用例中扩展点的当前状态来判断是否执行自己。但是扩展用例对基用例不可见）

·泛化关系

//2010A

1、哪个UML图用来描述系统操作人员在用例上下文中识别和处理的外部系统事件？活动图

4、为重用而设计

低耦合--降低依赖性，减少变化带来的影响，提高重用性----解决方案：分配职责

5、子系统概念

一个子系统是一个有单独规范和实现部分的包。它代表了模型中一个连续的，和系统中其它部分有清晰的接口的单元。它通常代表了系统中以功能或实现分界的一部分（子系统用来展示系统的一部分功能，这部分功能联系紧密）

6、在OO系统中，分配职责

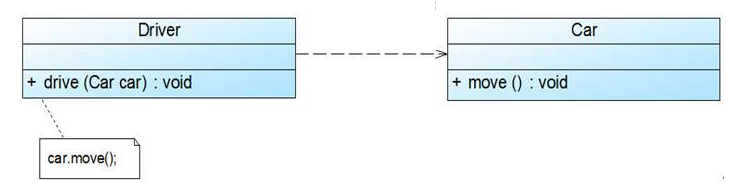
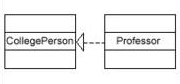
如何分配责任（对象责任）--在进行用例实化时，设计者应该定义具有功能上内聚的责任集的对象和类

GRASP:通用职责分配软件模式

10、当使用OOAD构件来组织和分配项目上的团队责任时，最好是：根据软件体系结构中的层划分团队，并让他们尽可能独立地工作，以最大限度地减少层之间的依赖。

13、类与类之间的关系-依赖

在UML中，依赖关系用带箭头的虚线表示，由依赖的一方指向被依赖的一方。例如：驾驶员开车，在Driver类的drive()方法中将Car类型的对象car作为一个参数传递，以便在drive()方法中能够调用car的move()方法，且驾驶员的drive()方法依赖车的move()方法，因此类Driver依赖类Car

1. 类与类的关系—实现关系

2、时序图含义

描述对象之间传递消息的时间顺序, 用来表示用例中的行为顺序, 是强调消息时间顺序的交互图。时序图中每条消息都代表了类的一个操作 或者 引起状态机改变的触发事件

时序图包括四个元素 对象(Object), 生命线(Lifeline), 激活(Activation), 消息(Message)

**对象**：表示系统的参与者或者任何有效的系统对象

---处于顶部 : 如果对象的位置在时序图顶部, 说明在交互开始的时候对象就已经存在了; ---不在顶部 : 如果对象的位置不在顶部, 那么对象在交互过程中创建的

**生命线**：生命线是一条垂直的虚线, 从时序图顶部一直到底部都存在, 其长度取决于交互的时间

**激活**: 代表时序图中对象执行一项操作的时期，该对象被占用以完成某个任务

**消息**：

---异步通信 : 指消息发送者通过消息把信号传递给消息的接收者，然后自己继续活动，不等待接收者返回消息或者控制。异步消息的接收者和发送者是并发工作的

---同步通信 : 指消息的发送者把控制传递给消息的接收者，然后停止活动，等待消息的接收者放弃或者返回控制

3、//问答

（1）描述软件开发的风险和避免这些风险的方法：用户或业务需要没有得到满足 需求得不到解决 模块未集成 维修困难 漏洞的发现晚 最终用户体验质量差 负载下的性能较差 没有团队协调 建立和释放问题

（2）RUP的阶段和工作流程。

Inception（初始阶段）：定义项目范围

Elaboration（细化阶段）：计划项目，指定功能和基线架构

Construction（构造阶段）：建立产品

Transition （交付阶段）：过渡产品到最终用户社区

（3）RUP中采用的软件体系结构的“4+1视图”

用例视图，逻辑视图，实现视图，过程视图，部署视图

（4）描述OOA/D中的用例分析步骤

-补充用例说明。

-找出用例中的行为，把行为合理分配给各个类。

-对每一个分析出来的类，描述其职责，属性，和类间的关联，限定分析机制。

-统一分析类。

-查检分析过程和结果

5、//**建模 有题目及答案**

**概念性内容** //复习ppt

1、基于UML的开发过程的三个特征：用例驱动 、以架构为中心、迭代开发

2、6个最佳实践：迭代开发、管理需求、使用基于构件的架构、可视化建模、持续验证质量、管理变更

3、什么是软件开发过程－who、what、when、how

4、RUP的四个阶段：初始、细化、构造、移交

5、RUP的四个主要里程碑：LCO、LCA、IOC、Release

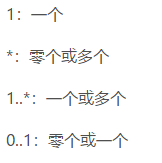
6、软件架构的“4＋1”视图：用例视图：逻辑视图、实现视图、进程视图、部署视图

7、分析类的三种构造型：边界类、实体类、控制类

8、识别运行时机制的步骤; 分析并发需求、识别进程和线程、识别进程生命周期、将设计元素映射到进程

9、请描述UML分析模型和UML设计模型之间的典型差异。

10、解释多重性

在UML中类之间的关系有一种称为关联，其中多重性用来描述类之间的对应关系，

**建模实例**

1、场景：在校园，学生可以申请消费卡。卡管理员将在收到申请时创建新的消费卡。 消费卡可用于校园超市。 当学生使用卡消费时，销售人员将从卡中扣除钱。 此外，卡管理员可以将钱存入卡中以供学生使用。 想象一下，你被雇用来开发支持软件。

1）为系统绘制UML用例图。

2）生成序列图以模拟使用卡消费的场景。

3）绘制系统的类图。

4）为消费卡类生成状态图。

** **

（deposit 储蓄/存款）

****

（report a loss报失 idle空闲）

2、场景：校园里有一系列教室。每个教室都有一系列设备，可能包括麦克风，黑板，高架投影仪，电视，互联网连接等。每台设备都有历史，条件以及设备特有的信息，如黑板和电视，移动或不移动麦克风，以及连接到互联网的速度。每个教室都有一份计划在其中进行的课程列表，每门课程的讲师，每门课程要求的设备和设置，以及负责设置设备的经理。系统应允许 教师查找教室中的设备列表，请求设备和设置，并查询设置状态。系统应该在教室中向教室经理发送设备和设置请求。经理可以更新设备信息和设置状态。如果无法根据请求设置设备，经理应尽快通知教师。系统还应允许 任何人向经理发送评论。

1. 绘制该系统的用例图。
2. **** 绘制该系统的类图。

****

//软件工程ppt

1、面向对象:以类为基本程序单元，对象是类的实例化,对象之间以消息传递为基本手段

2、现代程序设计技术-面向对象分析与设计技术（OOA/OOD）

3、对象、属性、方法等

接口（Interface）是“方法声明(Method Signature)”的集合

**UML建模（概念、组成）**

1. **用例图**

**2、时序图**

**3、类图**