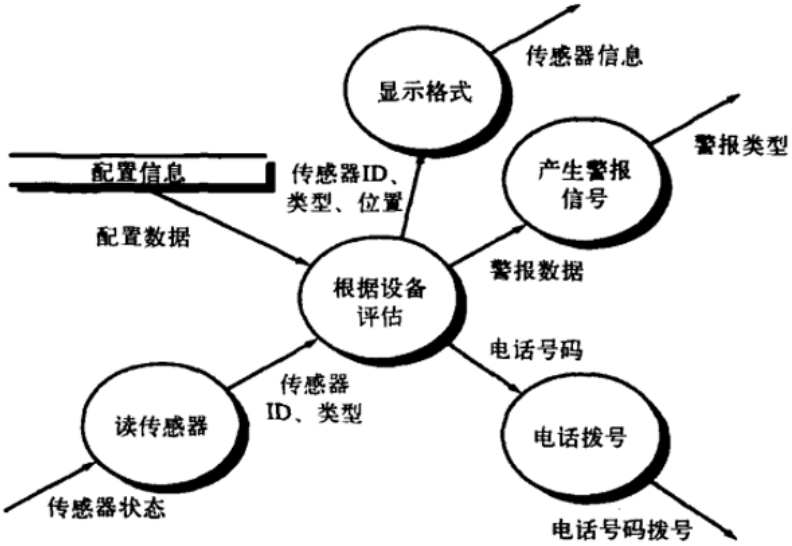
数据流图 + CRC + 状态图 + 软件体系结构 + test + use-case + NSU + event trace + class图

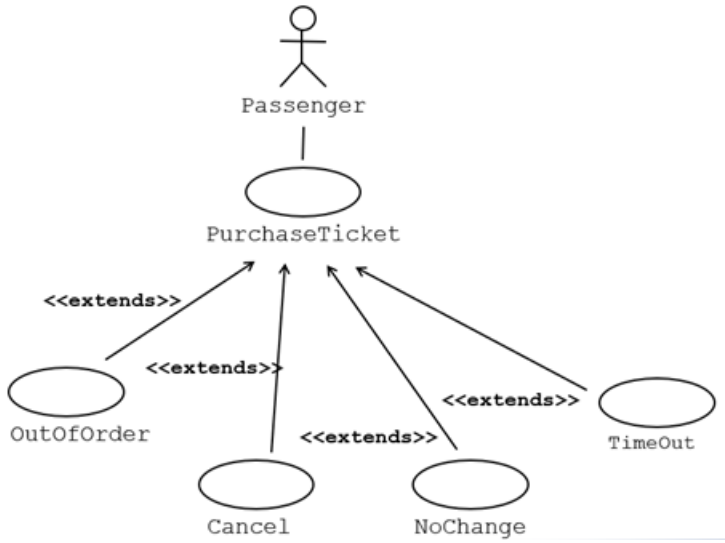
数据流图（Data Flow Diagram, DFD）

外部实体(方框)：指系统之外、又与系统有联系的事物，它表达了该系统数据的外部来源和去处。
数据加工(圆圈)：对数据的操作、变换。
数据存储(双横线)：某种数据保存后的逻辑统称，常见于xx记录、xx信息。
数据流(箭头)：处理功能的输入/输出；

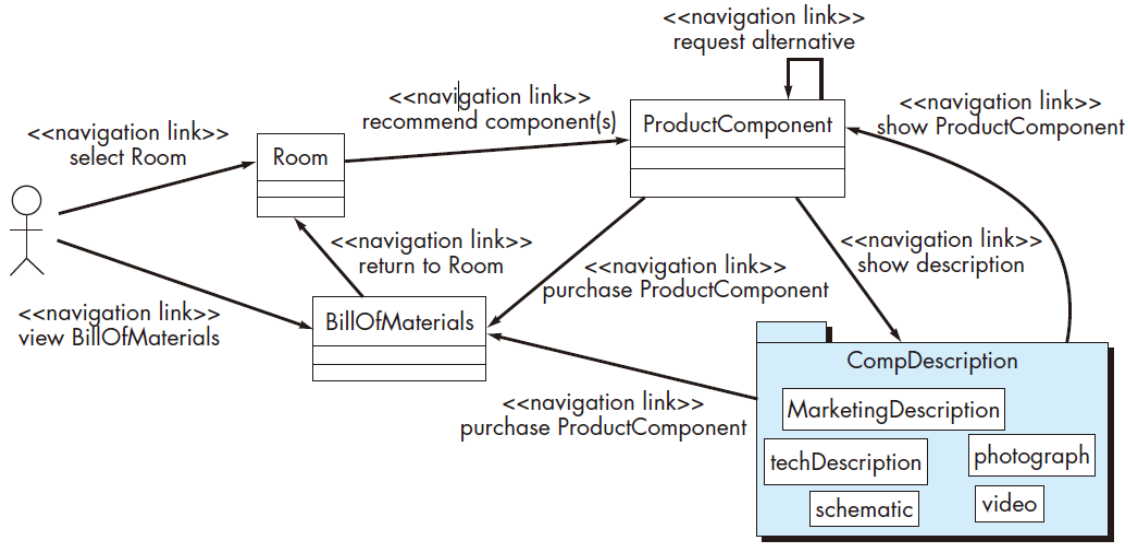


用例图（use-case diagram）

参与者(actor，小人)：位于系统之外的一类角色，也可以是某个外部的系统。
用例(use-case，椭圆)：系统能提供的一类功能。
关系(relationship，箭头)：包括拓展、包含和继承。
拓展：常常是处理特殊情况而来的特殊功能，箭头指向主用例，箭头上标注`<<extends>>`
包含：一个主用例包含子用例，常见于多个用例都使用了某一基本功能的情况，主用例指向字用例，箭头上标注`<<includes>>`
继承：处于继承关系中的用例在不同抽象层，其中被继承的一方是继承的一方更概括抽象的概念。例如：主用例是“用户识别”，“人脸识别”是用户识别的一种，“指纹识别”也是用户识别的一种。在继承关系中常常出现“...是...的一种”（is a kind of）这样的关系。由继承的一方指向被继承的一方，用空心箭头。



书上的NSU例子（虽然还是没懂具体如何组成）



CRC card

包括类名、说明、职责、以及每个职责对应的协作类

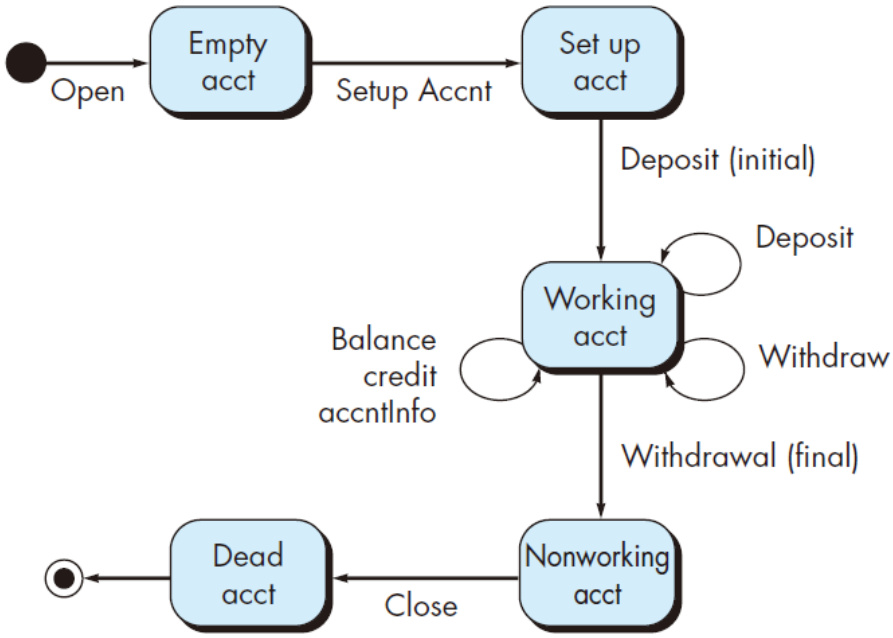
Class：销售类	
说明：完成一次销售	
职责：	协作类：
创建商品	商品类
计算总价	商品列表类
创建支付	支付类
计算找零	无

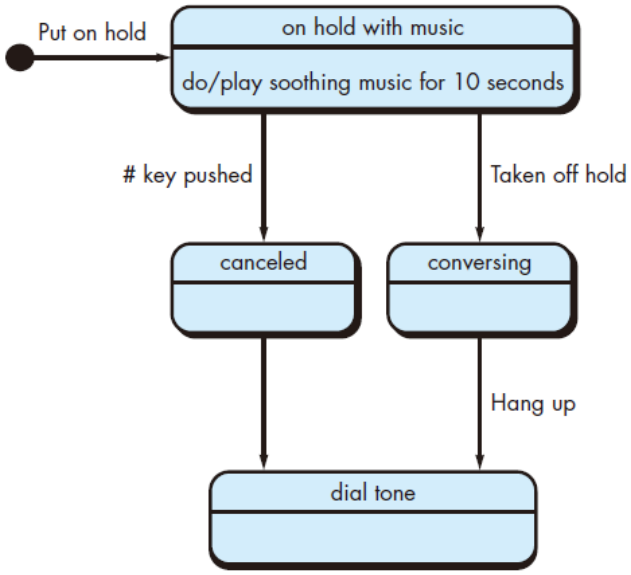
状态图

用来描述一个特定对象的所有可能状态及其引起状态转移的事件（类似DFA）

初始状态（实心圆圈）、终止状态（实心圆圈外加一个圆圈）、状态（分两格，上格放状态名称，下格放该状态要进行的动作）、转移、事件、动作

> entry/action：表示进入该状态时执行的动作。
> exit/action：表示退出该状态图时执行的动作。
> do/action：表示处于该状态时执行的动作。
> event/action：表示处于内部迁移状态时响应某个事件所执行的动作。



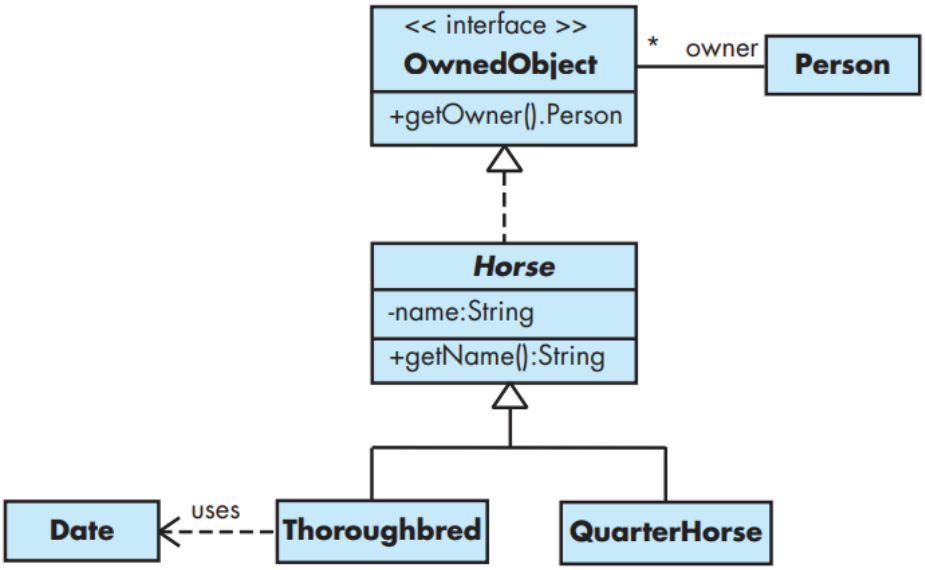


class图

从上到下分为三部分，分别是类名、属性和操作。类名是必须有的

> -表示private
> +表示public
> #表示protected

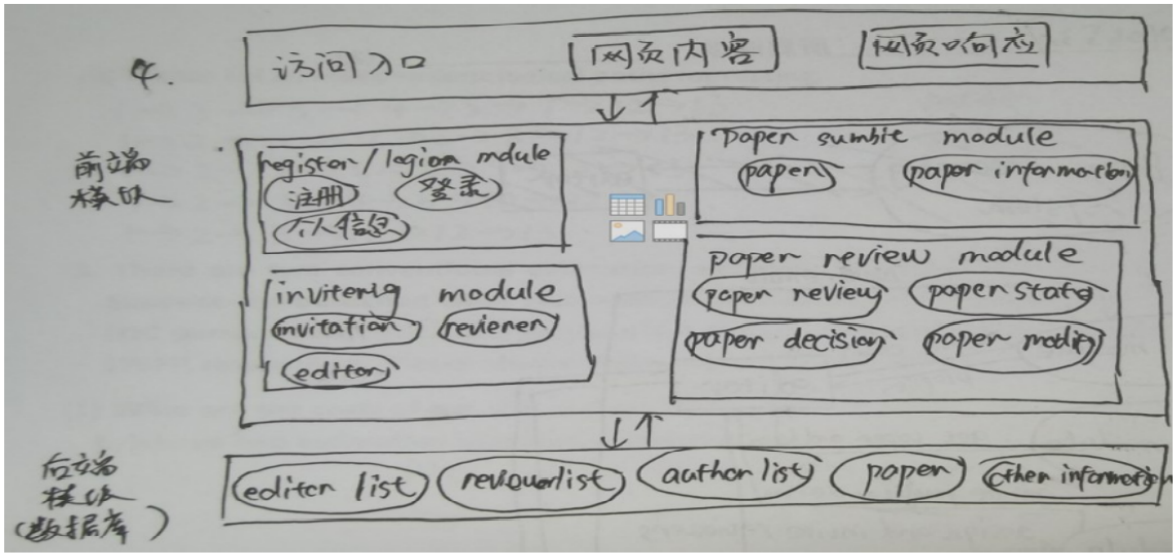
https://blog.csdn.net/qq_40332045/article/details/104026423



web-based software architecture

大致按以下示例回答吧（）

实例



test

单元测试 (Unit Testing)
又称模块测试。对软件的组成单位进行测试，其目的是检验软件
基本组成单位的正确性。测试的对象的是软件测试的最小单位：模块。

集成测试 (Integration Testing)
集成测试也称联合测试（联调）、组装测试：将程序模块采用适当的集成策略组装起来，对系统的接口及集成后的功能进行正确性检测的测试工作。集成主要目的是检查软件单位之间的接口是否正确。

系统测试 (System Testing)
系统测试：将软件系统看成是一个系统的测试。包括对功能、性能以及软件所运行的软硬件环境进行测试。时间大部分在系统测试执行阶段，包括回归测试和冒烟测试。

验收测试 (Acceptance Testing)
是部署软件之前的最后一个测试操作。它是技术测试的最后一个阶段，也称为交付测试。向软件购买者展示该软件系统满足原始需求。

性能测试 (Performance Testing)
检查系统是否满足需求规格说明书中规定的性能。
通常表现在以下几个方面：稳定性、响应时间、吞吐量（TPS）。

安全测试 (Safety Testing)
安全测试是一个相对独立的领域，需要更多的专业知识。如：WEB的安全测试、需要熟悉各种网络协议、防火墙、CDN、熟悉各种操作系统的漏洞、熟悉路由器等。

兼容性测试 (Campatibility Testing)
兼容性测试主要是指，软件之间能否很好的运作，会不会有影响、软件和硬件之间能否发挥很好的效率工作，会不会影响导致系统的崩溃。

易用性(用户体验性测试) (User ability testing)
易用性是交互的适应性、功能性和有效性的集中体现。又叫用户体验测试。

界面测试 (user interface Testing)
界面测试（简称UI测试），测试用户界面的功能模块的布局是否合理、整体风格是否一致、各个控件的放置位置是否符合客户使用习惯，此外还要测试界面操作便捷性、导航简单易懂性，页面元素的可用性，界面中文字是否正确，命名是否统一，页面是否美观，文字、图片组合是否完美等。

常考1 if语句测试需要最少case

For the statement if ((A>B)&&(C==D)||(E<=G)), what is the minimum number of test cases required to test every condition at least once? Please briefly verify your answer.

```
((A>B)&&(C==D)||(E<=G))

Define S = (A>B)&&(C==D).  Then the problem becomes (S || (E<=G)).
Lease cases needed: (T || T), (F || F).  That is, (T, <), (T, ==), (F, >).

For S being T: (T && T) = (>, ==).
For S being F: (T && F), (F && T). That is, (>, <), (>, >); (<, ==), (==, ==)

Hence the number is (S being T)*2 + (S being F) = 2 + 4 = 6.
```

if ((A>B)||(C==D))

5 (T, T) (F, F) => (>, T) (<, F) (=, F) => (>, =) (<, <) (<, >) (=, <) (=, >)

或的条件需要测双对和双错，与的条件需要测双对和一对一错；嵌套的情况合理使用乘法和加法。

常考2 面向对象相关

解释为什么封装、继承和多态是面向对象系统的三个重要特征。

答:类提供了一种封装(信息隐藏)机制，通过这种机制，数据(属性)的访问由一组操作控制。当适当地实现时，这将产生低耦合和高模块化的系统。继承提供了一种机制，通过这种机制，对高级类的更改可以快速传播到低级类。多态性允许许多不同的操作共享同一个名称，从而减少了扩展对象系统所需的工作量。

OOD(object-oriented design) 主要关心对象间的协作，SD(structured design) 主要关心构件间的控制流