哈尔滨工业大学 2013 年秋季学期

软件工程 试题(A)

学号	
姓名	

题号	1	2	3	4	5	6	总分
分数							

（本试卷满分 60 分）

注意行为规范

遵守考场纪律

主管领导审核签字

MyTime（www.mytime.com）是 2013 年 2 月在美国洛杉矶地区上线的一个本地商家预约平台。本地各类商家（餐馆、理发、健身、家政等）在该平台上发布自己的服务项目，顾客可在平台上进行服务预约。**MyTime** 的杀手锏是“分时段动态价格预约”，可根据不同时段客户需求的高低峰期来动态调整价格：商家自己设好标准价格和价格区间，**MyTime** 依据顾客在不同时段的预约数量来动态调整价格，从而提高商家的预订率。

(1) 已注册商家在 **mytime** 上添加自己可提供的服务项目，并为每个服务项目设定标准价格。每个商家可提供多个服务项目。

(2) mytime 可自动从已注册商家的 **Google Calendar** 中读取该商家各服务项目的可预约时间段（例如 **10** 月 **26** 日下午 **1:00-3:00** 为商家 **A** 的午餐服务项目的一个可预约时间段）。可预约时间段的长度是不同的，它取决于服务项目自身的特征，例如理发服务的时间段为 **1** 小时，餐馆服务的时间段为 **2** 小时。

(3) 针对不同服务项目的各个可预约时间段，商家在 **mytime** 上设定该时间段内可接受的价格区间（最高价格、最低价格），以及该时间段内可接受的最大预约数目。

(4) 顾客根据服务项目的关键字和自己期望的时间段在 **mytime** 上查询，找出所有满足要求的商家，并可看到 **mytime** 当前为其所设定的价格。**mytime** 按照商家与顾客当前位置之间的距离从近到远的次序排列查询结果。顾客选择某个商家的服务项目，形成预约单。

(5) 商家在 **mytime** 中可随时查看与自己的服务项目相关的预约单。顾客也可随时查看自己之前的所有预约单。

(6) 若某个顾客预约之后导致该商家在该时间段内的预约数目满足了某种特定规则，**mytime** 会马上启动动态价格调整，为该商家在该时段内的服务项目更新当前价格；若预约数目达到了商家设定的最大预约数目，则该时间段不能再被其他顾客预约。

(7) 每隔特定时间（例如 **10** 分钟），**mytime** 的系统也会周期性的启动价格动态调整，根据上一周期内的预约结果，对平台上所有商家各时间段内的服务项目价格进行变更。

(8) 不管是**(6)**还是**(7)**，**mytime** 均需为每个特定时间段内的服务项目记录其所有发生的价格动态变化信息（价格变化时间、原价格、新价格）。

(9) 对没有加入 **mytime** 的商家，**mytime** 员工在现实中收集他们的服务项目和价格，手工加入平台并根据经验指定可浮动的价格区间。顾客也可以按**(4)**的方式进行预约，**mytime** 也会按**(6)(7)**进行动态价格调整。顾客预约之后，系统会为 **mytime** 员工生成提醒信息。员工在系统中查看到提醒信息之后，打电话给相应的商家以帮助顾客完成预约，之后将电话确认的时间记录在顾客最初的预约单中，并记录该提醒信息已被电话确认。

(10) mytime 也可整合各商家在 **Yelp**（美国版的大众点评网）上的顾客点评，也可将顾客在 **mytime** 上对各商家的点评同步到 **Yelp** 上。

1. 用例建模(11 分)

(1) (8 分) 建立该系统的用例模型。无需绘图，以表格形式给出，第一列为 **actor**，第二列为与每个 **actor** 产生通讯关联的用例清单。**actor** 名和用例名均使用简短和无歧义的中文短语。请手工在表格中绘制水平线，以便于区分不同 **actor** 的用例集合。

Actor	用例清单

(2) (3 分) 以下给出了用例**“**顾客预约商家**”**的事件流描述，但不完整。根据上页需求陈述，补充其缺失的三个备选（扩展）事件流描述。

当顾客处于查询结果页面（展现 **mytime** 中满足条件的服务项目及其价格）时，执行该用例。
常规事件流：
1. 顾客从列表中选择某一商家的某一时间段，选择**“**预约**”**；
2. 系统生成预约单，展示预约单给顾客；
备选（扩展）事件流：
1.

2.

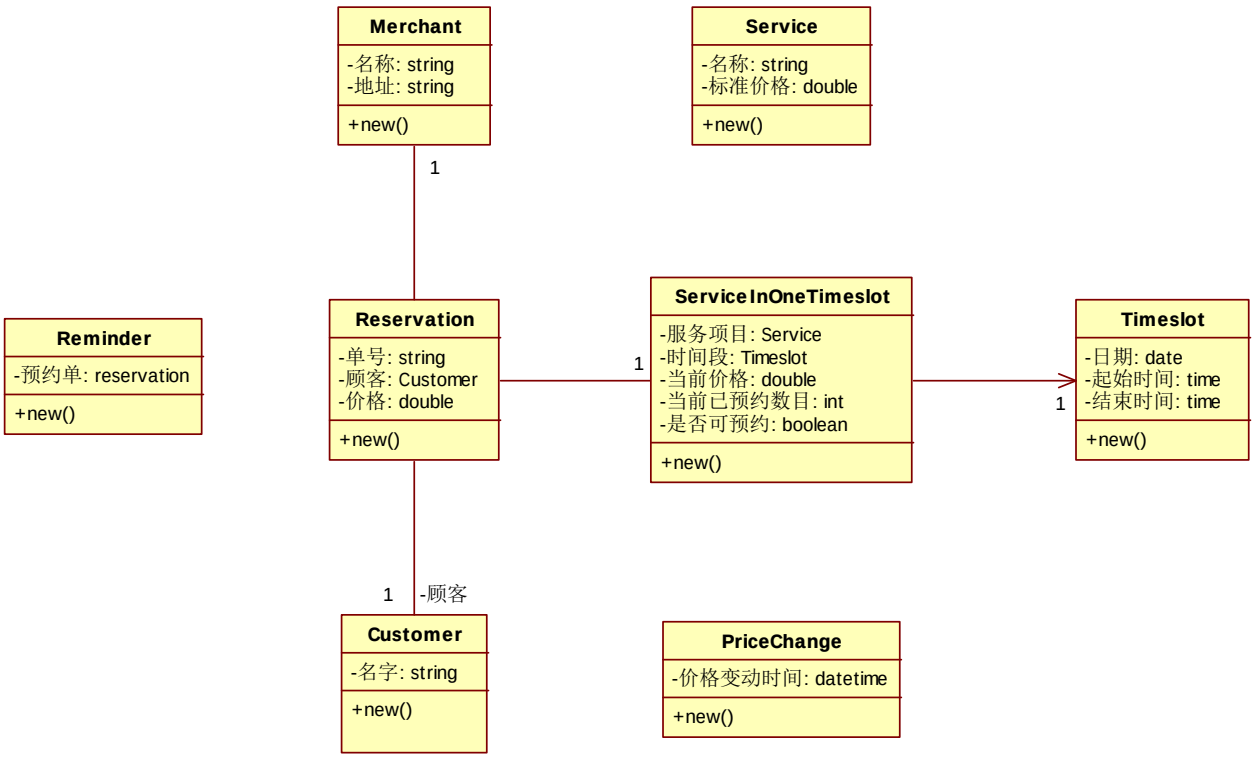
3.

2. 领域建模(13 分)

采用 OO 对该系统进行需求分析，所需的实体类及它们的部分属性如下表所示：

实体类	说明	属性
Merchant	在 **mytime** 上注册并提供服务的组织	名称、地址
Customer	在 **mytime** 上预约服务项目的个人	名字
Service	商家所提供的服务项目（理发、健身等）	名称、标准价格
Timeslot	商家的服务项目的可预约时间段	日期(如 **2013/11/26**)、起始时间、结束时间
ServiceInOne Timeslot	商家的一个服务项目在特定时间段内的详细信息	服务项目、时间段、当前价格、当前已预约数、是否可预约**(Y/N)**
Reservation	顾客针对特定商家在特定时间段内的某个服务项目的预约单	单号、顾客、价格
PriceChange	特定时间段内的一个服务项目的一次价格动态变化信息	价格变动时间
Reminder	对 **mytime** 员工的提醒，描述一个顾客对未注册商家的服务项目的预约信息	预约单

(1) (5 分) 下图给出了这些实体类之间的关系描述，但并不完整。请根据你对需求的理解以及上述表格所做出的限定，直接在类图上补充缺失的类间关系（继承、聚合、组合、关联），并补全各关系上的角色名和多重性信息。注：不能删除图中已有信息，只能增加或修改；无需补充或修改类的属性或操作。



(2) (8 分) 为了支持 mytime 各用例的事件流执行，上页表格和类图中给出的属性与操作是远不够的。为各实体类补充缺失的操作、属性及其数据类型，填入下表。若你认为某个（些）实体类的属性或操作不缺失，相应的格子可以留空。属性/操作名均用易理解的中文短语。

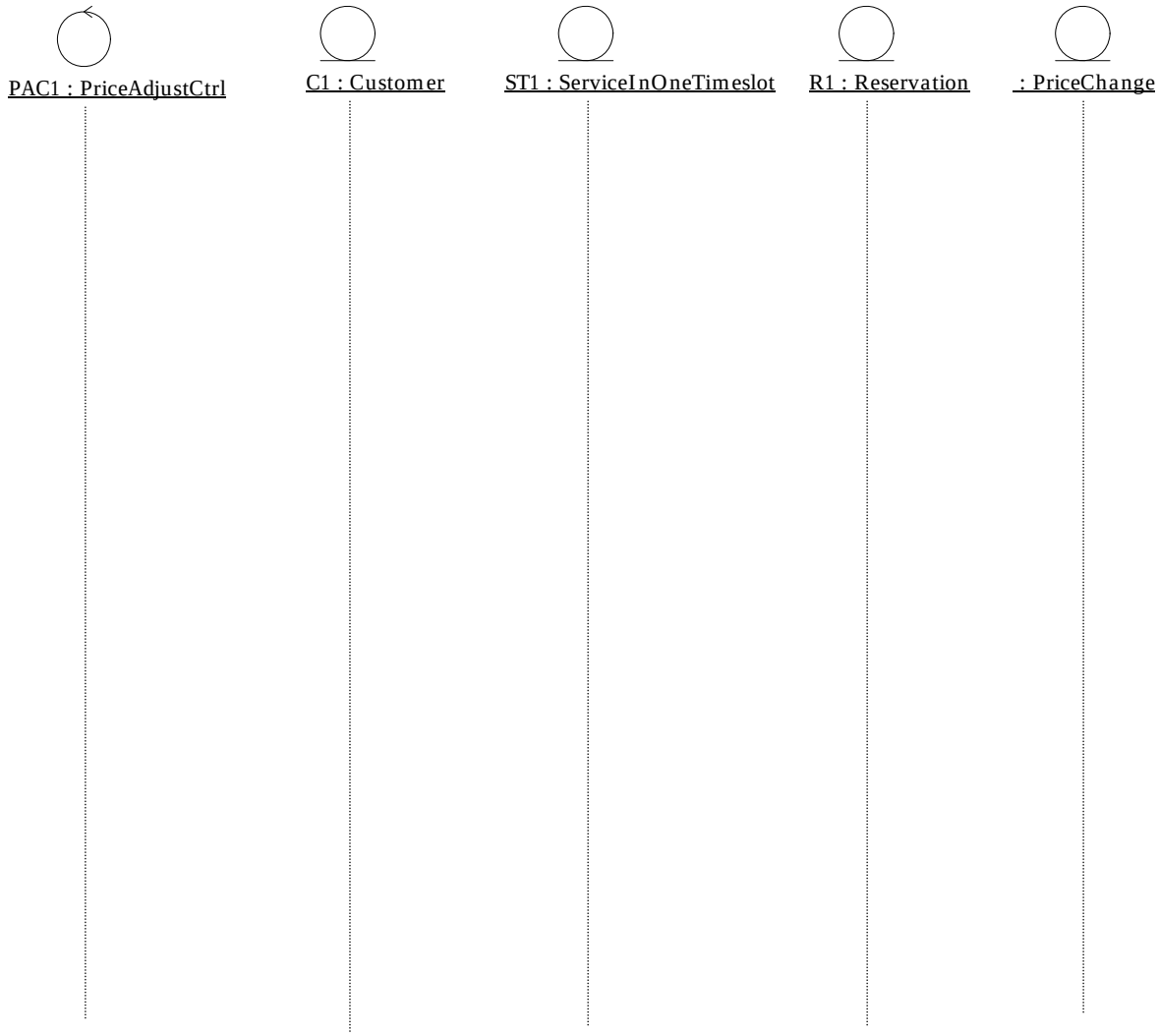
实体类	缺失的属性及各自的数据类型		缺失的操作
Merchant			
Customer			
Service			
Timeslot			
ServiceInOne Timeslot			
Reservation			
PriceChange			
Reminder			

3. 时序模型**(5** 分**)**

考虑 **mytime** 系统运行过程中的一个场景：某顾客 C_1 预约了某个时间段的某个服务项目 ST_1 并形成了预约单 R_1。假设此时程序运行空间内具有以下 **4** 个对象实例：

- 用于进行服务项目价格动态调整的控制类 **PriceAdjustCtrl** 的对象 PAC_1
- **Customer** 类的对象 C_1
- **Reservation** 类的对象 R_1
- 与 R_1 相关联的 **ServiceInOneTimeslot** 类的对象 ST_1

请在下图的基础上补全对象时序模型，使其可以覆盖需求描述中的第**(6)**和**(8)**项。起始点为控制类的 PAC_1 对象，按执行次序加入所需的消息序列，需要覆盖各 **object** 为了完成这些功能所需执行的全部操作。箭头上给出操作名，所有操作需来自于你在第 **2** 题第**(2)**小题中所识别的实体类操作列表。图中给出的各对象未必都要参加交互。



4. 架构设计(8 分)

Mytime 提供了 **iOS** 平台上的客户端 **app**，可借助手机 **GPS** 对顾客进行定位，方便顾客查询和预约自己当前位置周围的商家服务项目；向注册商家提供 **web** 端访问；向 **mytime** 员工提供 **C/S** 方式访问系统。后台系统则采用标准的 **app server** 和 **database** 分离的方式对上述三种客户端提供服务，并支持与 **yelp**、**Google Calendar** 的连接。

(1) (6 分)使用 **UML** 部署模型简要刻画上述所描述的 **mytime** 系统的物理架构。

以下问题 **2** 选 **1**，共 **2** 分。

(2) 若在 **database** 层采用分布式策略存储数据，可以采用何种方式来切分 **mytime** 内部存储的数据，从而可有效降低客户端请求的响应时间？

(3) 若在 **app server** 层和 **database server** 层之间增加 **cache** 来提高数据读取的效率，你认为能 **cache** 哪些数据、不能 **cache** 哪些？为什么？

5. 软件测试(14 分)

ServiceInOneTimeslot类中有一个操作**DynamicPricing()**，它根据特定的价格调整策略，对该类的属性“当前价格”**(livePrice)**做出修改。以下是该操作的一段伪代码：

```
    public void DynamicPricing (){
1       int min = MAX_VALUE;
2       for(int i=0; i < this.resvNum; i++) {
3           int price = this.ResvList[i].getPrice();
4           if(min > price)
5               min = price;
        }
6       if( (this.timeslot.getDate() - currentDate()) > 5 days
            || this.resvNum < this.maxAvailableResvNum * 0.3) {
7           if (this.livePrice * 0.9 < min)
8               this.livePrice *= 0.9;
            else
9               this.livePrice = min;
        }
        else
10          this.livePrice = this.maxPrice;
11      return;
    }
```

(1) (5 分) 针对上述代码的第 **6-10** 行，设计一组测试用例填入下表，使其同时满足语句覆盖和条件覆盖的要求。每个测试用例代表一个 **ServiceInOneTimeslot** 对象，前四列给出它的四个属性的值，最后一列填写该测试用例覆盖了第 **8**、**9**、**10** 行中的哪些行。注：假设前 **5** 行代码执行之后变量 **min=100**；第 **6** 行的 **currentDate()**执行结果为今天（**2013/11/26**）。另外，虽然表格给出了 **4** 行，但你未必一定将其全部填满。

测试用例编号	**timeslot.getDate()** (时间段，只需给出其日期属性的值)	**maxAvailableResvNum** (最大可预约数目)	**resvNum** (当前已预约数)	**livePrice** (当前价格)	覆盖的代码行
1					
2					
3					
4					

(2) (3 分) 绘制该程序的控制流图；

(3) (6 分) 列出所有的基本路径。

回答上述题目时请务必使用代码行号（1-11）。若需将第 i 行拆分为多行，请以 i1、i2 的形式表示。自己进行重新编号的答案不得分。

6. 简答题**(9** 分，**5** 选 **3**，每个 **3** 分，若答题超过 **3** 个，按所答前 **3** 个题目记分**)**

(1) “敏捷过程模型是增量方法和迭代方法的复合”。你是否认同这个观点？不管是否认同，均请给出理由。

(2) 在 **MVC** 架构中，若不使用 **Controller**，而是由 **View** 对 **Model** 进行直接调用，**Model** 将结果直接返回 **View**，会带来哪些不好的影响？给出理由。

(3) 结构化程序设计方法强调“高内聚、低耦合”，**OO** 设计方法强调应做到“类的责任单一”、“在不修改原有类代码的前提下实现功能扩展”。你认为这些设计原则共同追求的 **NFR** 是什么？为什么在设计中做到这些原则可以使该 **NFR** 变得更好？

(4) 传统观点认为：“在软件开发过程中越早开始写程序，就要花越长时间才能完成它”，产业界的统计数据也表明：“在一个程序上所投入的 **50-70**％是花费在第一次将程序交付给用户之后”。不过，最新的软件开发方法论则认为“应该尽早交付可运行的程序”。你怎么看待这两个看似矛盾的观点？

(5) 经常看到“某某公司的业务系统为了维护和升级需要耗费大量的时间，甚至导致系统不可用”的新闻报道。例如 **2013** 年 **6** 月 **23** 日全国各地工商银行发生故障，系统非常缓慢导致业务停办，其原因是由于系统的数据库升级时考虑不全面所导致的。你认为系统在升级时最大的难点是什么、有什么规避措施？