**简易约车系统设计文档**

**2018/4/22 王梓嘉**

1. **需求分析**

一个简易的约车系统至少包括乘客和出租车两个元素，他们需要在一定区域上分布上移动，因此还需要地图来提供位置信息，乘客向出租车传递信息的方式为发出请求，系统将通过调度系统进行任务分配。综上所述，至少应有乘客，出租车，地图，请求，调度器这五类。下面简单分析一下它们各自的需求：

1. 乘客

乘客出现在地图的某一个位置，拥有一个唯一的ID标识，可发出一个请求来呼叫出租车，乘客指定请求的目的地。

1. 出租车

出租车出现在地图的某一个位置，拥有一个唯一的ID标识，有如下几个状态：

1. 停止运行：出租车处于等待服务状态下且连续运行一定时间（没接到客）则需要停止运行一段时间再运行。
2. 等待服务：出租车处于运行状态且车内无乘客且未接到单。
3. 服务：出租车处于运行状态且车内有乘客。
4. 正在接单：出租车处于运行状态且车内无乘客，但已接到单，正在前往始发地。

出租车在等待服务状态时会进行抢单操作，抢单后等待调度器分配请求（有可能被其他出租车抢走），如果最终分配到的请求目的地无效，则拒绝，状态不变，否则变为接单状态，按照流程进行，服务完成后变回等待服务状态。

出租车还有信用度这一属性，初始为0，系统根据出租车请求完成状态和乘客评价动态更新其信用度。

1. 地图

地图是一张有一定大小的网格图（模拟经纬度），保存地形等信息，用于获取出租车运行规则（绕过障碍物等）。

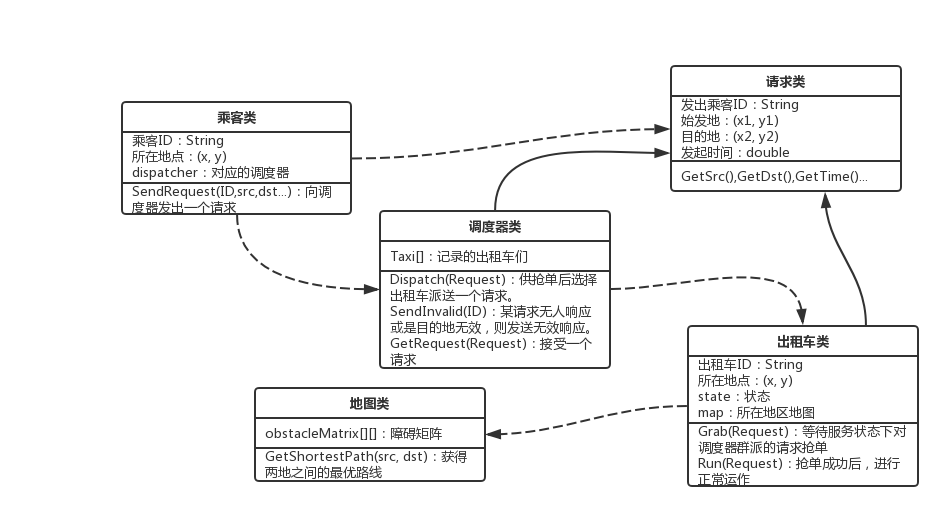
1. 请求

请求为乘客发出的，包含乘客ID，始发地，目的地，请求发起时间。

1. 调度器

乘客的请求不是直接传达给出租车，而是先传达给调度器，由调度器根据一定的规则（比如离始发地前k近的k辆车收到该请求）选择出租车群派请求供抢单。对于同一个请求，可能会有多辆车抢单，此时调度器选择信用度最高的出租车，此时该订单正式由该出租车完成。所有这些都设置最大延迟时间，超过某一时间的抢单视为无效，一定时间内无抢单则该请求无效，调度器传达给请求发起者无车响应信息。

1. **类图设计**



1. **SOLID分析**
2. SRP单一职责原则

每个类或者方法只有一个明确的职责。在上面的设计中，乘客模拟乘客发送请求的行为；请求类单纯的保存一个请求的所有信息，并提供查询接口；地图类保存一个区域的地形信息，给出查询两点之间的最优路径接口；调度器类对乘客发送的请求进行处理和分发，对无效或无响应请求进行回报；出租车类模拟出租车的行为，根据状态的变化进行抢单和送单操作。

1. OCP开闭原则

无需修改已有实现，而利用扩展来实现新功能。在本设计中，没有已有的实现可以进行扩展。

1. LSP里氏替换原则

任何父类出现的地方都可以使用子类代替且不会使得使用相应类的程序出现错误。本设计中没有用到继承。

1. ISP接口分离原则

一个类实现接口时，必须实现接口中的所有方法，因此接口中不宜囊括过多的方法。本设计中，各个类没有相似的行为动作，因此不需设计接口来实现。

1. DIP依赖倒置原则

要依赖于抽象而不依赖于具体编程，即面向接口而不面向实现编程。本例中设计接口的需求不明显。

1. **线程安全设计**

由于PPT给出的信息不足，因此只能做一个初步的设计。首先每辆出租车设计为一个线程，自动进行状态的转换，在不同状态做不同的动作，调度器也作为一个线程，每接收到一个请求，则向一定范围内的出租车发送抢单信息，后此线程睡眠一段时间，各出租车接收到以后进行响应（通过修改调度器的待调度请求名单），一些涉及对象访问冲突的方法设置代码块同步（synchronized），调度器线程恢复后，选择一个出租车进行派发请求，该车接收后进行状态转换，其余出租车状态不变。以上是大致的运行思路。