**桌面日程表**

**概要设计**

1. **前言**

该概要设计是以需求分析文档为基础，沿用思路继续设计进而得到。

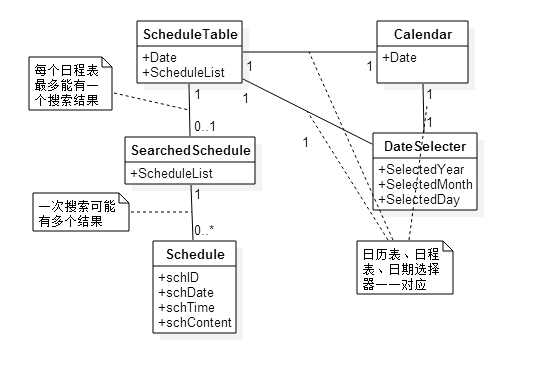
需求分析文档中已经包含

1. 设计背景
2. 需求分析
3. 用例说明
4. 显示流程

且已经较为完善。

文档通过需求分析、用例说明以及主要流程进一步分析得出，以用户需求为基础进一步完善了项目的设计思路，并且在此基础给出了数据库设计等方面的设计。

1. **领域模型**

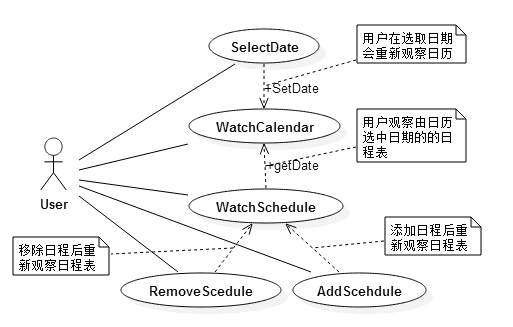
该项目领域模型如下图所示：  


(在该领域模型中，以实体类作为模块)

图中Calendar为日历类，与ScheduleTable为一对一关系，提供日历与日程表显示。用户通过DataSelecter进行日期设定并，因而与Calendar与ScheduleTable也为1对1关系。SearchedSchedule用处为按日期搜索日程，为需要调用时才创建的实体类，故和ScheduleTable为0..1对1关系。Schedule为日程实体，一次搜索中可能有多个日程，故为N对1关系。

1. 用例解释

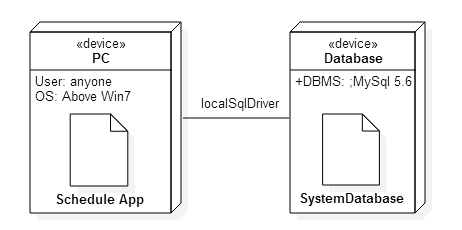
在原用例图的的基础上进一步的进行了解释，为后续设计提供方便。



(该应用用例较为简单，添加解释为主)

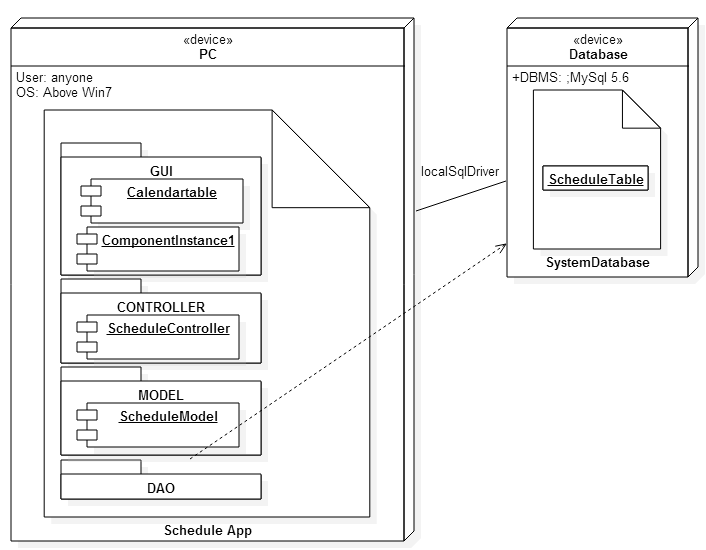
1. **架构设计**

第一层拆解：



该图指出了程序所运行的环境，数据库端所对应的数据库等，并给出了应用与数据库之间的联系

第二层拆解：



1. **数据库设计**

|  |  |  |  |
| --- | --- | --- | --- |
| schID(Primary Key)(Increment) | schDate | schTime | message |
| Int(10) | Date | Time | varchar(100) |

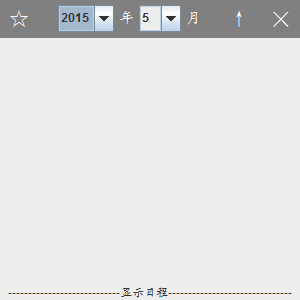
日程表对应的数据库中以日程的ID为主键，日程的日期时间和信息为记录，满足第二范式原则。

1. **UI设计**

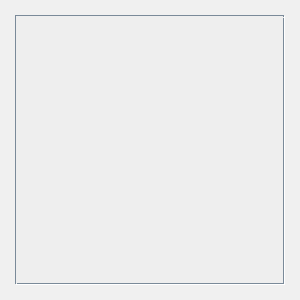
UI方面由于该应用为桌面组件形式，需要小巧灵活，能够浮于其他窗口上方，且有一定的透明性。当用户使用时可以轻松的拖拽该工具的日历部分至想要的位置并呼出日程表。

以下为早期完成的UI，尚未实现日历与日程表，并且无法实现透明等需求。

日历页面:



日程表显示页面



1. **其他要求**

对于日程表而言，需要保障的其他要求有：

能够悬浮于其他窗口之上方便查看

能够对输入的日程数据进行判断是否合法

能够自动根据选中日期给出日程表