Software Design Document

Project: 校园超速监控系统

Authors: 小组成员的姓名+学号

Published on: 2024-XX-YY

Version: 1.0

Background:

Software design is a process by which the software requirements are translated into a representation of software components, interfaces, and data necessary for the implementation phase. The SDD shows how the software system will be structured to satisfy the requirements. It is the primary reference for code development and, therefore, it must contain all the information required by a programmer to write code. The SDD is performed in two stages. The first is a preliminary design in which the overall system architecture and data architecture is defined. In the second stage—i.e., the detailed design stage—more detailed data structures are defined and algorithms are developed for the defined architecture.

This template is an annotated outline for a software design document adapted from the *IEEE Recommended Practice for Software Design Descriptions*. The *IEEE Recommended Practice for Software Design Descriptions* has been reduced in order to simplify this assignment while still retaining the main components and providing a general idea of a project definition report. For your own information, please refer to IEEE Std 1016[[1]](#footnote-0) for the full *IEEE Recommended Practice for Software Design Descriptions*.

**Table of Contents**

[1. Introduction 4](#_Toc169080215)

[1.1 Purpose 4](#_Toc169080216)

[1.2 Scope 4](#_Toc169080217)

[1.3 References 4](#_Toc169080218)

[1.4 Definitions and Acronyms 5](#_Toc169080219)

[2. System Overview 6](#_Toc169080220)

[3. System Architecture 7](#_Toc169080221)

[3.1 Architectural Design 7](#_Toc169080222)

[3.2 Decomposition Description 7](#_Toc169080223)

[3.3 Design Rationale 8](#_Toc169080224)

[4. Data Design 9](#_Toc169080225)

[4.1 Data Dictionary 9](#_Toc169080226)

[4.2 Data Storage 9](#_Toc169080227)

[5. Component Design 10](#_Toc169080228)

[6. Human Interface Design 11](#_Toc169080229)

[6.1 Overview of User Interface 11](#_Toc169080230)

[6.2 Screen Images 11](#_Toc169080231)

[6.3 Screen Objects and Actions 11](#_Toc169080232)

[7. Requirements Matrix 12](#_Toc169080233)

[8. Appendices 13](#_Toc169080234)

# Introduction

## Purpose

这份软件设计文档的目的是提供校园超速监控系统设计的详细描述。本文件的预期读者包括负责实施系统的开发人员以及参与系统开发、测试和维护的相关利益方。该文档将提供对软件架构和系统设计、功能和数据结构以及实现细节的全面理解，以帮助编写测试用例，并确保系统满足SRS文档中列出的要求。此外，本文件还将有助于系统的持续维护。

## Scope

本开发软件是一个校园超速监控系统，旨在监控在校园内行驶的车辆速度。该系统由部署在校园关键位置的多个车速检测器、多个车速显示器以及一个后台管理系统组成。系统的目标是通过检测并警告超过安全速度限制的司机、通过自动短信通知教育违规者以及生成分析和改进报告来提高校园内的道路安全。校园超速监控系统的好处包括减少超速导致的事故和提高校园内的整体道路安全。该系统还为学校的安保部门提供了一种有效管理交通违规行为和教育司机安全驾驶的方式。系统可以根据学校安保部门的具体需求轻松定制安全速度限制和通知频率。

## References

无

## Definitions and Acronyms

无

# System Overview

校园超速监控系统旨在监控在校园内行驶的车辆速度，并识别那些超过指定安全速度限制（如30公里/小时）的车辆。该系统包括部署在校园关键位置的多个车速检测器，以及显示安全速度限制的车速显示器。此外，还有一个可以通过PC和移动设备访问的后台管理系统，访问权限仅限于校园安保部门的授权用户。

车速检测器记录经过关键位置的所有车辆的速度和车牌号，并将其显示在路边显示屏上。系统还将检测记录存储在后台管理系统中。如果车辆超速，后台管理系统会自动向司机（如果是教职工）发送警告信息并教育其安全驾驶。对于临时访客，系统根据注册的车辆信息发送类似的警告信息，并通知校园内的相关人员。

后台管理系统可以定期进行统计处理，包括在每月最后一天中午通知责任单位领导那些当月超速三次或以上的教职工，还会在每季度最后一天中午将超速五次或以上的车辆列入黑名单，取消其下季度进入校园的权限。校园安保部门的指定负责人可以通过后台管理系统调整安全速度限制以及每月/每季度的超速次数限制。该负责人还可以查看各个校园单位的月度和季度报告，并添加或删除普通的后台管理人员。普通后台管理人员可以按车辆、司机或路段查看详细记录。

所有后台管理系统的授权用户还可以更新他们的个人信息，如电话号码和地址。校园超速监控系统通过监控车辆速度和教育司机安全驾驶，提供了确保校园安全的有效解决方案。

# System Architecture

## Architectural Design

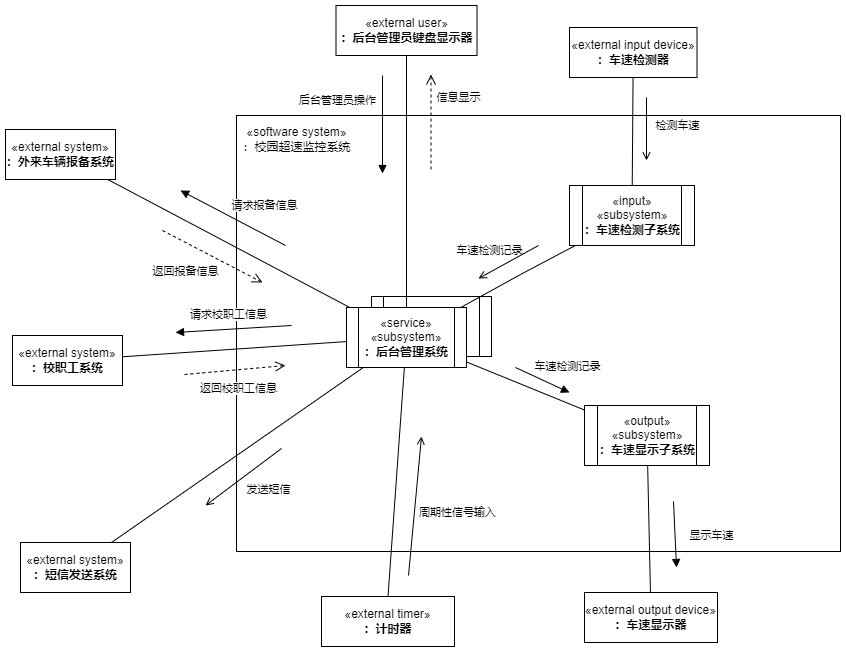
1. 运用了外部对象接口准则、客户端和服务准则、组合对象准则

车速检测子系统：该子系统负责检测在校园内关键位置经过的车辆的速度。它将由多个放置在校园指定点的车速检测器组成。每个检测器都能捕捉车辆的车牌号和速度，这些数据将显示在附近的车速显示器上，并发送到后台管理系统。该子系统捕获的数据将用于检测超速车辆。

车速显示子系统：该子系统接收来自车速检测子系统的信息，并向驾驶员显示检测到的每辆车的速度。该子系统还包括一个显示该区域安全速度限制的机制。显示子系统帮助驾驶员在校园内行驶注意车速。

后台管理子系统：该子系统负责管理所有由车速检测子系统收集的数据。它可以通过PC和移动设备访问，仅限校园保安部门授权人员使用。该子系统将存储所有由车速检测子系统发送的速度检测记录。此外，它还具有管理警告或通知、生成报告以及维护超速车辆黑名单的功能。

1. 子系统之间的并发通信图



## Decomposition Description

**Provide a decomposition of the subsystems in the architectural design.** Supplement with text as needed. You may choose to give a functional description or an object-oriented (OO) description. For a functional description, put top-level data flow diagram (DFD) and structural decomposition diagrams. For an OO description, put subsystem model, object diagrams, generalization hierarchy diagram(s) (if any), aggregation hierarchy diagram(s) (if any), **interface specifications**, and sequence diagrams here.

要写 -- 这里放置**【子系统设计】**结果：

1. 各个子系统的并发通信图；

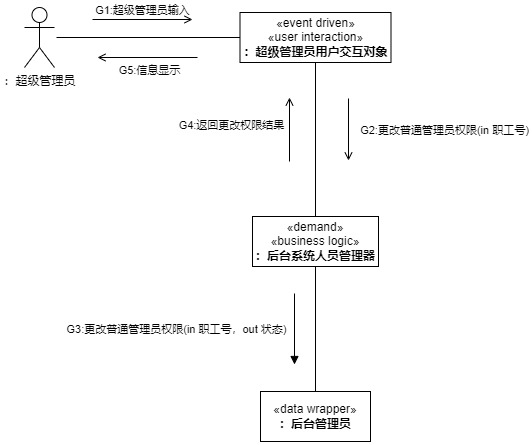
2. 所有对象及消息类型的确定依据，如运用的并发任务组织准则等。// 相当于 **Design Rationale**

3. 类/任务的接口设计（不要求每个对象/任务的通信图）-- 因为实体类的操作将在4.1小节给出，这里仅给出实体类以外的其它类/任务的接口。

**【特别说明】**：这里本应按照上述要求，放置子系统设计结果，但考虑到工作量较大，每个小组按照人数，**每人书写1个用例的并发通信图及对象/消息类型的确定依据**。另外，根据所选择的2-3个用例，**每人至少书写1个类/任务（实体类除外）的接口设计**，包括以目标类/任务为焦点的并发通信图（画了）、该类的完整类图（画了）及该类/任务的接口规约（写了）。

**用例并发通信图**：在动态交互建模阶段的用例图基础上，区分每个软件对象的主动和被动，并标识出主动对象的并发类型（event-driven、periodic、demand），还要区分对象之间每条消息的通信类型（即同步/异步）。对于每个对象及每条消息的类型，给出其判定依据，参考理论课第21讲PPT的P16-17、P22-23。

**类/任务的接口设计**：先从小组选择的2-3个用例图中确定需要进行接口设计的2-3个类/任务；然后，针对每个选定的类/任务，参照实验课第6讲PPT的任务进行类/任务的并发通信图及类图绘制，示例见P12、P17。类/任务的接口规约参照理论课第18讲P43-45及理论课第20讲P58-60。对于任务（即主动类），若考虑它的操作，则其接口规约需要同时包含类接 口规约（描述操作）与任务接口规约（描述消息）的相应部分。



**对象类型的确定依据：**

1. 超级管理员用户交互对象：一个用户交互对象，在显示器上输出消息并从键盘上接收输入，需要独立的控制线程，而键盘是事件驱动的外部I/O设备，被构造成一个事件驱动的用户交互任务。

2. 后台管理人员系统管理器，一个应用逻辑对象，需要独立的控制线程处理不同来源的信息，被构造成一个按需驱动的控制任务。

3. 后台管理员，一个数据包装器类对象，需要长期存储在数据库中，被构造成一个数据包装器类对象。

**消息类型的确定依据：**

1. 超级管理员输入：超级管理员向后台管理员系统控制发送更改普通管理员权限请求后，无需等待响应即可继续执行，因此为异步信息。

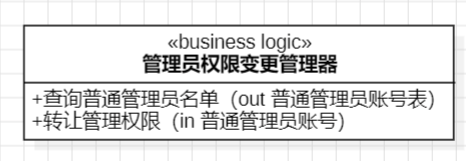
2. 更改普通管理员权限(in 职工号)：超级管理员用户选择发送更改普通管理员权限请求，无需等待响应即可继续执行，因此为异步信息。

3. 更改普通管理员(in 职工号，out 状态)：由被动的数据包装器对象确定的任务交互，任务会调用被动对象的操作，必是同步调用，因此为同步消息。

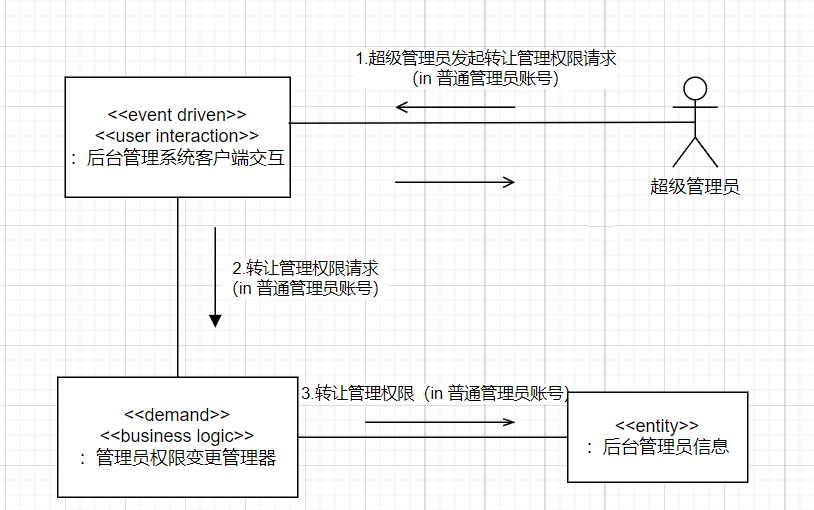
4. 返回更改权限结果：后台管理人员管理器对象返回更改权限结果后不需要再等待回复即可继续执行，因此为异步消息。

**管理员权限变更管理器类的接口设计**

**1.完整类图**



**2.并发通信图**



**3.接口规约**

信息隐藏类：管理员权限变更管理器

隐藏的信息：后台处理超级管理员转让管理权限给普通管理员请求的具体逻辑

类组织准则：业务逻辑对象

假设：无

预期的变化：超级管理员将自己管理权限转让给系统内的一位普通管理员

父类：无

继承的操作：无

提供的操作：

1. 查询普通管理员名单（out普通管理员账号表：String）

功能：向超级管理员展示目前普通管理人员的账号

前置条件：无

后置条件：后台管理系统客户端展示普通管理员账号。

输入参数：无

输出参数：普通管理员账号表——记录了目前系统内所有普通管理员的账号，供超级管理员选择

使用的操作：无

1. 转让管理权限（in 普通管理员账号）

功能：为相应账号的普通管理员赋予超级管理员的身份，并将原超级管理员的身份降级为普通管理员。

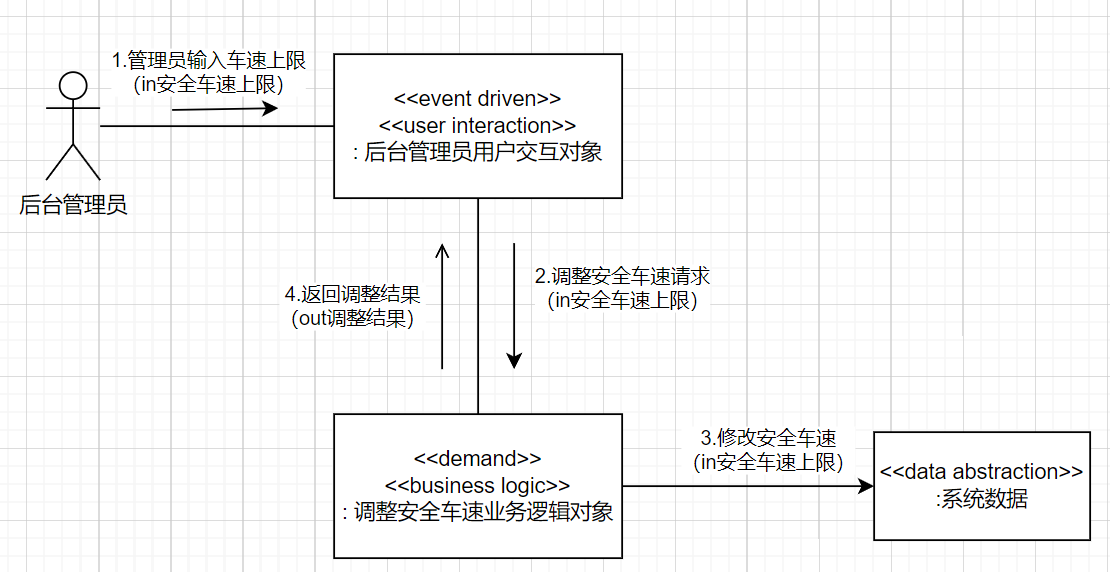
前置条件：无

后置条件：管理权限转让成功

输入参数：普通管理员账号——接受转让的普通管理员账号

输出参数：无

使用的操作：无



**对象类型的确定依据：**

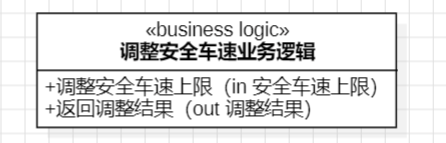
1. 后台管理员用户交互对象：一个用户交互对象，在显示器上输出消息并从键盘上接收输入，需要独立的控制线程，而键盘是事件驱动的外部I/O设备，被构造成一个事件驱动的用户交互任务。
2. 调整安全车速业务逻辑对象：一个业务逻辑对象，负责处理安全车速相关的业务逻辑，需要独立的控制线程，被构造成按需驱动的业务逻辑对象。
3. 系统数据：系统数据不需要储存在数据库中，是一个数据抽象类

**消息类型的确定依据：**

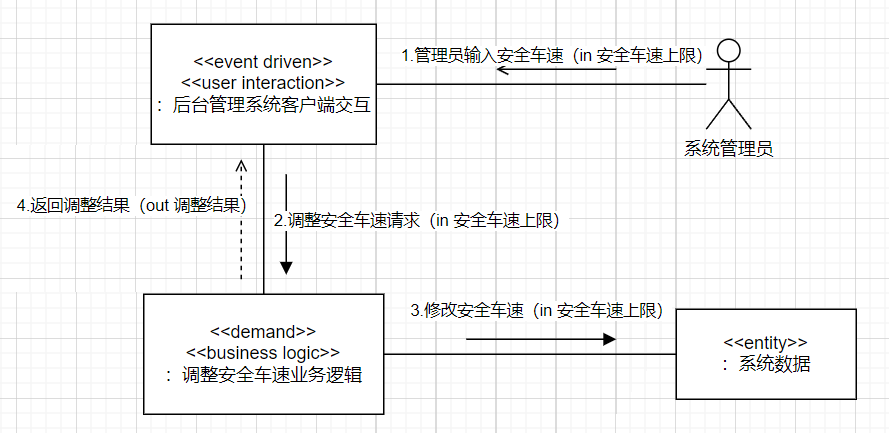
1. 管理员输入车速上限（in 安全车速上限）：管理员输入车速上限后，需等待客户端交互对象进行回应修改是否成功，才可以进一步操作，所以是同步消息。
2. 调整安全车速请求（in 安全车速上限）：客户端交互对象向业务逻辑对象发送车速上限更改请求，需等待逻辑对象处理成功才能继续，所以为同步消息。
3. 修改安全车速（in 安全车速上限）：业务逻辑对象修改系统数据，需要得到修改成功与否的回应，是同步消息。
4. 返回调整结果（out 调整结果）：业务逻辑对象给客户端交互对象发送修改结果，无需等待回复，可以立刻再次发送（假设有别的消息要发送），故为异步消息。

**调整安全车速业务逻辑类的接口设计**

**1.完整类图**



**2.并发通信图**



1. **接口规约**

信息隐藏类：调整安全车速业务逻辑类

隐藏的信息：封装了调整安全车速业务的应用逻辑

类组织准则：业务逻辑类

假设：无

预期的变化：用户输入合法的安全车速上限

父类：无

继承的操作：无

提供的操作：

1. 调整安全车速上限（in 时间范围（月度/季度）：String， in 安全车速上限：Real）

功能：根据用户输入调整月度或季度的安全车速上限

前置条件：无

后置条件：若安全车速上限合法，则更改系统数据中的安全车速上限。若不合法，则不加以修改并提示用户。

输入参数：安全车速上限——管理员期望的新安全车速上限

输出参数：无

使用的操作：无

1. 返回调整车速结果（out 调整结果：String）

功能：展示调整车速结果是否成功

前置条件：管理员尝试调整车速

后置条件：若成功调整，则返回是。如调整失败，则返回失败原因。

输入参数：无

输出参数：调整成功信息——告知管理员操作是否成功，如不成功则显示原因

使用的操作：无

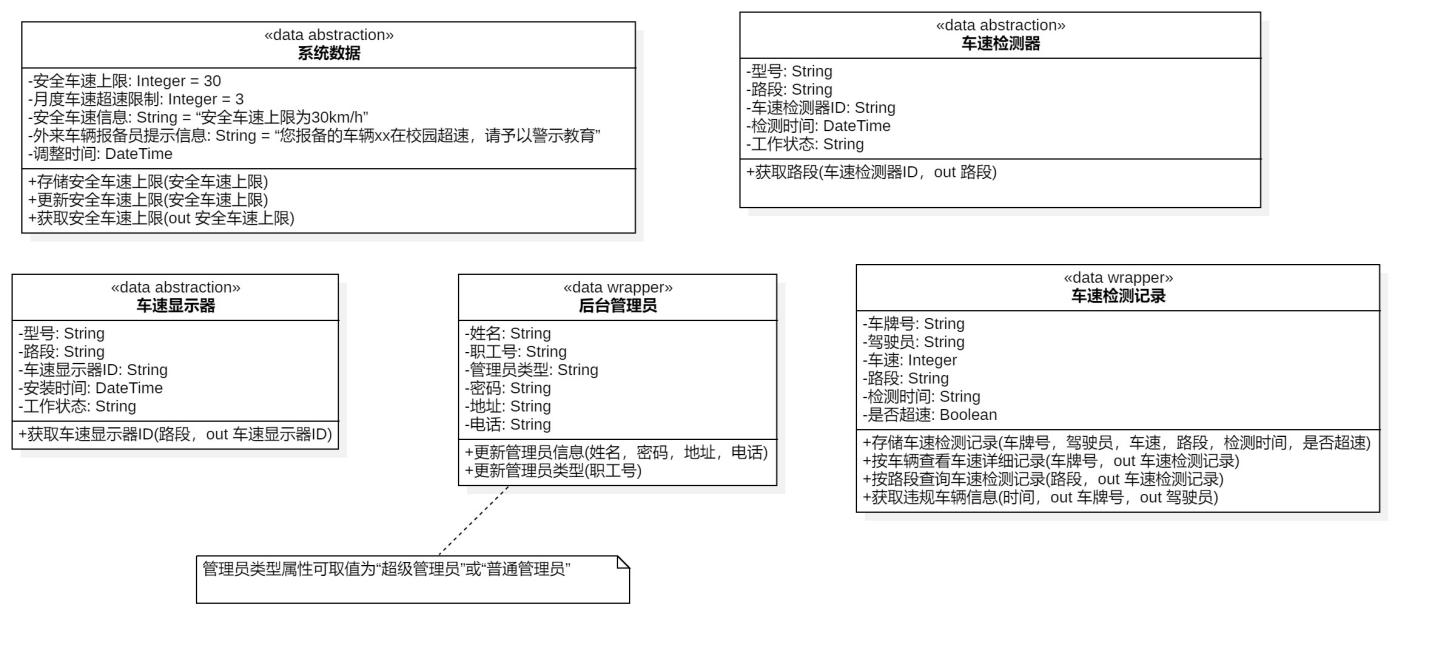
## Design Rationale

无

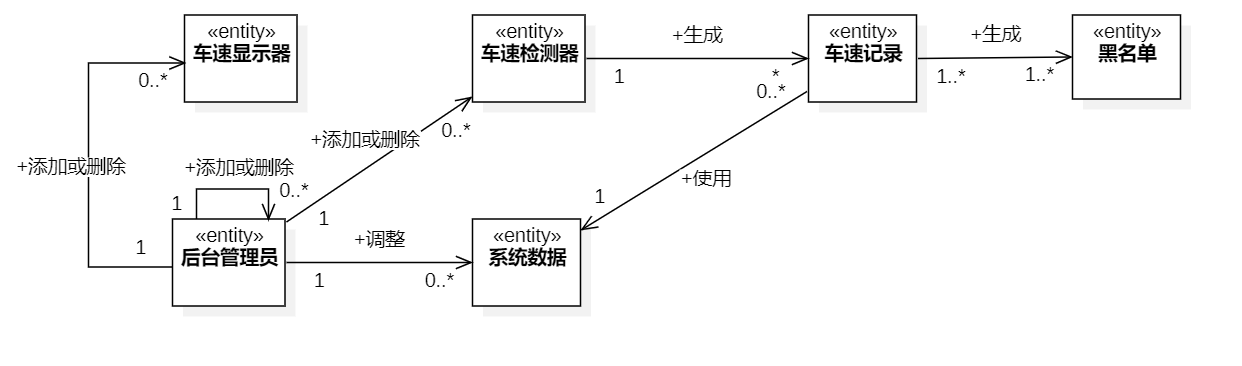
# Data Design

## Data Dictionary

**实体类图**



**实体类关联图**



## Data Storage

**实体类储存方式及其缘由**

系统数据为数据抽象类，因为系统数据信息不需要单独存储在数据库中，而是通过代码直接管理和调用。

后台管理员为数据库包装器类，因为后台管理员需要进行身份验证、权限管理等操作，这类信息需要存储在关系数据库中以便进行用户管理和记录。因此需要将其映射到数据库表。

后台管理员（姓名，职工号，管理员类型，密码，地址，电话）

车速检测记录为数据库包装器类，因为车速检测记录是动态生成的数据，需要存储在数据库中进行历史数据查询、统计分析等。因此需要将其映射到数据库表。

车速检测记录（车牌号，驾驶员，车速，路段，检测时间，是否超速）

车速显示器为数据抽象类，因为车速显示器主要负责显示实时的车速信息，不需要将其状态存储在数据库中。它更多是一个实时控制和显示的设备类。

车速检测器为数据抽象类，因为车速检测器主要负责实时检测车速，并将数据传输给其他系统或存储设备。检测器的配置信息可以作为系统数据的一部分，不需要单独存储。

**关系数据库表映射**

系统数据（安全车速上限，月度超速次数限制，安全车速信息，外来车辆报备员提示信息，调整时间，*后台管理员职工号*）

后台管理员（姓名，职工号，管理员类型，密码，地址，电话）

车速检测记录（车牌号，驾驶员，车速，路段，检测时间，是否超速）

车速显示器（型号，路段，车速显示器ID，安装时间，工作状态）

车速检测器（型号，路段，车速检测器ID，安装时间，工作状态）

# Component Design

**管理员权限变更管理器类详细设计**

**属性**：currentUser

**操作**：

查询普通管理员名单

operation queryRegularAdminList()

begin

if currentUser is not SuperAdmin then

throw UnauthorizedAccessError

end if

regularAdminList = []

for each user in userDatabase

if user.role == "RegularAdmin" then

add user to regularAdminList

end if

end for

return regularAdminList

end

转让管理权限

operation transferAdminRights(targetAccount)

begin

if currentUser is not SuperAdmin then

throw UnauthorizedAccessError

end if

targetUser = findUserByAccount(targetAccount)

if targetUser is not found then

throw UserNotFoundError

end if

if targetUser.role != "RegularAdmin" then

throw InvalidTargetError

end if

transaction begin

currentUser.role = "RegularAdmin"

targetUser.role = "SuperAdmin"

transaction commit

return Success

end

**调整安全车速业务逻辑类详细设计**

**属性**：

maxAllowedSpeed: Real，表示系统允许的最大车速

currentMaxSpeed: Real，当前系统中的最大车速限制

**操作**：

调整安全车速上限

operation adjustMaxSpeed(newMaxSpeed : Integer)

begin

if newMaxSpeed <= 0 then

throw InvalidSpeedException("输入的车速限制必须为正数。")

end if

if newMaxSpeed > maxAllowedSpeed then

throw InvalidSpeedException("输入的车速限制超出系统规定的最大值。")

end if

currentMaxSpeed = newMaxSpeed

end

返回调整结果

operation getAdjustmentResult(success : Boolean) : String

begin

if success then

return "最大车速限制修改成功。"

else

return "最大车速限制修改失败。输入的车速限制无效。"

end if

end

# Human Interface Design

## Overview of User Interface

无

## Screen Images

无

## Screen Objects and Actions

无

# Requirements Matrix

无

# Appendices

无

1. <http://ieeexplore.ieee.org/browse/standards/collection/ieee> [↑](#footnote-ref-0)