城市共享停车管理系统需求规格说明书

1. 概述

1.1 文档目的与范围

本文档旨在从软件工程角度，详细描述城市共享停车管理系统的功能需求和非功能需求。作为系统分析、设计和开发的依据。

1.2 系统定位

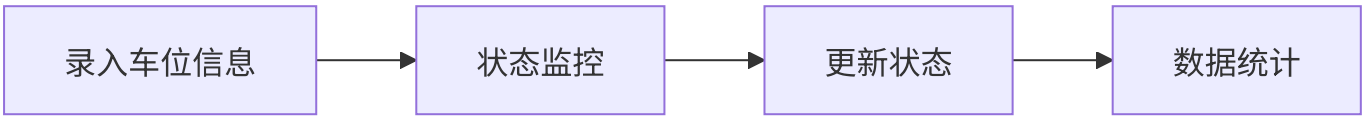
- 系统类型：B/S架构的Web应用系统
- 业务领域：智慧城市 - 停车管理
- 用户规模：城市级应用

2. 需求分析

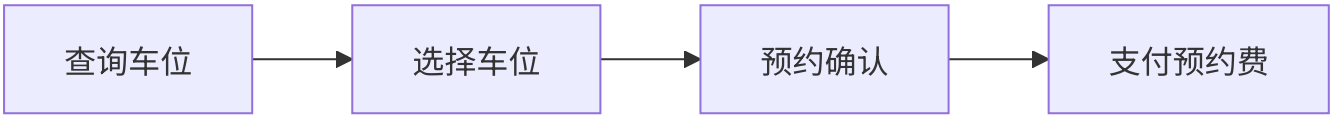
2.1 业务流程分析

2.1.1 核心业务流程

1. 停车位管理流程



2. 车位预约流程



2.1.2 关键用例分析

用例名称	主要参与者	前置条件	后置条件
车位查询	车主用户	用户已登录	展示可用车位

用例名称	主要参与者	前置条件	后置条件
车位预约	车主用户	存在空闲车位	预约成功并扣费
状态更新	管理人员	系统正常运行	车位状态已更新

2.2 功能需求分析

2.2.1 功能分解

```
系统功能树：
├── 用户管理
│   ├── 用户注册
│   ├── 用户认证
│   └── 权限管理
├── 车位管理
│   ├── 信息维护
│   ├── 状态监控
│   └── 统计分析
├── 预约服务
│   ├── 在线预约
│   ├── 状态确认
│   └── 取消处理
└── 支付结算
    ├── 费用计算
    ├── 在线支付
    └── 订单管理
```

2.2.2 功能规格说明

基于原始需求拆解的必要功能条款：

需求条款001：实时信息采集

```
<系统> = 停车管理系统
<功能> = 采集与维护
<对象> = 停车场车位信息
<性能> = 1
<单位> = 分钟
描述：系统应支持停车管理员实时采集和维护各类停车场（居民小区、大学校园、机关大院、公共停车场、路边）的车位信息，包括车位状态、位置等基本信息的录入与更新。
```

需求条款002：车位查询预约

<系统> = 停车管理系统
<功能> = 查询与预约
<对象> = 空置车位
<性能> = 3
<单位> = 秒
描述：系统应支持用户实时查询目的地周边的空置车位信息，并允许用户在30分钟内提前预约空置车位，预约成功后系统自动锁定该车位。

需求条款003：导航支付服务

<系统> = 停车管理系统
<功能> = 导航与支付
<对象> = 预约车位
<性能> = 5
<单位> = 秒
描述：系统应在预约确认后自动对接导航服务(如高德)提供路径指引，并支持从预约确认时间开始到车辆离场的全过程计费与支付服务。

需求条款004：状态监控

<系统> = 停车管理系统
<功能> = 监控
<对象> = 系统状态
<性能> = 实时
<单位> = 分钟
描述：系统应实时监控并同步车位使用状态、预约状态和支付状态，确保数据的准确性和一致性，支持异常情况的及时发现与处理。

2.3 数据需求分析

2.3.1 概念数据模型

1. 核心实体

- 用户：系统的使用者，包括车主和停车场管理员
- 停车场：提供车位的场所，可以是居民区、校园等
- 车位：具体的停车位置
- 预约：用户对车位的预定记录

- 支付记录：预约相关的支付信息

2. 实体关系说明

- 用户与预约：一对多，一个用户可以发起多个预约
- 停车场与车位：一对多，一个停车场包含多个车位
- 车位与预约：一对多，一个车位可以有多个预约（不同时间段）
- 预约与支付记录：一对一，每个预约对应一条支付记录

3. 主要属性描述

- 用户实体属性：
  - 用户ID：唯一标识符
  - 用户名称：实名信息
  - 联系电话：手机号码
  - 车牌号码：绑定车辆信息
- 停车场实体属性：
  - 停车场ID：唯一标识符
  - 场地类型：区分不同类型停车场
  - 地理位置：停车场坐标信息
  - 更新时间：信息更新时间戳
- 车位实体属性：
  - 车位ID：唯一标识符
  - 所属停车场：关联停车场ID
  - 使用状态：空闲/占用/预约
  - 具体位置：车位坐标信息
  - 更新时间：状态更新时间戳
- 预约实体属性：
  - 预约ID：唯一标识符
  - 用户ID：关联用户信息
  - 车位ID：关联车位信息
  - 预约时间：预定的时间段
  - 预约状态：处理状态标识
- 支付记录属性：
  - 支付ID：唯一标识符

- 预约ID：关联预约信息
- 起始时间：计费开始时间
- 结束时间：计费结束时间
- 支付金额：实际支付费用

4. 关系约束条件

- 一个用户同时最多可有3个有效预约
- 一个车位在同一时间段只能被一个预约占用
- 预约必须关联到具体用户和车位
- 每个预约必须生成对应的支付记录

2.3.2 数据字典

基于概念模型的核心数据项：

数据项	数据类型	长度	约束	说明
user_id	VARCHAR	32	主键	用户唯一标识
username	VARCHAR	50	非空	用户姓名
phone	VARCHAR	20	非空,唯一	手机号码
car_plate	VARCHAR	10	非空	车牌号码
parking_id	VARCHAR	32	主键	停车场唯一标识
type	TINYINT	1	非空	1-居民小区,2-大学校园,3-机关大院,4-公共停车场,5-路边车位
space_id	VARCHAR	20	主键	车位唯一标识
status	TINYINT	1	非空	0-空闲,1-占用,2-预约中
reservation_id	VARCHAR	32	主键	预约唯一标识
reserve_time	DATETIME	-	非空	预约生效时间
payment_id	VARCHAR	32	主键	支付记录唯一标识
start_time	DATETIME	-	非空	计费开始时间
end_time	DATETIME	-	可空	计费结束时间
amount	DECIMAL	10,2	非空	支付金额
location	GEOMETRY	-	非空	地理坐标位置
update_time	DATETIME	-	非空	最后更新时间

2.4 质量属性分析

作为一个实时处理、涉及支付交易的城市级停车管理系统，系统必须满足高性能、高可靠性和严格的安全要求。

2.4.1 性能需求

系统的性能直接影响用户体验和停车资源的利用效率，需要在实时性、准确性和业务处理方面达到严格的标准：

1. 实时性要求
- 车位状态采集周期：≤1分钟
- 状态更新延迟：≤1分钟
- 查询响应时间：≤3秒
- 预约处理时间：≤5秒
2. 数据准确性
- 车位状态准确率：≥99.9%
- 位置信息精度：≤5米
- 计费误差：0
3. 业务约束
- 预约提前时间：≤30分钟
- 预约确认时效：≤5分钟
- 导航对接延迟：≤5秒

2.4.2 可靠性需求

考虑到系统服务的持续性要求和数据安全的重要性，系统需要保证高可用性和快速的故障恢复能力：

- 系统可用性：99.9%
- 故障恢复：30分钟内
- 数据备份：实时同步

2.4.3 安全需求

由于系统涉及用户隐私和支付信息，需要建立完善的安全保护机制：

- 身份认证：多因素认证
- 访问控制：基于RBAC
- 数据加密：传输和存储加密

2.5 接口需求分析

2.5.1 外部接口

1. 用户接口
   - Web前端：REST API
   - 移动端：APP API
2. 系统接口
   - 支付系统：异步通知
   - 地图服务：HTTP接口

2.5.2 内部接口

1. 模块间接口
   - 服务调用：RPC
   - 消息通信：消息队列

2.6 约束分析

2.6.1 技术约束

- 开发框架：Spring Cloud
- 数据库：MySQL
- 缓存：Redis

2.6.2 业务约束

- 预约时效：<=30分钟
- 支付期限：<=15分钟
- 取消规则：预约后5分钟内

3. 验证与确认

3.1 需求验证方法

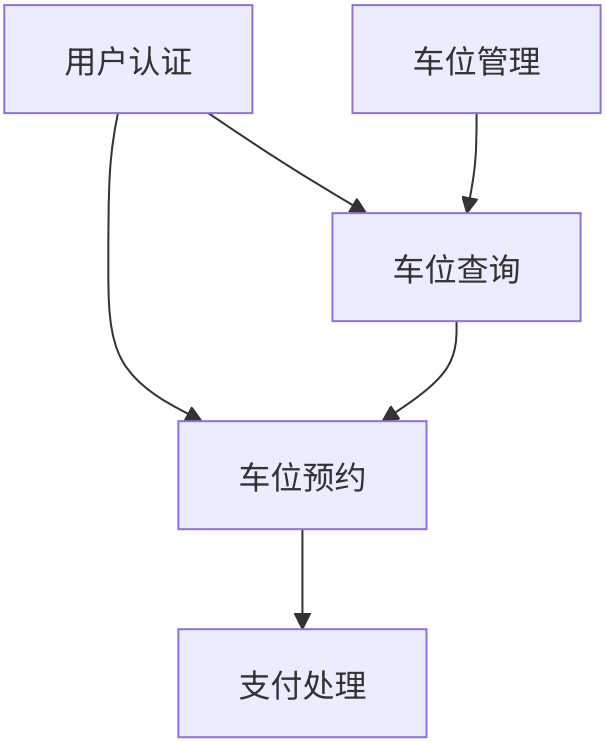
1. 评审会议
2. 原型验证
3. 用例测试

3.2 验收标准

需求类型	验收指标	标准
功能需求	功能完整性	100%
性能需求	响应时间	≤3秒
可靠性	系统可用性	≥99.9%

4. 需求跟踪与分析

4.1 需求依赖分析



4.2 需求优先级分析

需求ID	功能描述	优先级	MoSCoW分类
REQ001	用户注册登录	P0	Must
REQ002	车位状态管理	P0	Must
REQ003	车位查询	P0	Must
REQ004	在线预约	P0	Must
REQ005	支付功能	P1	Should

需求ID	功能描述	优先级	MoSCoW分类
REQ006	数据统计	P2	Could

4.3 需求变更控制

4.3.1 变更流程

1. 变更申请
2. 影响分析
3. 评审决策
4. 变更实施
5. 验证确认

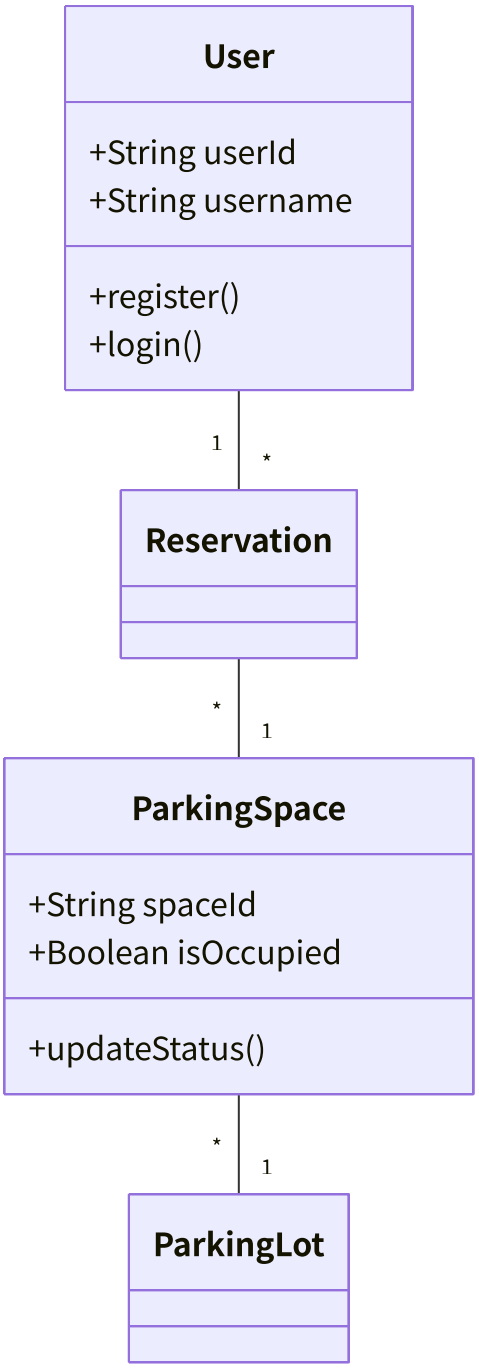
4.3.2 变更评估维度

评估维度	评估指标	评估方法
技术可行性	开发难度	专家评估
工期影响	延期风险	项目经理评估
成本影响	人力成本	成本分析
质量影响	系统稳定性	架构师评估

5. 系统建模分析

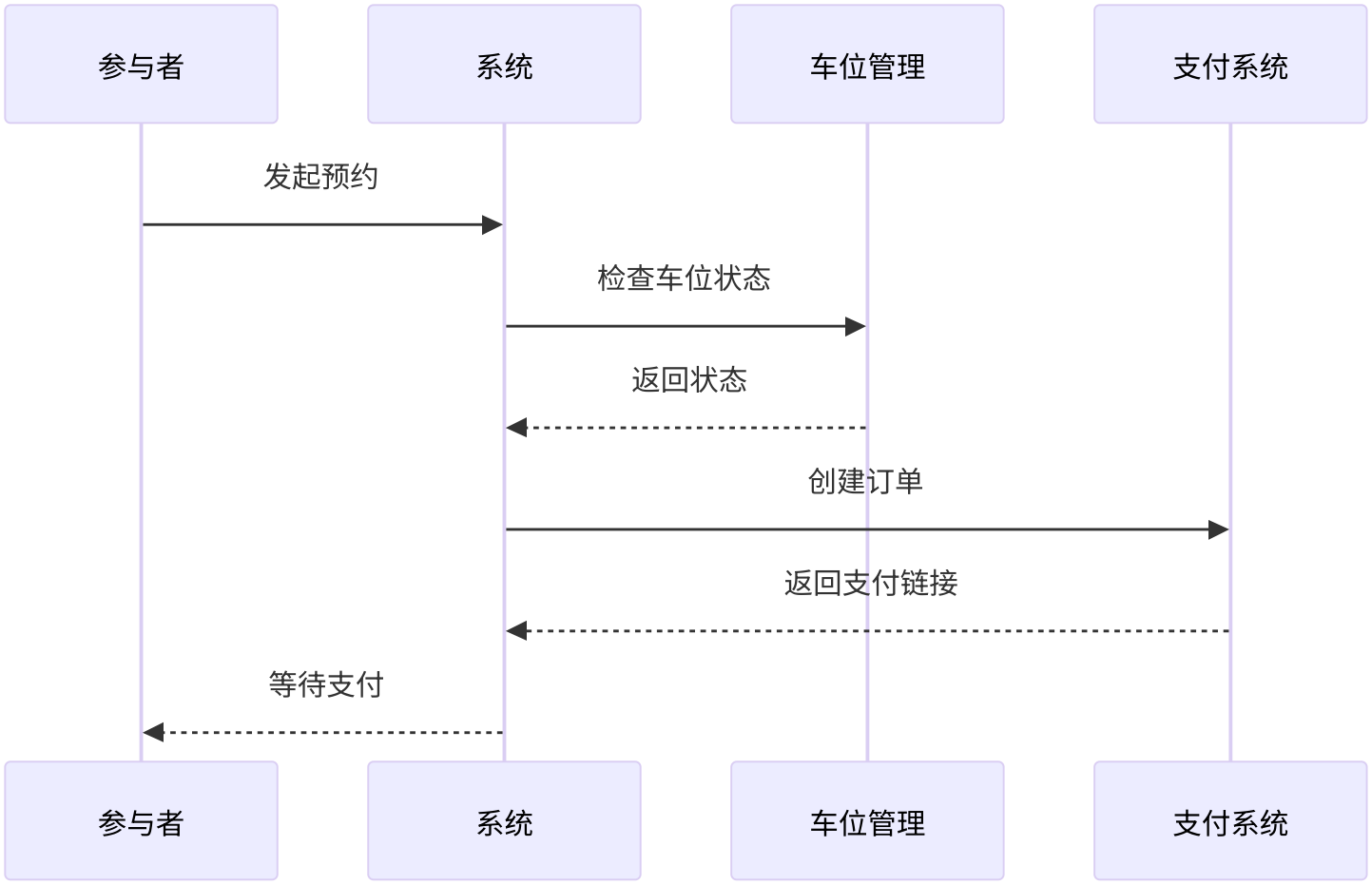
5.1 静态结构模型

系统的静态结构主要包括用户管理、停车场管理、预约管理和支付管理四个核心模块，它们之间通过明确的接口进行交互：



5.2 动态行为模型

系统的核心业务流程是车位预约，从用户发起预约到完成支付的完整交互过程如下：



5.3 模块交互分析

5.3.1 模块耦合度评估

基于系统的可维护性和扩展性要求，对主要模块间的耦合关系进行评估：

模块对	耦合类型	优化建议
用户-预约	数据耦合	保持现状
预约-支付	控制耦合	考虑解耦
车位-统计	内容耦合	需要重构

5.3.2 接口复杂度分析

从接口设计的简洁性和可维护性角度，评估主要接口的复杂度：

接口名称	参数数量	复杂度	建议
创建预约	5	中	简化参数
更新状态	3	低	保持现状

接口名称	参数数量	复杂度	建议
统计查询	8	高	拆分接口

6. 架构约束与决策

6.1 技术架构约束

考虑系统的规模和复杂度，采用分层架构和微服务架构相结合的方式：

1. 分层架构
   - 表现层：处理用户交互
   - 业务层：实现核心逻辑
   - 持久层：数据访问封装
   - 数据层：数据存储管理
2. 微服务架构
   - 服务独立部署：确保模块隔离
   - 服务间通信：采用消息机制
   - 服务治理：统一监控管理

6.2 关键技术决策

基于系统的性能和可靠性要求，对关键技术选型进行决策：

决策点	方案选择	决策理由
并发控制	乐观锁	冲突率低
缓存策略	多级缓存	性能要求
消息队列	Kafka	可靠性高