文档修订记录

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 版本 | 日期 | AMD | 修订者 | 说明 |
| 1.0 | 2018-05-01 | A | 曾星伟 | 版本1.0建立 |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |

（A-添加，M-修改，D-删除）

目录

[第1章 项目意义 1](#_Toc513406050)

[1.1 项目概述 1](#_Toc513406051)

[1.2 需求要点 1](#_Toc513406052)

[1.3 设计目标 2](#_Toc513406053)

[1.4 参考资料 2](#_Toc513406054)

[1.5 简写名词表 2](#_Toc513406055)

[第2章 总体架构 3](#_Toc513406056)

[2.1 功能架构 4](#_Toc513406059)

[2.2 工作流程 4](#_Toc513406060)

[2.3 运行环境 7](#_Toc513406061)

[第3章 前端体系 10](#_Toc513406062)

[3.1 共享单车系统（前端）架构 10](#_Toc513406063)

[3.2 共享单车主要工作流 11](#_Toc513406064)

[3.3 共享单车与后端服务器的数据接口 17](#_Toc513406065)

[第4章 后端架构 18](#_Toc513406066)

[4.1 软件架构 18](#_Toc513406067)

[4.2 应用层 19](#_Toc513406068)

[4.3 计算层 20](#_Toc513406069)

[4.4 存储层 20](#_Toc513406070)

[第5章 方案可行性 21](#_Toc513406071)

[5.1 技术成熟度 21](#_Toc513406073)

[5.2 负载能力 21](#_Toc513406074)

[5.3 安全可靠性 23](#_Toc513406075)

# 项目意义

## 项目概述

共享单车是指企业在校园、地铁站点、公交站点、居民区、商业区、公共服务区等提供自行车单车共享服务，是一种分时租赁模式。共享单车是一种新型环保共享经济。

目前，市场上最火的共享单车，如摩拜，OFO、小蓝单车等均是无桩、无网点、随地停放、靠押金来盈利。我们旨在开发一种新型智能出行平台，它分片区、有网点、有车桩、电动、可临时停放等。这种智能出行平台集便于管理、速度快、轻便小巧、价格便宜等诸多优点于一身，既符合绿色出行的初衷，又不会给公共交通带来不便。

## 需求要点

此智能共享单车出行平台目前是一个共享单车的信息管理平台，共分为8个模块，分别是：用户管理、基础信息管理、运行管理、营收管理、维护管理、客服管理、版本管理和消息推送。系统模块化清晰，静态数据和动态数据相分离。根据用户的权限，显示不同的模块，管理更加简洁方便。将平台的运行、营收和维护独立出来，更加方便用户管理这个系统。

根据不同功能划分，本系统主要定义了以下角色：

表1-1 功能划分图

|  |  |
| --- | --- |
| 角色 | 描述 |
| 系统管理员 | 拥有系统中所有管理权限，采用Web方式访问系统。 |
| 总运营商 | 总运营商，不具备系统设置权限，采用Web方式访问系统 |
| 运营商 | 运营商管理员，是某个或某几个片区的管理者，只能看到自己片区的信息，管理上需要按片区进行分配，采用Web方式访问系统 |
| 片区管理员 | 片区管理员，只能看到自己片区的相关信息，采用Web和App方式访问系统。 |
| 回收人员 | 回收桩外和故障车辆的人员，采用APP方式访问系统 |
| 保洁人员 | 负责指定网点的保洁工作的人员，采用APP方式访问系统 |
| 财务人员 | 可以使用平台营收服务的人员，采用Web方式访问系统 |
| 客服人员 | 可以使用平台客服服务的人员，采用Web方式访问系统 |
| 合作商 | 可以使用平台的营收服务的人员，仅能访问安装了合作商设备单车的营收报表。采用Web方式访问系统 |
| 普通用户 | 单车的使用者，采用APP方式访问系统 |

## 设计目标

* 构建一个用户使用更简单方便、实惠、人性化的智能出行系统；
* 加入车桩、网点的概念，使共享单车更便于管理和维护；
* 通过多数据库结合，实现静态数据和动态数据相分离；
* 划分多个角色将平台的运行、营收、维护独立出来，使投资方、合作商、运营商的营收更加清晰明确。

## 参考资料

《智能共享单车运营管理平台用户需求规格说明书v3.0》

## 简写名词表

表1-2 简写名词表

|  |  |
| --- | --- |
| **名词** | **简写** |
| 智能共享单车运营管理平台 | 项目一期 |
|  |  |
|  |  |
|  |  |

# 总体架构



智能共享单车运营管理平台的总体架构如图2.1所示，包括三个单元：前端应用、后台ERP系统、数据库。

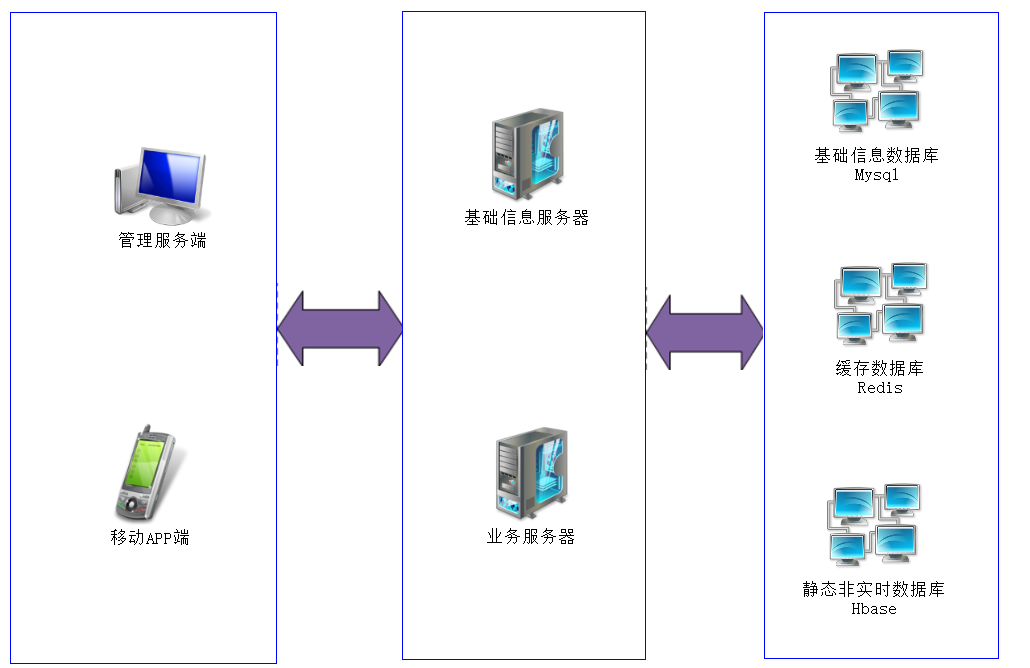


图2.1 智能共享单车运营管理平台架构

智能共享单车运营管理平台应用主要分为普通客户端和管理服务端。

管理服务端主要是Web端应用和服务，主要包括八大模块：基础信息管理端、用户管理、运行管理、营收管理、维护管理、客服管理、版本管理、消息管理。其中，主要使用Web服务端的角色包含系统管理员、运营商、片区管理员、财务人员和客服人员。主要具体功能是进行一些基础信息的维护，以及营收报表等的管理查看。

移动App端，主要提供给普通的借车用户，包含借车、还车、预约车以及临时锁车等功能。

基础信息服务主要对一些实时性不高的、基础的数据进行处理，比如用户信息、车辆基础信息等。

业务服务器主要对复杂功能逻辑、复杂业务流程、高并发的数据进行处理。如借车、还车、预约车、临时锁车等复杂、并发性高的业务逻辑流程。

大数据分析中心主要进行数据的维护，营收报表的分析与制作，车历史信息和实时信息等非即时的、静态数据的存储和维护。

## 功能架构

智能共享单车项目的核心业务包括两大功能：管理服务端和普通客户端。其中管理服务端功能包括：基础信息管理端、用户管理、运行管理、营收管理、维护管理、客服管理、版本管理、消息管理。基础信息管理包括运营商、供应商、合作商、片区、网点、车辆等信息的管理；运行管理主要是车辆状态、网点状态的管理维护；营收管理是指营收报表（运营商、合作商、供应商）、车辆报表、用户报表；维护管理包含故障、报警、维修等管理维护。普通客户端功能包括：借车、还车、预约车、临时锁车。具体功能结构如下图所示：

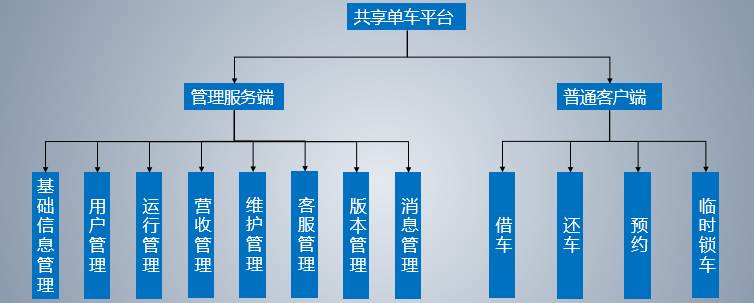


图2.2 共享单车平台功能架构

## 工作流程

**车辆状态转换图**：



图2.3 车辆状态转换图

转换触发条件

1. 当系统新建车辆信息时，车辆状态为待上线，线下人员投放车辆，车辆入桩。系统检测车辆信息以后状态改为桩内？。
2. 用户借车，车辆状态由在桩改为运行中。
3. 用户将使用中的车推入桩内还车，状态由运行中改为在桩。
4. 用户在使用过程中车辆发生故障，用户上报故障，车辆状态改为故障待确认。
5. 片区管理员将桩外车入桩，状态由桩外改为在桩。
6. 片区管理员在判别故障待确认列表中故障车，发现上报属实需要回收，则选择回收，车辆状态由故障待确认改为故障。
7. 片区管理员在判别故障待确认列表中故障车，发现上报不属实不需要回收，则选择不回收，车辆状态由故障待确认改为桩外。
8. 回收人员线下回收在app端进行回收操作，后台状态改为维修中。
9. 当回收人员将维修完成的车辆运输到网点并入桩，车辆状态由维修改为在桩。
10. 当片区管理员到达故障待确认的车辆定位位置以后并未发现车辆，则上报丢失。车辆状态由桩外改为丢失。
11. 当回收人员到达故障列表中的车辆定位位置以后并未发现车辆，则上报丢失。车辆状态由桩外改为丢失。
12. 片区管理员维保时上报故障。状态由在桩改为故障。
13. 片区管理员管理桩外车辆时发现故障。
14. 片区管理员管理桩外车辆时发现故障，确认丢失以后在APP端or Web端？添加车损信息。
15. 片区管理员管理桩外车辆时发现故障，确认车辆报废以后在APP端or Web端？添加车损信息。
16. 片区管理员管理桩外车辆时发现故障，系统检测到上报丢失车辆入桩信息，状态改为桩内。

**用户状态转换图**：



图2.4 用户状态转换图

触发条件

1. 用户默认均是未注册状态，通过注册登录以后改为一般状态。
2. 用户登录满足借车条件以后，在桩上借车使用，状态改为使用中。
3. 用户将使用车辆进行还车操作。状态改为正常状态。
4. 用户使用行为恶劣等原因，后台将用户账号冻结。用户状态改为冻结。
5. 冻结用户满足一定条件以后后台将用户解冻。状态改为正常

**桩状态转换图：**



图2.5 桩状态转换图

状态转换触发条件：

1. 当线下桩修建以后，系统录入桩的信息。而线下投放使用前，桩的状态为待开放。系统检测到控制柜上传了桩的信息以后桩的状态改为空桩。
2. 当车入桩后，桩的状态改为使用中。
3. 当用户借车以后，对应桩状态改为空桩。
4. 车在桩上而用户借车时，桩未反应。用户上报故障，片区管理员确认故障。此时桩状态改为故障。
5. 故障桩线下维修完成以后，车为入桩。此时桩的状态为空桩。
6. 推车入桩。桩未上锁等原因，片区管理员确认以后上报故障，桩状态为故障。或者片区管理员进行维保时，桩状态为故障。
7. 当网点关闭时，桩状态变为关闭。

## 运行环境

**软件环境需求**

表 2‑1软件环境需求

|  |  |  |
| --- | --- | --- |
| **项目** | **最低配置** | **建议配置** |
| 操作系统 | Linux | Ubuntu16.04 LTS |
| 数据库 | Mysql5.5、hbase1.2.5、redis3.2 | Mysql5.7、Hbase1.2.5、redis3.2 |
| 中间件 | Tomcat7，emqtt2.1 | TOMCAT8,emqtt2.1 |
| 集群 | Hadoop2.5 | Hadoop2.7 |
| JDK | sun1.7 | sun1.7 |
| OFFICE | EXCEL2000~2003 | EXCEL2010（导出为xls导入需支持xlsx） |
| 客户端浏览器 | Firefox48（推荐浏览器IE） | Firefox53（推荐浏览器IE） |
| 通信协议 | Mqtt |  |

**硬件环境需求**

表 2‑2 硬件环境需求

|  |  |  |
| --- | --- | --- |
| **最低配置-数据库服务器** | | |
| **类别** | **要求** | **备注** |
| CPU | 最少两个3.0GHZ以上CPU |  |
| 内存 | 最少8GB |  |
| 主机硬盘 | 最少240GB SSD，1T 高可用机械硬盘 |  |
| 网络 | 1000M局域网 |  |
| 操作系统 | Linux |  |
| 数据库软件 | Mysql5.7 |  |
| 外部网络 | 需要访问INTERNET，独占下行网速不小于20M/S，独占上行网速不小于5M/S |  |
| 数据库服务器个数 | 最少2个 | 主从复制读写分离，一个主写， 1个从读节点 |
| **最低配置-应用服务器** | | |
| **类别** | **要求** | **备注** |
| CPU | 最少四个3.0GHZ以上CPU |  |
| 内存 | 最少32GB |  |
| 主机硬盘 | 最少512GB SSD，1T的高可用机械硬盘 |  |
| 网络 | 1000M局域网 |  |
| 操作系统 | Linux |  |
| 数据库软件 | Redis |  |
| 应用服务器软件 | TOMCAT8 |  |
| 外部网络 | 需要访问INTERNET，独占网速不小于40M/S，独占上行网速不小于5M/S |  |
| 应用服务器数量 | 2个 |  |
| **最低配置-Emqtt中间件服务器** | | |
| **类别** | **要求** | **备注** |
| CPU | 最少四个4.0GHZ以上CPU |  |
| 内存 | 最少32GB |  |
| 主机硬盘 | 最少240GB SSD，1T的高可用机械硬盘 |  |
| 网络 | 1000M局域网 |  |
| 操作系统 | Linux |  |
| 应用服务器软件 | Emqtt2.1 |  |
| 外部网络 | 需要访问INTERNET，独占网速不小于40M/S，独占上行网速不小于5M/S |  |
| 应用服务器数量 | 2个 |  |
| **最低配置-Hadoop集群** | | |
| **类别** | **要求** | **备注** |
| CPU | 最少2个3.0GHZ以上CPU |  |
| 内存 | 最少32GB |  |
| 主机硬盘 | 最少512GB SSD，15T的高可用机械硬盘 | 只能保存半年数据 |
| 网络 | 1000M局域网 |  |
| 操作系统 | Linux |  |
| 应用服务器软件 | Hadoop |  |
| 数据库软件 | Hbase |  |
| 外部网络 | 需要访问INTERNET，独占网速不小于40M/S，独占上行网速不小于5M/S |  |
| 应用服务器数量 | 3个 | 集群一个主节点Master，2个slave。若采用负载均衡。需要数量加倍 |

上述服务器数量为最低配置，未采用负载均衡，为了防止宕机与停止服务，需要至少双机热备。

**网络环境需求**

表 2‑3网络环境需求

|  |  |  |
| --- | --- | --- |
| 类别 | 最低配置 | 建议配置 |
| 下行速度 | 40M/S | 100M/S |
| 上行速度 | 4M/S | 10M/S |

# 前端体系

## 共享单车系统（前端）架构

共享单车系统前端主要分为Web端后台管理系统和APP端客户系统（暂不考虑APP管理端）。Web端后台管理平台主要分为八大模块：用户管理、基础信息管理、运行管理、营收管理、维护管理、客服管理、版本管理、消息推送管理。

用户管理：普通用户管理主要是显示App用户的基础信息。包含：显示所有会员信息，若有条件显示满足条件的用户信息；显示用户借车的信息；将用户状态由借车，变为还车状态；用户账户冻结请求。

基础信息管理：管理运营商的基础信息。包含：显示运营商信息列表；弹出添加页面，当用户输入添加的运营商信息后，失败返回错误信息，成功返回到列表；弹出编辑页面显示可编辑的项，提交编辑，失败返回错误信息，成功返回列表页；选中的删除项，若删除失败，返回错误信息，删除成功返回列表页；显示选中行的详情页。

运行管理：管理车辆的运行中的信息。若账号等级高于片区则提供分区筛选功能。主要功能为：显示车辆目前状态信息，若有条件显示符合条件信息；弹出地图，显示当前车辆在地图上面的位置；当普通用户状态改变为结束时车的状态随之改变；显示所选片区状态信息；显示更改结果。

营收管理：查看所有运营商的营收状态。主要显示当年的所有运营商营收状况，若有条件显示符合条件的信息。

维护管理：查看主控柜、车子、服务器的所有报警情况，如权限高于片区则可进行片区筛选。报警可能由车子、服务器或主控柜产生。具体输出为：显示所有主控柜、车子、服务器报警情况，若有条件显示符合条件的信息；显示分区所有主控柜、车子、服务器报警情况。

客服管理：管理反馈意见。显示所有反馈意见，若有条件显示符合条件的信息；弹出添加页面，填写意见相关信息，成功返回列表，失败则返回错误信息；弹出修改页面，填写修改信息，成功返回列表，失败则返回错误信息；意见信息转移给财务。

版本管理：管理平台上发布的app版本。显示APP版本信息列表，若有条件，根据条件显示APP列表；弹出添加页面，添加app信息，返回添加成功信息，及显示最新app列表；显示选中的app行的详情列。

消息推送管理：管理平台上下发消息到App上，其中推送包括空桩、满桩、人工代还车以及手动触发的推送。显示推送信息列表，若有条件，根据条件显示信息列表；弹出添加页面，添加推送信息内容，添加成功信息或失败信息；删除未发送成功的消息；显示消息详情页面。

普通用户APP业务功能主要包括：基本业务功能、个人信息业务功能、客服业务功能、搜索业务功能、信息设置、消息管理六大功能模块。

基本业务功能是目前该app最核心的功能，用户在app上可以查看附近网点，到达网点以后用户可以直接使用app进行借车使用。若未注册则先注册再使用。包含四个主要业务流程：借车业务、还车业务、预约车业务、一键寻车。

个人业务功能：用户可通过注册登录模块进行注册以及登录，包含：注册登录、个人基本信息查看、行程信息管理、个人钱包管理。

客服业务功能：用户遇到单车的任何故障问题均可以通过故障上报功能上报。回收人员也可以通过该功能直接上报。包括：故障上报、意见反馈、人工服务三个子功能。

搜索业务功能：用户通过搜索功能可以查询目的地附近的网点，并路径规划以及动态导航。

信息设置：用户可以通过设置清理本地缓存、查看app官方基本信息以及查看基本协议。

消息管理：查看官方推送给用户的消息，可以查看历史消息，可以删除消息。

## 共享单车主要工作流

1. 用户管理（普通用户管理）：

业务流程图如下图所示：



图3-1 用户管理业务流程图

操作流程：

1. 点击用户管理，点击普通用户管理；分页显示所有会员信息；在搜索框输入用户信息，显示满足条件的用户信息。
2. 点击某行的用户详情，弹出页面，显示用户的用车历史记录。
3. 点击借车状态图标，弹出借车状态页面，将借车状态改变为否（结束行程）。
4. 用户账户冻结。
5. 基础信息管理（运营商信息管理）：

业务流程图如下图所示：



图3-2 基础信息管理业务流程图

操作流程：

1. 显示运营商列表信息，若有条件输入，显示满足条件的信息列表
2. 填写添加信息内容，点击保存
3. 点击每行对应的编辑按键，弹出页面，修改需要编辑的内容，点击保存
4. 选中要删除的行，点击删除。或者点击每行对应的删除按钮
5. 点击每行对应的详情键，弹出详情页。
6. 运行管理（车辆状态管理）：

业务流程图如下图所示：



图3-3 运行管理业务流程图

操作流程：

1. 显示车辆目前状态信息，若有条件显示符合条件信息。
2. 弹出地图，显示当前车辆在地图上面的位置。
3. 当普通用户状态改变为结束时车的状态随之改变。
4. 显示所选片区状态信息。
5. 显示更改结果。
6. 营收管理（运营商报表）

业务流程图如下图所示：



图3-4 营收管理业务流程图

操作流程：

1. 登录系统，点击营收管理，再在弹出的下级目录点击运营商报表。
2. 输入条件，进行条件查询。
3. 维护管理（报警记录）：

业务流程图如下图所示：



图3-5 维护管理业务流程图

操作流程：

1. 查看报警记录。
2. 输入条件，进行条件查询
3. 选择分类进行分类查看。
4. 选择报警来源进行来源分类查看
5. 选择分区进行分区报警记录查看
6. 借车业务：

业务流程图如下图所示：



图3-6 借车业务业务流程图

操作流程：

1. 用户到达网点，按下接车按钮，如果提示灯是红色，桩提示该车故障。用户请选择其他桩借车。
2. 如果按钮为（黄、绿）颜色，系统整合车辆信息下发二维码，桩上屏幕显示二维码，用户打开app点击扫码用车，进入扫码界面，扫码后判断是否已登录，未登录则跳转至登录界面。用户登录以后判断用户是否满足借车条件（账户余额充足），若不满足提示对应未满足的条件。
3. 登陆以后app判断该车是否是预约车，如果该车是预约车，判断该用户是否是在该网点预约的用户，不是则提示用户该车已被预约。如果不是预约车或则该用户是预约车对应用户则直接扫描成功。
4. App返回扫码结果给后台，若果符合借车条件则扫码成功后桩解锁，用户借车成功，app进入计费界面。
5. 还车业务：

业务流程图如下图所示：



图3-7 还车业务业务流程图

操作流程：

1. 借车用户到达网点，将车推入空闲车桩，完成还车。系统提示行程收费单。
2. 用户到达网点以后，无空闲车桩，将车推进临时还车点，app上申请还车，完成还车，系统提示行程收费单。
3. 由于个人原因，用户需要就地还车。用户在合适位置锁车以后app上申请就地还车，接受就地还车费用后完成还车，系统提示行程收费单。
4. 预约车业务：

业务流程图如下图所示：



图3-8 预约车业务业务流程图

操作流程：

1. 用户在地图中选择网点，点击预约，系统判断是否是登录用户，如果没有登录就跳转登录页面。判断用户是否符合使用条件（押金、余额）。
2. 系统判断用户的预约是当天前4次预约，则免费预约，如是第4次之后则提示收费预约。
3. 预约成功开始计时，结束前三分钟提醒用户是否继续预约，若用户点击继续预约则开始计时收费；若取消，则预约时间到期后自动结束预约。
4. 如果用户没有任何操作，超过预约时间自动取消预约。

## 共享单车与后端服务器的数据接口

详见接口文档。

# 后端架构

## 软件架构

本项目整体的后端架构分为四个层次：访问层，应用层、计算层、存储层。



图 4‑1 软件架构图

**访问层：**

APP用户（普通APP租车用户和APP管理用户）以及WEB管理用户进行数据访问。单车、控制柜与服务器进行交互。

**应用层：**

单车与控制柜进行消息的处理（传向计算层进行逻辑计算或下发指令给单车和控制柜）。移动App和Web端对访问层的访问请求等进行接口支撑。

**计算层：**

处理复杂业务流程如借车、还车、预约车、临时锁车等；进行数据的处理、计算；将高实时性、常用数据等状态信息存入缓存。Web端和App端通过Thrift云服务与计算层进行远程数据调用。

**存储层：**

将计算层计算、解析出的非实时的静态的规模大的数据存入HBase数据库。MySQL数据库用于存储基础类信息。Redis用于存储常用状态信息。各集群通过Zookeeper来进行管理。

**具体流程描述：**

1. 共享单车和控制柜通过MQTT协议与服务器进行通讯，车、桩将状态等信息发送给EMQTT集群服务器，EMQTT将控制信息发送到车、桩；
2. EQMTT集群将消息命令解析后存入KAFKA集群；
3. STORM集群按不同topic负责处理KAFKA队列里的消息；
4. STORM集群将处理后的消息或存入HBASE，或存入REDIS，或下发控制命令给车、桩；
5. WEB端和APP通过Tomcat集群访问服务器，进行正常业务操作；
6. Tomcat集群通过Nginx进行负载均衡，通过RPC对STORM、HBASE、REDIS进行远程服务调用。

## 应用层

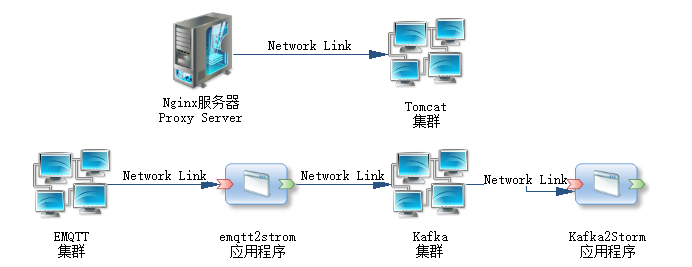


图 4‑2 应用层

Web端和App端的访问请求经NGINX服务器、Tomcat集群进行数据访问。

车和桩的请求通过MQTT物联网协议到EMQTT集群，EMQTT集群进行指令解析后按不同topic存入KAFKA队列中，Kafka队列中的数据经由Kafka2Storm应用程序送入计算层。

## 计算层

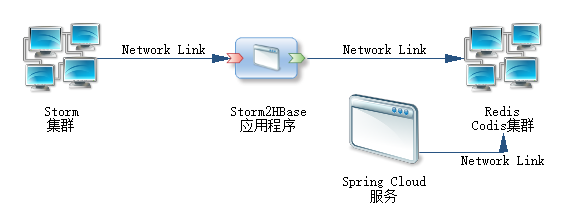


图 4‑3 计算层

应用层的数据到达计算层后通过STORM集群进行数据流处理，然后将需要存储的数据通过Storm2HBase应用程序按需要分别将数据存入Redis或者HBase数据库

## 存储层

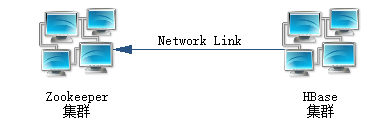


图 4‑4 存储层

Mysql和Redis存储的数据都较为简单，并且具有结构化的特点。相反，HBase数据库不是传统的面向列的数据库，它是Nosql型数据库，存储麻烦，需要对其RowKey进行特殊处理和设计。存入Hbase的数据主要有车的行程记录、车的实时位置信息、车的消费记录等。因为这些数据规模大，若每辆车平均每天产生3条消费数据，10万辆车一天就要产生30万条消费数据，Mysql数据库对于如此大规模的数据量维护、管理麻烦，查询效率低。

# 方案可行性



## 技术成熟度

平台技术设计主要采用了分布式流式大数据处理架构Storm，并采用了MQTT—KAFKA—STORM—HBASE这一套分布式物联网数据处理框架。采用的技术、框架多且复杂。

表5-1给出了各项主流技术和框架的技术成熟度等相关信息，由表中数据可看出，这一套架构所涉及的各项技术都较为成熟且应用广泛。技术风险在一个可控的范围。

表 5‑1 开发技术成熟度

|  |  |  |
| --- | --- | --- |
| **技术名称** | **发明年代** | **技术成熟度** |
| MQTT | 1999年 | 90% |
| KAFKA | 2010年 | 80% |
| STORM | 2011年 | 70% |
| HADOOP+HBASE | 2007年 | 85% |
| REDIS | 2013年 | 70% |

## 负载能力

**Codis集群测试：**

通过codis-proxy代理服务访问,由codis-config配置redis节点，在dashboard 可以操作各节点并查看节点状态。目前尝试过使用redis3.0.9加入节点group失败，使用codis-server加入节点group成功可用。各节点数据同步测试成功，写入master节点的数据会自动同步到slave节点。Group下节点损坏不影响proxy代理的使用。代理服务访问的模式下，value长度大于888B会导致set性能大幅下降。

**Kafka测试：**

简介：kafka是一种高吞吐量的分布式发布订阅消息系统，它可以处理消费者规模的网站中的所有动作流数据。

测试条件：kafka集群。两台虚拟机，在同一物理机上。

测试数据：1到100,0000。

测试结果：kafka 性能超强。经测试100万条简单数据，4.75秒发送完毕。

**Storm测试：**

简介：Storm是一种实时的分布式计算系统。

测试条件：storm集群。五台虚拟机，分别分布在二台物理机上。开启5个Worker，每个Worker最大并发设为200。

测试数据：kafka发送的数据100万条。乘与2。并打印在日志。

测试结果：经简单测算，处理100万条简单数据用时24.68秒。

结论：如多增加物理机台数，性能会更好。增加worker数量和增加并发会提高性能。目前这些服务器开大了会太卡。

**MQTT测试：**

服务器信息：

表 5‑2 服务器信息

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 服务器 |  | 系统 | CPU | 内存 | 备注 |
| Emqtt服务器1 |  | Ubuntu14.0.4 | i5-4460 | 8G |  |
| Emqtt服务器2 |  | CentOs7 | i5-4460 | 4G |  |

测试概述

本次测试使用单台机器做测试，利用多台机器发起mqtt连接请求，对上述两台服务器进行mqtt连接数进行压力测试。

测试结果

服务器1：能够支持的稳定mqtt并发数为5w，当并发量发到5w时，系统内存使用量为4G，系统不卡顿，但是新的连接请求到来时无法建立连接，并发量只能维持在5w

服务器2：能够支持稳定的mqtt并发数为5w，当并发量达到5w是，系统内存使用量接近4G，系统非常卡顿，新的连接到来也无法建立，并发量维持在5w

结果分析：

服务器1是Ubuntu系统，官方文档使用的是centos系统，可能在配置参数方面存在问题，导致连接数不能增加

服务器2内存太小，使得可以接收的连接数较少

## 安全可靠性

安全策略属于网络信息安全的上层建筑领域，是网络信息安全的灵魂和核心。安全策略为保证信息基础的安全性提供了一个框架，提供了管理网络安全性的方法，规定了各部门要遵守的规范及应负的责任，使得信息网络系统的安全有了切实的依据。在本系统中，可采用如下措施来确保本系统的安全可靠性：

1. 遵照业界通信准则，完全遵循国际上有关数据安全标准。
2. 结合用户实际的情况，尽可能采用当前先进的数据安全技术和产品，并确保系统达到所设计的安全强度。
3. 登陆加密采用SSL加密处理。
4. 用户信息采用MD5加密；用户登录采用验证码。
5. 终端与服务器端的通信采用加密方式，选用对称加密算法。