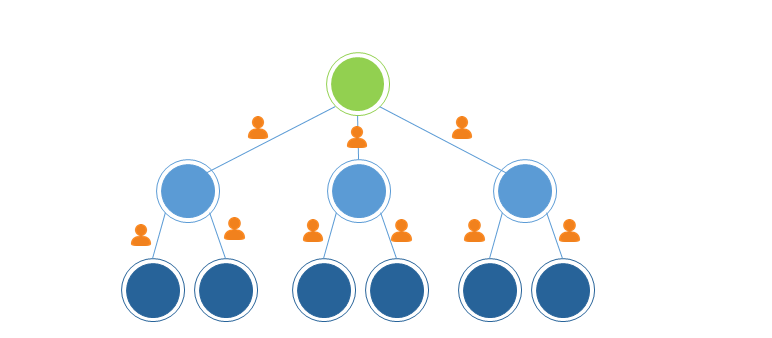
PPM系统架构设计

# ****文档基本信息****

|  |  |  |  |
| --- | --- | --- | --- |
| **日期** | **版本** | 修订内容 | 作者 |
| **2020.02.25** | V1.0 | 基础版本 | liming |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

# 整体架构



用户分为：管理员、普通用户，用户与组关联，管理员与顶级组关联。

组可以一级一级的往下发展，上级组管理下级组的用户信息、终端信息以及终端监控信息。



1、用户通过PPM新增终端信息，上传应用包，设置终端监控配置，查询终端监控信息。

2、终端通过Pxretailer获取终端的一些配置信息，下载应用包，上传终端运行数据。

# 技术框架

## **技术分层**

系统的技术分层情况如下图所示：



### **界面层**

用于显示数据和接收用户输入的数据，对数据做简单的处理，为用户提供一种交互式操作的界面。前端采JSP，js，JQuery，bootstrap，Ajax来实现页面的渲染及与后台的通信。

### [数据访问层](https://baike.baidu.com/item/%E6%95%B0%E6%8D%AE%E8%AE%BF%E9%97%AE%E5%B1%82)

主要是对非原始数据（数据库或者文本文件等存放数据的形式）的操作层，而不是指原始数据，也就是说，是对数据库的操作，而不是数据，具体为业务逻辑层或表示层提供数据服务。

系统采用hibernate框架来实现数据访问层数据的读写，数据库连接池采用ComboPooledDataSource。Hibernate是一个[开放源代码](https://baike.baidu.com/item/%E5%BC%80%E6%94%BE%E6%BA%90%E4%BB%A3%E7%A0%81/114160)的对象关系映射框架，它对[JDBC](https://baike.baidu.com/item/JDBC/485214)进行了非常轻量级的对象封装，它将POJO与数据库表建立映射关系，是一个全自动的orm框架，hibernate可以自动生成SQL语句，自动执行，使得Java程序员可以随心所欲的使用对象编程思维来操纵数据库。

### 业务逻辑层

主要是针对具体的问题的操作，也可以理解成对数据层的操作，对数据业务逻辑处理，它的关注点主要集中在业务规则的制定、业务流程的实现等与业务需求有关的系统设计。

PPM后台采用Spring+SpringMVC框架来简化业务逻辑层的实现。SpringMVC提供清晰的角色划分：控制器（controller）、验证器（validator）、 命令对象（command object）、表单对象（formobject）、模型对象（model object）、 Servlet分发器（DispatcherServlet）、处理器映射（handler mapping）、视图解析器（view resolver）等。可以通过简单的配置来实现数据的校验、处理、传输。Spring和SpringMVC一起使用，可以方便、高效的管理对象，处理事务及日志。IoT后台为前端提供数据操作接口，为了方便和前端的交互系统使用swagger自动生成API结构，给前后端开发以及测试人员提供了良好的接口调试环境。

Pxretailer采用spring+vertx框架来实现。Vert.x最大的特点就在于异步（底层基于Netty），通过事件循环（EventLoop）来调起存储在异步任务队列（CallBackQueue）中的任务，大大降低了传统阻塞模型中线程对于操作系统的开销。因此相比较传统的阻塞模型，异步模型能够很大层度的提高系统的并发量。

## 技术风险

开发人员在编写代码时，会受各种因素的影响，导致系统代码或多或少的存在漏洞，甚至致命bug。因此系统在开发过程中，要求开发人员需在IDE中安装sonar，通过sonar自发的进行代码模块的review。同时定期组织项目管理人员和开发人员进行前后端代码评审，完善系统代码，提高代码质量，提升系统的稳定性和容错率。

Vert.x异步也带来了编码上的复杂性，想要编写优美的异步代码，就需要对lambda表达式、函数式编程、Reactive等技术非常熟悉才行，否则很容易导致你的代码一团糟，完全没有可读性。另外，异步模型的性能调优、异常处理与同步模型有很大差异，网络中相关资料较少，使用中遇到问题排查困难。

## 技术演变预测

随着系统开发的不断深入以及项目需求的不断扩大，系统数据将不断增加，可以采用mycat来实现数据库的分表分库，或者采用MongoDB结合mysql来实现数据的读写。

# 数据架构

1. 关键数据表设计：



1. 动静态数据处理：系统中通过PPM系统新增、修改信息均相对稳定；用户的操作日志以及终端上送的运行状态、运行日志等信息增量较大。目前所有数据都是保存在mysql中，随着时间的推移，数据不断增加，需要考虑分表分库。
2. 数据压力情况：系统中数据量最大的是终端的监控记录和操作日志。数据会由终端的增长和使用时间的积累不断累加，因此对应的数据量也会剧增。
3. 数据管理策略：mysql和redis都采用双机热备方式，保证数据的安全性。并对数据库添加访问权限的设置，同时对数据库中的敏感数据进行加密处理，防止数据库被非法操作、篡改。对数据库的操作行为进行监视与记录，一旦数据库出现安全问题，可以及时进行查询与分析。

# 接口设计

## Pxretailer提供给终端的接口

Pxretailer提供了http接口，数据使用json格式。

## PPM提供给外部系统的接口

ppm提供了http的restful接口，数据使用json格式，接口通过PPM给用户生成的api密钥进行权限验证。

# 部署架构

Nginx+tomcat+redis

比如我们向一台服务器的80端口发送了请求，该请求首先由Nginx监听到，其接受到请求内容后再转发给其他服务器（tomcat），其他服务器处理完后再将结果传送给Nginx，最后由Nginx来统一返回给初始请求端。

Web服务部署在多个tomcat服务器下时，采用redis来存储session，应用服务器接受新请求将Session信息保存到Cache DB中，当应用服务器发生故障，Web服务器（Apache/Nginx）会遍历寻找可用节点，分发请求，当应用服务器发现Session不在本机内存，则会去Cache DB中查找，如果找到，则复制到本机，这样就实现了Session的共享和高可用

