**基于WIFI探针的商业大数据分析系统**

**需求规格说明书**

**版本 <1.0>**

**文档信息及版本历史**

|  |  |  |  |
| --- | --- | --- | --- |
| 文档信息 | | | |
| 项目名称 | 基于WIFI探针的商业大数据分析系统 | | |
| 项目编号 | A1 | | |
| 文档名称 | 需求规格说明书 | | |
| 存储位置 | /Project/doc | | |
|  | | | |
| 版本 | 作者/修改者 | 日期 | 描述 |
| V1.0 | BugKiller | 2017-05-17 | 完成数据采集与书记展示模块 |
| V1.2 | BugKiller | 2017-05-24 | 完成数据模拟脚本 |
| V2.0 | BugKiller | 2017-05-27 | 增加数据图表 |
| V2.1 | BugKiller | 2017-06-03 | 修复数据预测模块BUG |
| V3.0 | BugKiller | 2017-05-07 | 优化算法 |
|  |  |  |  |

**版权信息**

本文件内容由【Zheng Li】负责解释

本文件的版权属于【Zheng Li】和

【BugKiller】

任何形式的散发都必须先得到本文档版本所属单位许可

# 概述

利用探针数据的客流分析打破模式束缚，不仅仅只是提供可信的客流数据分析，同时还利用延伸的标杆管理才能，深刻洞悉并提供有助于推动实际客流量和消费者习惯行为的一系列因素。这种专业才能呈现了经济分析，社交和环境等一些超出你控制范围的因素，却对商业绩效产生主要的冲击力。

## 编写目的

帮助商店或商场更好的了解客户需求，在信息化时代充分利用计算机作为管理手段提高管理水平和业务处理。

## 适用范围

中小型企业，商场。

## 术语和缩写

|  |  |
| --- | --- |
| **术语和缩写** | **解释** |
|  |  |

## 参考资料

《大数据基础与应用》，陈明，北京师范大学出版社

《大数据技术概论》, 娄岩，清华大学出版社

《大数据时代》，维克托·迈尔·舍恩伯格， 浙江人民出版社

《精通Python设计模式》，Sakis Kasampalis，人民邮电出版社

# 项目综述

## 项目介绍

*1.*

基于WIFI探针的商业大数据分析技术，通过WIFI探针收集MAC地址、出现时间、出现地点、与探针距离，探针设备会定时(3s)发送数据到服务端，通过一段时间数据平台产生大量的用户数据，然后使用大数据分析技术，采用离线计算和实时计算技术相结合的方式，能够对商业环境中门店的门前人流量、进店客流量、进入跳出量、新老顾客数量及新增量、在店平均时长进行分析，客流情况更新迅速，便于实时查看线下人群密集程度，有利于总部统一管理并及时调整营销活动方案。新老顾客比例，来访周期，顾客活跃度清晰展示，便于及时掌握入场顾客的构成，为销售策略调整提供参考。通过驻店时长可以快速掌握店招、产品或营销活动的吸引力。

*2.*

利用计算机运算速度快、存储信息容量大、处理逻辑问题强、功能强大的优势，从企业管理特别是查询与决策信息的管理需求出发，针对性强，功能齐备，旨在通过帮助该商场管理实现信息化，以提高商场管理的效率。

## 项目面向的用户

本项目面向商场店铺等一切直接面向顾客的商业环境，通过提供可靠的客流数据分析，同时还利用延伸的标杆管理才能，刻洞悉并提供有助于推动实际客流量和消费者习惯行为的一系列因素。这种专业才能呈现了经济分析，社交和环境等一些超出你控制范围的因素，却对商业绩效产生主要的冲击力。

## 项目应当遵循的标准或规范

**基本原则**

方便代码的交流和维护.

不影响编码的效率,不与大众习惯冲突.

使代码更美观,阅读更方便.

使代码的逻辑更清晰,更易于理解.

**编码**

\*所有的 Python 脚本文件都应在文件头标上如下标识或其兼容格式的标识

\*设置编辑器，默认保存为 utf-8 格式

\*不论什么情况使用 UTF-8 吧！这是王道！

# -\*- coding:utf-8 -\*- 或 #coding=utf-8

**命名**

一致的命名可以给开发人员减少许多麻烦，而恰如其分的命名则可以大幅提高代码的可读性，降低维护成本.

Python库的命名约定有点混乱,所以我们将永远不能使之变得完全一致--- 不过还是有公认的命名规范的.

新的模块和包(包括第三方的框架)必须符合这些标准,但对已有的库存在不同风格的, 保持内部的一致性是首选的

\*一些特殊的字符要避免.如小写字母'l','o'

**模块名**

1. 模块应该是不含下划线的,简短的,小写的名字.

例:module.py

2. 对于包内使用的模块，可以加一个下划线前缀.

例:\_internal\_module.py

**类名**

几乎没有例外，类名总是使用首字母大写单词串(CapWords)的约定.

不使用下划线连接单词，也不加入 C、T 等前缀

例:

class ThisIsAClass(object):

pass

**函数名**

函数名全部小写，由下划线连接各个单词

类似mixedCase函数名仅被允许用于这种风格已经占优势的上下文(如: threading.py) 以便保持向后兼容.

例:

def this\_is\_a\_func(self):

pass

**变量名**

变量名全部小写，由下划线连接各个单词

不论是类成员变量还是全局变量，均不使用 m 或 g 前缀

私有类成员使用单一下划线前缀标识，多定义公开成员，少定义私有成员。

变量名不应带有类型信息，因为 Python 是动态类型语言。如 iValue、names\_list、dict\_obj 等都是不好的命名。

**常量名**

常量名所有字母大写，由下划线连接各个单词

例:

WHITE = 0xffffffff

THIS\_IS\_A\_CONSTANT = 1

**异常名**

如果模块对所有情况定义了单个异常,它通常被叫做"error"或"Error"

似乎内建(扩展)的模块使用"error"(例如:os.error), 而Python模块通常用"Error" (例如: xdrlib.Error).

趋势是使用(CapWords)

缩写

1. 命名应当尽量使用全拼写的单词

2. 常用的缩写，如 XML、ID等，在命名时也应只大写首字母

例:class XmlParser(object):pass

3. 命名中含有长单词，对某个单词进行缩写

例:function 缩写为 fn;text 缩写为 txt;object 缩写为 obj等

**特殊命名**

1. 用下划线作前导或结尾的特殊形式是被公认的

\_single\_leading\_underscore(以一个下划线作前导): 弱的"内部使用(internal use)"标志

例: "from M import \*"不会导入以下划线开头的对象

2. single\_trailing\_underscore\_(以一个下划线结尾): 用于避免与Python关键词的冲突

例:"Tkinter.Toplevel(master, class\_='ClassName')"

3. \_\_double\_leading\_underscore(双下划线): 从Python 1.4起为类私有名

4. \_\_double\_leading\_and\_trailing\_underscore\_\_: 特殊的(magic) 对象或属性,存在于用户控制的(user-controlled)名字空间

例:\_\_init\_\_, \_\_import\_\_ 或 \_\_file\_\_

**缩进**

1. 使用制表符还是空格?

　　-- 永远不要混用制表符和空格.建议使用空格.

-- 我们内部应该都是使用的4个空格的tab.

**空行**

适当的空行有利于增加代码的可读性，加空行可以参考如下几个准则

1. 在类、函数的定义间加空行

-- 用两行空行分割顶层函数和类的定义,类内方法的定义用单个空行分割.

2. 额外的空行可被用于分割一组相关函数

3. 在 import 不同种类的模块间加空行

4. 在函数中的逻辑段落间加空行，即把相关的代码紧凑写在一起，作为一个逻辑段落，段落间以空行分隔

**空格**

空格在 Python 代码中是有意义的，因为 Python 的语法依赖于缩进，在行首的空格称为前导空格.这里不谈这个.

非前导空格在 Python 代码中没有意义，但适当地加入非前导空格可以增进代码的可读性。

3. 在二元算术、逻辑运算符前后加空格

例:a = b + c(好) a=b+c(不好)

4. “:”用在行尾时前后皆不加空格，如分枝、循环、函数和类定义语言；用在非行尾时两端加空格，如 dict 对象的定义

例:d = {'key' : 'value'}(好) d = {'key':'value'}(不好)

5. 括号（含圆括号、方括号和花括号）前后不加空格

例:do\_something(arg1, arg2)(好) do\_something( arg1, arg2 )(不好)

6. 逗号后面加一个空格，前面不加空格

例:print x, y(好) print x , y(不好)

7. 函数调用，索引切片，字典取值不要用空格

例:spam(x)(好)　spam (x)(不好); list[i](好) list [i](不好); dict[key](好) dict [key](不好)

**断行**

\*尽管现在的宽屏显示器已经可以单屏显示超过 256 列字符，但本规范仍然坚持行的最大长度不得超过 78 个字符的标准

1. 为长变量名换一个短名

例: this.\_is.a.very.long.variable\_name = this.\_is.another.long.variable\_name (不好)

variable\_name = this.\_is.another.long.variable\_name (好)

2. 在括号（包括圆括号、方括号和花括号）内换行

如果行长到连第一个括号内的参数都放不下，则每个元素都单独占一行

3. 在长行加入续行符强行断行，断行的位置应在操作符前，且换行后多一个缩进，以使维护人员看代码的时候看到代码行首即可判定这里存在换行

**导入**

Imports 通常被放置在文件的顶部,仅在模块注释和文档字符串之后,在模块的全局变量和常量之前.Imports应该有顺序地成组安放

对于内部包的导入是不推荐使用相对导入的.对所有导入都要使用包的绝对路径

import应该按照从最常用到最不常用的顺序分组放置,这几种模块中用空行分隔开来

import标准库

import第三方库

importGoogle App Engine 相关库

importDjango 框架相关库

importSoC framework 相关库

import基于 SoC 框架的模块

import应用程序特有的内容

2.从一个包含类的模块中导入类时

例:

from MyClass import MyClass

from foo.bar.YourClass import YourClass

如果发生地名字冲突，可以

import MyClass

import foo.bar.YourClass

然后再按下面使用即可

MyClass.MyClass

foo.bar.YourClass.YourClass

**注释**

\*\* 业界普遍认同 Python 的注释分为两种的概念，一种是由 # 开头的“真正的”注释，另一种是 docstrings。

\*\* 前者表明为何选择当前实现以及这种实现的原理和难点，后者表明如何使用这个包、模块、类、函数（方法），甚至包括使用示例和单元测试。

\*\* 坚持适当注释原则。对不存在技术难点的代码坚持不注释，对存在技术难点的代码必须注释

\*\* 推荐对每一个包、模块、类、函数（方法）写 docstrings，除非代码一目了然，非常简单

\*\* 包、模块、类、函数的第一个语句如果是字符串那么就是一个 \_\_doc\_\_ String。

**文件注释**

\*\* 每个文件开头都应该包含一个带有版权信息和许可声明的块注释。

## 主要特征

首先开发探针设备能够采集客户唯一的定位标识，比如MAC地址，通过数据分析技术，采用离线计算和实时计算结合的方式，为商业环境提供科学的、全面的数据决策依据。不仅对营销能力的评估，也可以对管理上进行优化, 及时掌握入场顾客的构成，为销售策略调整提供参考。通过驻店时长可以快速掌握店招、产品或营销活动的吸引力, 这种专业才能呈现了经济分析，社交和环境等一些超出你控制范围的因素，却对商业绩效产生主要的冲击力。

## 项目中的角色

|  |  |
| --- | --- |
| 角色名称 | 职责描述 |
| 李政 | 大数据平台的搭建，数据采集模块，数据存储模块。 |
| 李明志 | 数据实时分析与数据离线计算。 |
| 孔岳 | 大数据预测算法 |
|  |  |

# 功能性需求

## 功能性需求分类

|  |  |  |
| --- | --- | --- |
| 一级模块 | 二级模块 | 功能点编号及名称 |
| M1\_日常工作管理 | M1\_C1 数据库应用 | F\_1 mysql & hive Connect() |
| F\_2 mysql & hive Init() |
| F\_3 mysql & hive Execute() |
| M1\_C2数据接收 | F\_4 DataImport |
| F\_5 DataGatheringServer |
| M1\_C3数据处理 | F\_6 passenger() |
| F\_9 data\_operate() |
| F\_10 data\_process() |
| F\_11 Predict() |
|  | M1\_C3数据展示 | F\_11 Flask route() |

## 模块一

### 子模块一

数据库应用模块：提供使用数据库的方法与接口，包括mysql数据库与hive数据库的初始化。

#### 功能一

* **输入要素描述**

1. **输入画面设计**

由程序自动调用，无输入画面。

**b.输入字段定义**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 字段名 | 类型 | 数据种类 | 是否必输 | 长度 | 范围(精度） | 输入限制  （或数据字典） | 说明 |
| Sql | String | 字符 | 是 | 不限 | 无 | 全部 | 需要执行的SQL语句 |

**c.输入字段间的约束关系**

**无。**

* **功能描述**

|  |  |
| --- | --- |
| 功能编号 | F\_1 |
| 功能名 | mysql & hive Connect() |
| 功能描述 | 数据库链接函数。 |
| 优先级 | 1 |
| 使用角色 | 需进行数据库操作的函数。 |
| 运行条件 | 网络正常。 |
| 业务处理流程 | Pymysql模块生成的数据库连接对象。 |
| 补充说明 | 无。 |

* ***输出要素描述***

1. ***输出画面设计***

***无。***

***b.输出字段定义***

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 字段名 | 类型 | 数据种类 | 是否可选 | 说明 |
| DB | DBConnectObject | Object | 否 | 数据库连接对象。 |

#### 功能二

* **输入要素描述**

1. **输入画面设计**

由程序自动调用，无输入画面。

* **功能描述**

|  |  |
| --- | --- |
| 功能编号 | F\_2 |
| 功能名 | mysql & hive Init() |
| 功能描述 | 数据库初始化函数。 |
| 优先级 | 2 |
| 使用角色 | 初次使用本系统的环境。 |
| 运行条件 | 网络正常，具有数据库连接对象。 |
| 业务处理流程 | 通过数据库连接对象，执行内部建表语句。 |
| 补充说明 | 无。 |

#### 功能三

* **输入要素描述**

1. **输入画面设计**

由程序自动调用，无输入画面。

**b.输入字段定义**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 字段名 | 类型 | 数据种类 | 是否必输 | 长度 | 范围(精度） | 输入限制  （或数据字典） | 说明 |
| Sql | String | 字符 | 是 | 不限 | 无 | 全部 | 需要执行的SQL语句 |

**c.输入字段间的约束关系**

**无。**

* **功能描述**

|  |  |
| --- | --- |
| 功能编号 | F\_3 |
| 功能名 | mysql & hive Execute() |
| 功能描述 | Sql语句执行函数。 |
| 优先级 | 3 |
| 使用角色 | 需进行数据库操作的函数。 |
| 运行条件 | 网络正常，具有数据库连接对象，具有合理数据表。 |
| 业务处理流程 | 编辑数据库查询语句，执行数据库查询语句，返回查询结果。 |
| 补充说明 | 无。 |

* ***输出要素描述***

***b.输出字段定义***

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 字段名 | 类型 | 数据种类 | 是否可选 | 说明 |
| Cursor | CursorObject | 字符 | 是 | Cursor.\_rows 为结果集。 |

### 子模块二

数据采集模块：数据采集服务器，能采集并发数大于1000的JSON数据对象，并对数据进行清洗，分类，存储。

#### 功能一

DataImport()——数据导入模块。

* **输入要素描述**

1. **输入画面设计**

由程序自动调用，无输入画面。

1. **输入字段定义**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 字段名 | 类型 | 数据种类 | 是否必输 | 长度 | 范围(精度） | 输入限制  （或数据字典） | 说明 |
| SqoopCmd | String | 字符 | 是 | 不限 | 无 | 全部 | 需要执行的数据导入语句 |

**c.输入字段间的约束关系**

**无。**

* **功能描述**

|  |  |
| --- | --- |
| 功能编号 | F\_1 |
| 功能名 | DataImport（） |
| 功能描述 | 数据导入模块。 |
| 优先级 | 1 |
| 使用角色 | 每天定时自动执行，将历史数据由mysql导入到hive |
| 运行条件 | 网络正常，Hadoop平台开启，HiveServer2服务开启。 |
| 业务处理流程 | 执行Sqoop工具查询Mysql数据表，将结果集导入HDFS系统，之后再导入相应的Hive数据表。 |
| 补充说明 | 分全量导入与增量导入，导入完成自动清除数据表。 |

* ***输出要素描述***

***a.输出画面设计***

***无。***

#### 功能二

DataGatheringServer——数据导入模块。

* **输入要素描述**

1. **输入画面设计**

建立在HTTP服务器之上，等待输入POST数据。

1. **输入字段定义**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 字段名 | 类型 | 数据种类 | 是否必输 | 长度 | 范围(精度） | 输入限制  （或数据字典） | 说明 |
| Data | JSon | Json | 是 | 不限 | 无 | 全部 | 探针采集到的Json数据 |

**c.输入字段间的约束关系**

**无。**

* **功能描述**

|  |  |
| --- | --- |
| 功能编号 | F\_2 |
| 功能名 | DataGatheringServer |
| 功能描述 | 采用Flask Web框架，系统稳定，支持大量并发数据采集不失真。 |
| 优先级 | 1 |
| 使用角色 | 开启系统自动启动 |
| 运行条件 | 网络正常。 |
| 业务处理流程 | 将接收到的JSON进行数据清洗纠错后批量存入数据库，将个别数据整理归纳存入特别数据表。 |
| 补充说明 | 支持数据并发数在1500以上，硬件环境支持理论数据量将更高。 |

* ***输出要素描述***

***a.输出画面设计***

***POST数据请求。***

### 子模块三

数据处理模块：将在数据库中的数据利用大数据算法进行归类处理，将处理结果直接展示或者存入数据库。

#### 功能一

Passanger()

* **输入要素描述**

1. **输入画面设计**

由程序自动调用，无输入画面。

**b.输入字段定义**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 字段名 | 类型 | 数据种类 | 是否必输 | 长度 | 范围(精度） | 输入限制  （或数据字典） | 说明 |
| Command | String | 字符 | 是 | 不限 | 无 | 全部 | 不同数据页面的不同代表。 |
| Time | DateTime | 日期格式 | 是 | Y-M-D H:M:S | 无 | 全部 | 当前时间。 |

**c.输入字段间的约束关系**

**无。**

* **功能描述**

|  |  |
| --- | --- |
| 功能编号 | F\_1 |
| 功能名 | Passenger（） |
| 功能描述 | 根据不同的Command调用不同的算法计算需要展示的实时数据，然后发回数据。 |
| 优先级 | 2 |
| 使用角色 | HTML页面中的JavaScript |
| 运行条件 | 网络正常，JavaScript请求数据。 |
| 业务处理流程 | 访问相应页面会生成相应的Command传入本函数，然后将计算结果返回HTML图表进行展示。 |
| 补充说明 | 无。 |

* ***输出要素描述***

1. ***输出画面设计***

***返回Json数据。***

***b.输出字段定义***

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 字段名 | 类型 | 数据种类 | 是否可选 | 说明 |
| Data | Json | Json | 否 | 数据展示的结果集。 |

#### 功能二

* **输入要素描述**

1. **输入画面设计**

由程序自动调用，无输入画面。

**b.输入字段定义**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 字段名 | 类型 | 数据种类 | 是否必输 | 长度 | 范围(精度） | 输入限制  （或数据字典） | 说明 |
| Mac | String | 字符 | 是 | 不限 | 无 | 全部 | Mac地址 |

**c.输入字段间的约束关系**

**无。**

* **功能描述**

|  |  |
| --- | --- |
| 功能编号 | F\_2 |
| 功能名 | 来访周期计算函数 |
| 功能描述 | 计算每日的来访周期并存入数据库 |
| 优先级 | 1 |
| 使用角色 | System |
| 运行条件 | 网络正常。 |
| 业务处理流程 | 根据传入的mac地址来判断data\_set中是否有该mac,若没有则将mac和时间插入data\_set,若有则根据新老时间来计算来访周期,删除旧数据,将mac,时间和来访周期存入表中. |
| 补充说明 | 无。 |

* ***输出要素描述***

1. ***输出画面设计***

***无。***

#### 功能三

* **输入要素描述**

**无。**

* **功能描述**

|  |  |
| --- | --- |
| 功能编号 | F\_2 |
| 功能名 | Data\_operate调用函数 |
| 功能描述 | 根据相应mac地址调用来访周期处理函数 |
| 优先级 | 1 |
| 使用角色 | System |
| 运行条件 | 网络正常。 |
| 业务处理流程 | 接收data\_get()函数的返回值,根据得到的数据长度将数据分为4部分,启动四个进程来执行data\_operate()函数,将四部分数据分别分给四个进程来执行. |
| 补充说明 | 无。 |

* ***输出要素描述***

1. ***输出画面设计***

***无。***

# 非功能性需求

## 软件环境需求

|  |  |
| --- | --- |
| 需求名称 | 详细要求 |
| 操作系统 | Linux Gnome图形界面 |
| JDK版本 | 1.7及以上 |
| 中间件 | Sqoop |
| 数据库 | Hive Mysql |
| Python版本 | 2.7 |
| 大数据系统 | Hadoop2.0 |

## 硬件环境需求

|  |  |  |  |
| --- | --- | --- | --- |
| 需求名称 | 数量 | 每台需要存储空间 | 详细要求 |
| PC | 3 | 300G | 处理器：2.9 GHz 32位或者64位处理器  　内存：8 GB 及以上  　显卡：支持DirectX 9 128M 及以上(开启AERO效果)  　硬盘空间：300G以上(主分区，NTFS格式) |

## 故障处理

|  |  |
| --- | --- |
| 故障描述 | 详细要求 |
| 服务器故障 | Web服务器无法运行，请检查是否有程序占用5000 4000端口。 |
| 系统运行故障 | 模块丢失请自行安装系统所需模块。 |
| 用户操作不正确 | 数据库链接相关信息在config.conf中，若修改请修改本文件。 |
| Hadoop启动失败 | 本机IP需要与master主机名对应，从机IP需要与nodex对应，三台机器需配置SSH免密码登陆，否则可能无法启动从机或者Hadoop。 |
| Hive启动失败 | Hadoop未启动，或者Hadoop位于SafeNodeMode，请手动解除。 |
| 提示“须在脚本目录运行” | 启动命令须在脚本目录下，请切换至脚本所在目录。 |

# 附件

