|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **文档编号** | 1-2 | | **当前版本** | **1.0** |
| **当前状态** | 草稿 |  | **发布日期** | 2017-08-25 |
| 发布 | √ |

< HeatPoint可视化系统>

概要设计说明

**拟制人：** 张仁童

**审核人：**

**批准人：**

**目录**

[1引言 2](#_Toc491455820)

[1.1编写目的 2](#_Toc491455821)

[1.2背景 2](#_Toc491455822)

[1.3参考资料 2](#_Toc491455823)

[2总体设计 3](#_Toc491455824)

[2.1需求规定 3](#_Toc491455825)

[2.2运行环境 4](#_Toc491455826)

[2.3系统流程图 5](#_Toc491455827)

[3接口设计 5](#_Toc491455828)

[3.1用户接口 5](#_Toc491455829)

[3.1.1常用接口 5](#_Toc491455830)

[3.1.2非常用接口 7](#_Toc491455831)

[3.1.3静态接口 8](#_Toc491455832)

[3.2外部接口 9](#_Toc491455833)

[4系统数据结构设计 10](#_Toc491455834)

[4.1逻辑结构设计要点 10](#_Toc491455835)

[4.1.1 Location： 10](#_Toc491455836)

[4.1.2 Show： 10](#_Toc491455837)

[4.1.3 Trade： 10](#_Toc491455838)

[4.2物理结构设计要点 11](#_Toc491455839)

[5系统出错处理设计 13](#_Toc491455840)

[5.1出错信息 13](#_Toc491455841)

[5.2系统维护设计 17](#_Toc491455842)

[5.2.1 Windows下的错误日志 17](#_Toc491455843)

[5.2.2 liunx下的错误日志 17](#_Toc491455844)

# 1引言

## 1.1编写目的

软件概要设计是从总体上把握系统设计框架，他包括模块划分、处理流程和接口设计，概要设计说明书对上述内容作了总体描述，体现了用户需求与应用系统实现之间的关系，在设计过程中起到了提纲挈领的作用。

预期读者：HeatPoint可视化系统项目团队。

## 1.2背景

说明：

我们根据需求开发的一款能够智能抓取互联网上所有包含地理位置信息的内容，并分类可视化于地图上，使用户可以方便快速的获取到需要的信息。主要功能是实现互联网事件天级或更高频次抓取，能够采集政治会议、展会、体育赛事、演唱会、突发异常天气等各类中比较主要的事件，并能够事件属性自动化提取。属性需包括主办方级别、影响区域范围、影响力度、影响人群等。

## 1.3参考资料

1. 文档格式要求按照我国GB/T8567-1988国家标准和IEEE/ANSI830-1993标准规范要求进行。包括以下文件：
2. 软件工程项目开发文档范例
3. 软件工程国家标准文档
4. 软件需求说明书编写规范

# 2总体设计

## 2.1需求规定

1．实现互联网事件天级或更高频次抓取，能够采集政治会议、展会、体育赛事、演唱会、突发异常天气等各类中比较主要的事件，每个类型的事件数据至少来自2个数据源（网站）。

表2.1 数据来源

|  |  |  |
| --- | --- | --- |
| 数据类型 | 数据源（网站） | |
| 展会 | E展会 | http://www.eshow365.com/ |
| 中国会展门户 | http://www.cnena.com/ |
| 体育赛事  演唱会 | 大麦网 | https://www.damai.cn/ |
| 永乐票务 | http://www.228.com.cn/ |
| 突发异常天气 | 中国天气 | http://www.weather.com.cn/ |
| 政治会议 | 活动家 | https://www.huodongjia.com/ |
| 航班信息 | 飞常准 | http://www.variflight.com/ |

2.实现事件的去重功能，主要有两个方面，一是不同数据源（网站）的事件去重，二是不同天抓取的事件去重。

**去重算法：**本项目最初采用相似度阈值去重法，但因大部分展会、演唱会等名称相似但实则不同的信息存在被误删的现象，如：第X届、XX站等。最终采取近义词匹配去重法，结合时间及地理位置信息，有效避免了大量信息误删的情况。

3.事件画像建模，即事件属性自动化提取。属性需包括主办方级别、影响区域范围、影响力度、影响人群等。

在“设置>>下载数据”中选择要下载的文件类型，服务器将自动提取事件属性并生成相应文件。

4.可视化展现，可视化展示事件的影响情况，形式不限。

**主视图**采用百度地图清晰准确的展示热点的地理位置信息，并提供可交互的动态展示详细信息。

**统计视图**采用echarts图表工具展示，包括：

**分时地区**统计视图：根据用户选择的时间动态生成改时间段内各省份的各种数据总量，并展示在小型中国地图中。

**全局日期**统计视图：把数据库中所有今年的数据，按日期分类统计，并生成一个3D视图，其中x轴为日期，y轴为月份，z轴为数量。

**局部日期**统计视图：把该场馆在该时间段内的所有类型数据的开始时间展示到二维日历视图中。

## 2.2运行环境

表2.2 运行环境

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 开发环境 | | Window 10、MyEclipse 10、Tomcat 6、Java EE 6、MySQL 5.5 | | |
| 实验平台 | | CentOS 6.5、Java JDK 1.8、Tomcat 8、MySQL 5.6 | | |
| 开发语言 | | Java、JavaScript、HTML、CSS | | |
| 第三方代码 | 后端 | Jsoup | 1.8.3 | https://github.com/jhy/jsoup |
| Xsoup | 0.3.1 | https://github.com/code4craft/xsoup |
| Log4j | 1.2.17 | https://github.com/apache/log4j |
| Poi | 3.16 | https://github.com/apache/poi |
| CrawlerSYS | 0.1.0 | https://github.com/zrtzrt/CrawlerSYS |
| HikariCP | 3.16 | https://github.com/brettwooldridge/HikariCP |
| boilerpipe | 0.1.0 | https://github.com/kohlschutter/boilerpipe |
| 前端 | Bootstrap | 3.3.7 | https://github.com/twbs/bootstrap |
| Echarts | 3 | https://github.com/ecomfe/echarts |
| BaiduMap | 2 | http://map.baidu.com |

## 2.3系统流程图

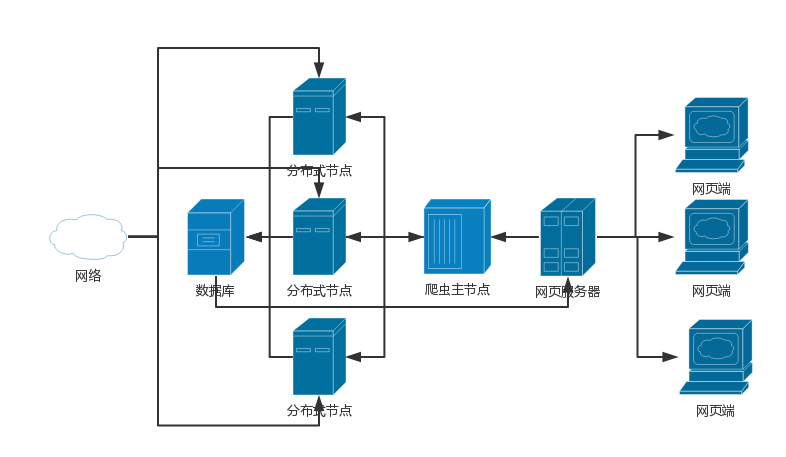


图2.3 系统流程图

# 3接口设计

## 3.1用户接口

### 3.1.1常用接口

#### Crawler(List<String> url, List<String> xpath) Crawler

Create new crawler by List of url and xpath.

#### Crawler(List<String> url, Map<String,String> xpath) Crawler

Create new crawler by List of url and Map of lable(key),xpath(value).

#### Crawler(List<String> url, String urlRegex, int limit) Crawler

Create new auto crawler by List of url. Use urlRegex to limit url type.It will get title and content automatic without xpath.

#### run() void

Start running in this thread. It will wait until all node finished.

#### start() void

Start running in other thread.

#### periodRun(int timeoutInHours) void

Run in several hours.

#### fixTimeRun(String time) void

Run in fixed time like 12:00:00. Format: "HH:mm:ss"

#### distributed(String[] node) Crawler

Add node. Default: "127.0.0.1:6543". Format: "ip : port".

#### exception(String urlRegex, List<String> xpath) Crawler

Select special url and dispose with special xpath.

#### save(boolean asList, String table, String[] lable) Crawler

Data persistence. Save to your database (mysql).If asList is false only first date pre xpath.

#### jdbc(String ip,String user,String pw) Crawler

JDBC configuration. Default: "127.0.0.1:3306/crawler","root","root". Format: "ip : port /dataBaseName".

#### recursion(String linkpath, int limit) Crawler

Use breadth-first crawler. Default: "//a/@href",5000

#### limit(int limit) Crawler

The amount of data to send each time.Less limit more accurate monitoring but slower.Default: 30

#### thread(int thread) Crawler

Change crawler speed.Depend on your network bandwith.Recommend: 2~3 per Mbps.Default: 10

#### sleep(int sleep) Crawler

Slow down crawler to avoid connection time out error.Default: 0

#### custom(Dispose d) Crawler

If you want to processing acquired data or use your own way to save date.you can implements Dispose and use it as parameter.

### 3.1.2非常用接口

#### exceptions(String[] urlRegex, List<List<String>> xpath) Crawler

Add several exceptions at one times.

#### header(Map<String,String> header) Crawler

Add unify header for any url. Default:{"User-Agent", "Mozilla/4.0 (compatible; MSIE 5.0; Windows NT; DigExt)"}

#### cookie(Map<String,String> cookie) Crawler

Add unify cookie for any url.

#### header(List<String> key,List<String> val) Crawler

#### cookie(List<String> key,List<String> val) Crawler

#### recursion(String linkpath) Crawler

If you use it without limit.It will never over.

#### jdbc(String ip,int port,String dbName,String user,String pw) Crawler

#### save(boolean asList, String table) Crawler

#### startServer() Crawler

Start crawler server by default port.

#### startServer(int port) Crawler

Default: 6543

### 3.1.3静态接口

#### WebCrawler.getConnect(String url, Map<String,String> header, Map<String,String> cookie) Connection

Return org.jsoup.Connection for next step.

#### WebCrawler.send(String url) void

Send http request without response.

#### WebCrawler.get(String url, Map<String,String> header, String encode) String

Get json from http request.

#### WebCrawler.getByXpath(Document doc, String[] xpath, int sleepTime) List<List<String>>

Extraction data by xpath from Document.

#### WebCrawler.ignoreSsl() void

Ignore SSL certificate.To avoid Validation failure.

#### StringHelper.toInt(String str) String

Same as Integer.parseInt(String s) but more smart.

#### similarity(String str1,String str2) double

Return similarity between two String.

#### StringHelper.similarity(List<String> strList,String str,double filter) String

Return most similar String in List.If less than filter return 0.

#### StringHelper.sqlCREATE(String table,String[] lable,int[] length) String

#### StringHelper.sqlSELECT(String table,String[] lable,String[] condition) String

#### StringHelper.sqlINSERT(String table,String[] lable,String[] data) String

#### StringHelper.sqlDate(Date date) String

## 3.2外部接口

**resultDispose(CrawlerReturnEntity back,HikariDataSource ds) CrawlerReturnEntity**

| **name** | **class** | **describe** |
| --- | --- | --- |
| url | String | page url |
| res | List<List<String>> | date by xpath order |
| resMap | Map<String, List<String>> | you can use it if you add lables |
| link | List<String> | add link if you want |

# 4系统数据结构设计

## 4.1逻辑结构设计要点

### 4.1.1 Location：

int id;int type;String name; String city; String province;String detail;Double longitude; Double latitude;

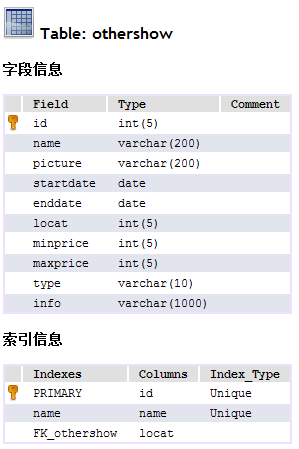
### 4.1.2 Show：

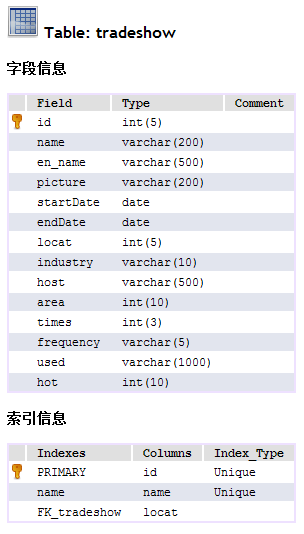
int id;String name;String picture;Date startDate;Date endDate; LocationEntity locat; int MINprice;int MAXprice;int showtype;String info; List<String> url;

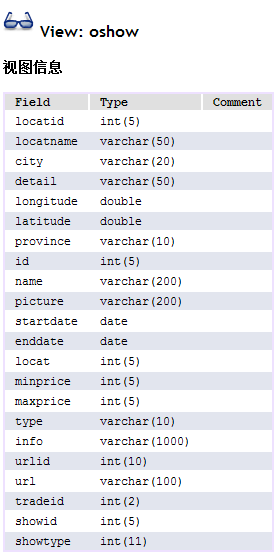
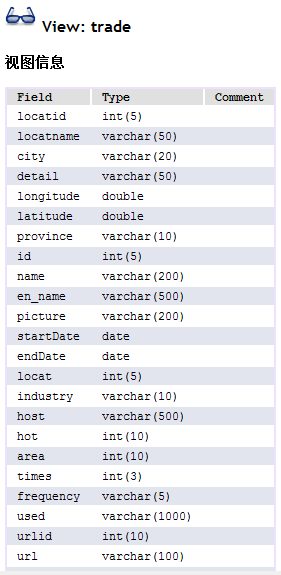
### 4.1.3 Trade：

int id;String name;String en\_name;String picture;Date startDate;Date endDate; LocationEntity locat;String industry;String host;int area;int times;String frequency;String used;List<String> url; int hot;

## 4.2物理结构设计要点







# 5系统出错处理设计

## 5.1出错信息

用一览表的方式说朗每种可能的出错或故障情况出现时，系统输出到错误日志中信息的形式、含意及处理方法。

表6.1 故障处理表

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 故障 | | 系统输出信息 | 含义 | 处理方法 |
| Read timed out或connect timed out | | Exception  java.net.SocketTimeoutException: Read timed out  at java.net.SocketInputStream.socketRead0(Native Method)  at java.net.SocketInputStream.read(SocketInputStream.java:129)  at com.sun.net.ssl.internal.ssl.InputRecord.readFully(InputRecord.java:293)  at com.sun.net.ssl.internal.ssl.InputRecord.readV3Record(InputRecord.java:405)  at com.sun.net.ssl.internal.ssl.InputRecord.read(InputRecord.java:360)  at com.sun.net.ssl.internal.ssl.SSLSocketImpl.readRecord(SSLSocketImpl.java:789)  at com.sun.net.ssl.internal.ssl.SSLSocketImpl.readDataRecord(SSLSocketImpl.java:746)  at com.sun.net.ssl.internal.ssl.AppInputStream.read(AppInputStream.java:75)  at java.io.BufferedInputStream.fill(BufferedInputStream.java:218)  at java.io.BufferedInputStream.read1(BufferedInputStream.java:258)  at java.io.BufferedInputStream.read(BufferedInputStream.java:317)  at sun.net.www.http.HttpClient.parseHTTPHeader(HttpClient.java:687)  at sun.net.www.http.HttpClient.parseHTTP(HttpClient.java:632)  at sun.net.www.protocol.http.HttpURLConnection.getInputStream(HttpURLConnection.java:1064)  at java.net.HttpURLConnection.getResponseCode(HttpURLConnection.java:373)  at sun.net.www.protocol.https.HttpsURLConnectionImpl.getResponseCode(HttpsURLConnectionImpl.java:318)  at org.jsoup.helper.HttpConnection$Response.execute(HttpConnection.java:516)  at org.jsoup.helper.HttpConnection$Response.execute(HttpConnection.java:493)  at org.jsoup.helper.HttpConnection.execute(HttpConnection.java:205)  at org.jsoup.helper.HttpConnection.get(HttpConnection.java:194)  at CrawlerSYS.node.SingleUrlCrawler.call(SingleUrlCrawler.java:49)  at CrawlerSYS.node.SingleUrlCrawler.call(SingleUrlCrawler.java:1)  at java.util.concurrent.FutureTask$Sync.innerRun(FutureTask.java:303)  at java.util.concurrent.FutureTask.run(FutureTask.java:138)  at java.util.concurrent.ThreadPoolExecutor$Worker.runTask(ThreadPoolExecutor.java:886)  at java.util.concurrent.ThreadPoolExecutor$Worker.run(ThreadPoolExecutor.java:908)  at java.lang.Thread.run(Thread.java:619) | 在网络带宽不足的情况下，开启同时读取网络信息的线程过多，导致的网络带宽被占用，无法正常获取网络连接。 | 减少线程数，或增加爬虫休眠时间，  或增加网络爬虫节点数量，或换用更大带宽、更高质量网络。 |
| java.lang.reflect.InvocationTargetException  Caused by: java.lang.IndexOutOfBoundsException: Index: 0, Size: 0 | | Exception  java.lang.reflect.InvocationTargetException  at sun.reflect.GeneratedMethodAccessor10.invoke(Unknown Source)  at sun.reflect.DelegatingMethodAccessorImpl.invoke(DelegatingMethodAccessorImpl.java:25)  at java.lang.reflect.Method.invoke(Method.java:597)  at CrawlerSYS.node.SingleUrlCrawler.call(SingleUrlCrawler.java:89)  at CrawlerSYS.node.SingleUrlCrawler.call(SingleUrlCrawler.java:1)  at java.util.concurrent.FutureTask$Sync.innerRun(FutureTask.java:303)  at java.util.concurrent.FutureTask.run(FutureTask.java:138)  at java.util.concurrent.ThreadPoolExecutor$Worker.runTask(ThreadPoolExecutor.java:886)  at java.util.concurrent.ThreadPoolExecutor$Worker.run(ThreadPoolExecutor.java:908)  at java.lang.Thread.run(Thread.java:619)  Caused by: java.lang.IndexOutOfBoundsException: Index: 0, Size: 0  at java.util.ArrayList.RangeCheck(ArrayList.java:547)  at java.util.ArrayList.get(ArrayList.java:322)  at dataSource.MeetingFromHDJ.resultDispose(MeetingFromHDJ.java:69)  ... 10 more | 在网页上读取不到有用信息，源信息位置不存在，可能是由网站改版导致的信息错位。 | 请联系开发人员对该网站做重新适配。 |
| Data truncation: Data too long for column 'frequency' at row 1 | Exception:INSERT INTO `tradeshow`(name,en\_name,picture,startDate,endDate,locat,industry,host,area,times,frequency,hot) VALUES("2017第九届海南医疗器械展览会" , "9th hainan medical equipment exhibition (2017)" , "http://www.eshow365.com/UserUpload/ZhanHui/Title/201612070141190518.png" , "2017-11-10" , "2017-11-12" , "203" , "医疗保健" , "海南省卫生和计划生育委员会,海南省商务厅,海南广播电视总台" , "6000" , "9" , "展会周期未定" , "2025")  com.mysql.jdbc.MysqlDataTruncation: Data truncation: Data too long for column 'frequency' at row 1  at com.mysql.jdbc.MysqlIO.checkErrorPacket(MysqlIO.java:3489)  at com.mysql.jdbc.MysqlIO.checkErrorPacket(MysqlIO.java:3423)  at com.mysql.jdbc.MysqlIO.sendCommand(MysqlIO.java:1936)  at com.mysql.jdbc.MysqlIO.sqlQueryDirect(MysqlIO.java:2060)  at com.mysql.jdbc.ConnectionImpl.execSQL(ConnectionImpl.java:2536)  at com.mysql.jdbc.StatementImpl.executeUpdate(StatementImpl.java:1564)  at com.mysql.jdbc.StatementImpl.executeUpdate(StatementImpl.java:1485)  at com.zaxxer.hikari.proxy.StatementProxy.executeUpdate(StatementProxy.java:108)  at com.zaxxer.hikari.proxy.HikariStatementProxy.executeUpdate(HikariStatementProxy.java)  at dao.impl.TradeDaoImpl.insert(TradeDaoImpl.java:376)  at dao.impl.TradeDaoImpl.save(TradeDaoImpl.java:111)  at dataSource.TradeFromEShow.resultDispose(TradeFromEShow.java:252)  at sun.reflect.GeneratedMethodAccessor3.invoke(Unknown Source)  at sun.reflect.DelegatingMethodAccessorImpl.invoke(DelegatingMethodAccessorImpl.java:25)  at java.lang.reflect.Method.invoke(Method.java:597)  at CrawlerSYS.node.SingleUrlCrawler.call(SingleUrlCrawler.java:89)  at CrawlerSYS.node.SingleUrlCrawler.call(SingleUrlCrawler.java:1)  at java.util.concurrent.FutureTask$Sync.innerRun(FutureTask.java:303)  at java.util.concurrent.FutureTask.run(FutureTask.java:138)  at java.util.concurrent.ThreadPoolExecutor$Worker.runTask(ThreadPoolExecutor.java:886)  at java.util.concurrent.ThreadPoolExecutor$Worker.run(ThreadPoolExecutor.java:908)  at java.lang.Thread.run(Thread.java:619) | | 源数据格式有误，超出数据库可以存放的范围。"展会周期未定"不可被数据库存入。 | 调整数据库可接受数据的范围，或手动录入数据库。 |

## 5.2系统维护设计

因本爬虫系统需要不间断运行在服务器上，为不干扰系统的正常运行，需要维护人员在服务器在查看错误日志。

日志分为两种，一是正常情况下的所有操作信息记录，名为heatpoint-log.log。另一种是只有在系统出错时才记录的日志，名为heatpoint-error.log。

### 5.2.1 Windows下的错误日志

Windows下的错误日志存放位置为C:\Logs

### 5.2.2 liunx下的错误日志

liunx下的错误日志存放位置为tomcat安装目录下的/tomcat/logs/