# 普林云爬虫架构设计

目录

[普林云爬虫架构设计 1](#_Toc446506757)

[1 平台特点 4](#_Toc446506758)

[1.1 分布式 4](#_Toc446506759)

[1.2 反爬技术 4](#_Toc446506760)

[1.2.1 IP替换策略 4](#_Toc446506761)

[1.2.2 爬取频率控制 4](#_Toc446506762)

[1.2.3 失败数据回收循环爬取 5](#_Toc446506763)

[1.2.4 验证码破解 5](#_Toc446506764)

[1.3 可视化管理监控 5](#_Toc446506765)

[1.4 自然语言处理 6](#_Toc446506766)

[2 系统架构 6](#_Toc446506767)

[2.1 设计目标 6](#_Toc446506768)

[2.2 设计图 6](#_Toc446506769)

[2.3 对象名词 7](#_Toc446506770)

[2.3.1 Job 7](#_Toc446506771)

[2.3.2 Job Group 7](#_Toc446506772)

[2.3.3 生成器 7](#_Toc446506773)

[2.3.4 下载器 7](#_Toc446506774)

[2.3.5 解析器 7](#_Toc446506775)

[2.3.6 调度规则 8](#_Toc446506776)

[2.3.7 防重器 8](#_Toc446506777)

[3 采集层 8](#_Toc446506778)

[3.1 设计目标 8](#_Toc446506779)

[3.2 主要对象 8](#_Toc446506780)

[3.3 设计图 8](#_Toc446506781)

[3.4 流程图 8](#_Toc446506782)

[4 结构化层 8](#_Toc446506783)

[4.1 设计目标 8](#_Toc446506784)

[4.2 主要对象 9](#_Toc446506785)

[4.2.1 监控器 9](#_Toc446506786)

[4.2.2 解析器 9](#_Toc446506787)

[4.2.3 去重器 9](#_Toc446506788)

[4.3 设计图 10](#_Toc446506789)

[4.4 流程图 11](#_Toc446506790)

[5 存储层 11](#_Toc446506791)

[5.1 设计目标 11](#_Toc446506792)

[5.2 设计图 12](#_Toc446506793)

[5.3 主要对象 12](#_Toc446506794)

[5.3.1 MongoDB 12](#_Toc446506795)

[5.3.2 MySQL 12](#_Toc446506796)

[5.3.3 NFS 12](#_Toc446506797)

[5.3.4 小文件 13](#_Toc446506798)

[5.3.5 大文件 13](#_Toc446506799)

[5.3.6 文件索引 13](#_Toc446506800)

[5.4 流程图 14](#_Toc446506801)

[6 系统层 15](#_Toc446506802)

[6.1 设计目标 15](#_Toc446506803)

[6.2 主要对象 15](#_Toc446506804)

[6.3 设计图 15](#_Toc446506805)

[6.4 流程图 15](#_Toc446506806)

# 平台特点

## 分布式

云爬虫使用业内最为成熟的、分布式、可水平扩展方案设计完成，同时支持单机内多线程、多进程运行，大大提高爬虫的可扩展性。

## 反爬技术

随着爬虫应用的日益广泛，很多网站也采取了越来越多的反爬技术，通过限制爬虫服务器IP访问等行为来保护源数据不被获取，我们进行了一系列的技术调研，来提高单次爬取及累计爬取数据的成功率，最终实现将所有数据爬取并存储下来的目标

### IP替换策略

从互联网上爬取免费的IP代理地址，加入IP池，也可通过购买，或人工加入IP列表到IP池；爬虫程序读取IP池，获取可用IP，按一定频率替换云爬虫服务器的IP，以此来应对目标数据源服务器的封IP行为。

### 爬取频率控制

IP池有限的情况下，爬取频率的控制很有意义，需要在爬虫程序中设置单位时间内的爬取速度，提高单个IP的爬取成功率。IP池资源丰富时，可把爬取频率设置到最高。

### 失败数据回收循环爬取

对于爬取失败的数据，除了进行必要的日志记录外，同时将失败的数据收集起来，作为列表输出，本来爬取结束后，立即启动下一轮爬虫任务，对失败名单重新爬取，形成一个闭环，直到所有数据爬取成功为止。支持设置一个阈值，爬取失败超过一定粗疏的名单不再爬取，已应对不存在的数据这种情况

### 验证码破解

验证码破解步骤：经过图片验证码类型识别，图片处理，降噪，二值化，建模等步骤，最终输出图片内容结果

## 可视化管理监控

通过可视化的界面监控并管理采集层、分析层、存储层的行为。比如设置爬虫爬取数据的频率、监控云爬虫的运行状态等

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **序号** | **模块名称** | **子模块** |  | **说明** |
|  | 爬虫管理监控平台 | 爬虫管理 |  | 爬虫代码配置、爬虫日志、爬虫下载日志、爬虫分析日志、数据下载 |
|  | 实时监控 |  | 最近24小时分钟级别监控 |
|  | 系统管理 |  | 爬虫参数、操作日志 |

## 自然语言处理

将自然语言与计算机数据进行互转，对文本进行分词、词性分析、提取核心词组、文本聚类、情感分析，结合业务需求，最终将非结构化的文本数据提取为结构化的数据

# 系统架构

## 设计目标

* 分层化，减少耦合
* 错误跟踪，能够定位到具体层面
* 报警提示，可定制
* 完善的运行日志

## 设计图



## 对象名词

### Job

Job是核心对象。Job定义为一个数据源，比如法院破产网。包含如下内容：

1. 生成器
2. 生成器调度规则
3. 下载器
4. 下载器调度规则
5. 解析器
6. 解析器调度规则
7. 源数据存储
8. 解析后数据存储
9. 生成器日志
10. 下载器日志
11. 解析器日志
12. 错误日志
13. 报警提示
14. 运行日志
15. 父Job
16. 所属Job Group

### Job Group

定义为：Job的分组

### 生成器

定义为：能够生成URI，并输出到stdout的工具

### 下载器

定义为：根据URI，获得对方数据的工具。

### 解析器

定义为：专门解析下载器下载的数据，并转化为JSON格式输出的工具

### 调度规则

定义：类似Crontab的语法

### 防重器

定义：一种可以根据指定规则，来防止重复的工具。规则有：

1. 内容MD5
2. ID主键

# 采集层

## 设计目标

* 获得源数据，并提供给结构化层进行下一步的处理
* 能够支持拓展到1000台+机器
* 支持主流的URI，比如HTTP、HTTPS。
* 可以扩展新的URI协议
* 具备如下的反爬虫模块：动态页面、验证码识别、代理自动切换
* 错误追踪
* 报警提示，阈值设置

## 主要对象

1. URI生成器对象

|  |
| --- |
| 生成器管理对象 |
| -任务列表  -任务数量  -执行时序 |
| -添加任务()  -删除任务()  -获取一个任务()  -更新一个任务()  -查看一个任务() |

1. 生成器调度对象

|  |
| --- |
| 生成器调度对象 |
| -生成器列表  -管理节点列表  -生成器与节点映射表 |
| -分配生成器给节点()  -获得一个生成器对应的节点()  -获得一个节点对应的生成器() |

1. 下载器对象

|  |
| --- |
| 下载器管理对象 |
| -下载器列表  -下载器数量  -执行时序 |
| -添加下载器()  -删除下载器()  -获取一个下载器()  -更新一个下载器()  -查看一个下载器() |

1. 下载器调度对象

|  |
| --- |
| 下载器调度对象 |
| -下载器列表  -管理节点列表  -下载器与节点映射表 |
| -分配任务给节点()  -获得下载器对应的节点()  -获得一个节点对应的下载器任务() |

1. 生成日志对象

|  |
| --- |
| 生成日志 |
| -添加时间  -所在主机  -运行总时间  -运行生成器  -生成文件大小 |
|  |

1. 下载日志对象

|  |
| --- |
| 下载日志对象 |
| -添加时间  -所在主机  -运行总时间  -运行下载器  -生成文件大小 |
|  |

1. 错误日志对象

|  |
| --- |
| 错误日志对象 |
| -错误原因  -添加时间  -所在主机  -运行总时间  -运行生成器 |
|  |

1. 报警日志对象

|  |
| --- |
| 报警日志对象 |
| -添加时间  -所在主机  -报警原因 |
|  |

1. 代理对象

|  |
| --- |
| 代理 |
| -代理总数  -代理列表 |
| -添加代理()  -删除代理()  -获取一个代理()  -自动爬取代理() |

1. 图片解析对象

|  |
| --- |
| 图片解析 |
| -图片内容类型  -解析器列表 |
| -添加解析器()  -删除解析器()  -跟新解析器()  -获取一个解析器()  -解析图片()  -获得图片内容类型() |

1. 下载节点管理对象

|  |
| --- |
| 管理节点对象 |
| -节点列表 |
| -添加节点()  -删除节点()  -跟新节点()  -获取一个节点信息()  -获得一个节点运行任务信息() |

## 设计图



## 流程图











# 结构化层

## 设计目标

* 支持多种格式源数据，包含HTML、JSON、PDF和图片
* 支持自然语言处理
* 支持图片文件结构化（OCR）
* 支持解析器节点横向扩展
* 实现数据去重
* 预留插件接口，根据接口编写插件对特定源数据解析
* 实现解析结果日志、错误日志和报警日志
* 实现解析规则配置、日志配置、调度配置和容错策略配置

## 主要对象

### 监控器

监控器，所有解析器都由监控器统筹，当解析器出现错误时会根据配置的策略自动重启解析器，重启一定次数后仍没有解析完成的生成报警日志。

### 解析器

解析器分为系统插件和第三方插件。系统插件为HTML、JSON、PDF、Image和Binary类型文件针对不同数据源的一些标准解析器。第三方插件为运营人员提供接口，用来开发一些高度定制的解析器。

对于特定类型的解析器会根据不同数据源开发一些标准的插件，比如HTML解析器提供工商网站、百度和谷歌的标准解析插件等。

解析器相当于监控器的工人，被分配完成特定的解析任务，比如HTML页面提取某几个字段的值并存入目标数据库，一旦发生错误在崩溃前必定会记录错误日志并通知监控器。

### 去重器

去重器解决了相同数据源多次结构化消耗系统性能的问题。通过配置去重规则实现数据去重，比如可以配置MD5或Sha-256值来判定是否为同一份数据，对于同一份数据可以配置数据失效时间来更新数据。

## 设计图

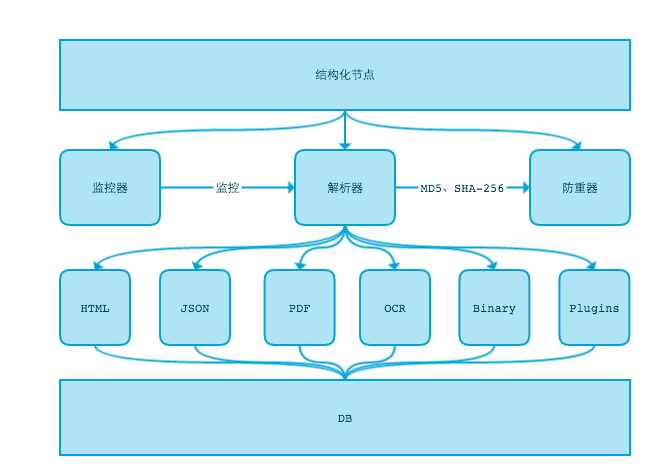


图4-1 结构化设计图

## 流程图

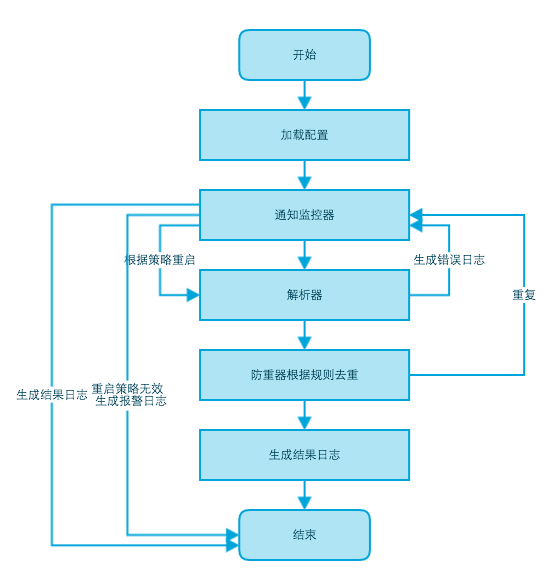


图4-2

# 存储层

## 设计目标

* 分业务、分片存储
* 容量可以扩展到P级
* 可水平扩展
* 便于检索

## 设计图



## 主要对象

### MongoDB

定义为：MongoDB的集群服务

### MySQL

定义为：MySQL的集群服务，主要用于存储配置类数据。

### NFS

定义为：类似本地磁盘的网络文件系统集群

### 小文件

定义：大小在(0, 100M)的二进制的文件

### 大文件

定义：大小在(100M, +)的二进制的文件

### 文件索引

定义：记录文件位置的对象。包含如下属性：

1. ID。全局的唯一主键
2. 存储介质。MongoDB或者是NFS
3. URI。访问路径
4. 添加时间
5. 文件字节数
6. 来源主机IP
7. 数据内容MD5值

## 流程图



# 系统层

## 设计目标

## 主要对象

## 设计图

## 流程图