**网站检查系统的概要设计**

**说明：本文以爬虫角度对系统的整体设计进行简单介绍。**

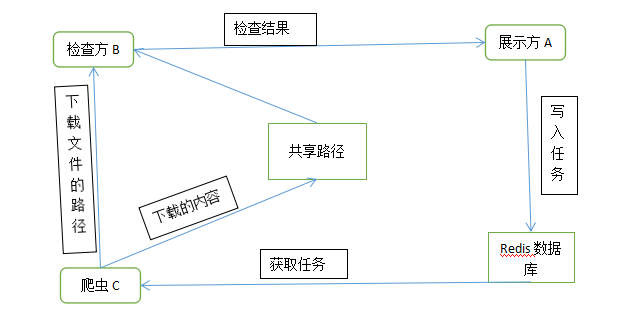
1. 组成

1.结果展示和下达检查任务方：A

2.执行检查方：B

3.数据收集方（爬虫）：C

1. 设计图



1. 流程

A下达检查任务（请查看task\_descriptor.txt）

C执行任务，下载网页、文档、图片

B对下载后的文件进行关键字检查

1. 交互关系

A 与 C：

1. 共享redis数据库的任务列表，A通过redis下达任务
2. C在爬取的过程中实时报告下载的数量和是任务状态

B 与 C：

1. C将收集的检查材料放到共享目录中
2. C向B发送消息，B开始检查，消息内容请查看task.proto

A 与 B

1. 共享Redis，A向B实时报告检查数量和检查状态
2. 共享MySQL，A将违规信息存储到数据库中

（完）

**爬虫内部设计**

从redis获得任务

SpiderTaskDistributor

分发任务

SpiderTask

Receiver

SpiderTask

Receiver

SpiderTask

Receiver

SpiderTask

Receiver

Controler

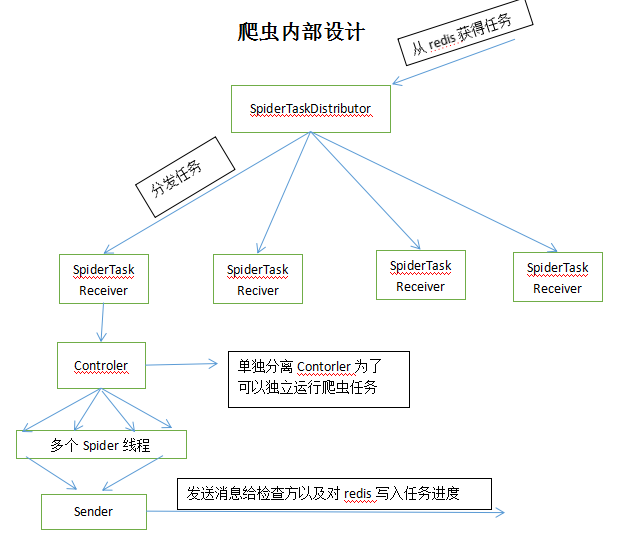
单独分离Contorler为了

可以独立运行爬虫任务

多个Spider线程

发送消息给检查方以及对redis写入任务进度

Sender



**声明：本说明只对网站爬虫进行了介绍，微博、微信、邮件检查功能不在此介绍。但其实现与网站爬虫思路相近，只是替换文件Spider.py和SpiderController.py。**

**一．环境**

运行环境：win8.1企业版 64位

开发工具：python 2.7.12 32位

第三方库：protobuf、zmq、redis

下载工具：pip可以用于下载zmq、redis

**二．配置环境**

**一．安装protobuf**

1. protobuf-python-3.0.0-beta-3.zip和protoc-3.0.0-beta-3- win32.zip（本资源已经放在压缩包里）

2. 解压protoc-3.0.0-beta-3-win32.zip得到protoc.exe

解压protobuf-python-3.0.0-beta-3.zip 得到 protobuf-3.0.0-beta-3文件夹

3. 将protoc.exe复制到protobuf-3.0.0-beta-3/src中，并将 protoc.exe的路径添加到环境变量中

4.打开cmd命令框，通过cd命令，进入目录 protobuf-3.0.0-beta-3/python下

先执行 python setup.py build

再执行 python setup.py install

注意：protobuf库需要依赖某些模块，例如：我在其他电脑上运行程序时报错：No module named six，需要pip install six 这视具体情况而定。

**关于生成.proto对应的.py文件**：

1. 编写proto文件
2. 用protoc.exe生成py文件，cmd命令：

protoc -I=$SRC\_DIR --python\_out=$DST\_DIR $SRC\_DIR/xx.proto

在DST\_DIR目录下会生成xx\_pb2.py

1. **安装pip**
2. pip-8.1.2.tar.gz （本资源已经放在压缩包里）
3. 解压后在解压后的文件夹中找到setup.py 文件
4. 打开cmd命令框，通过cd命令进入到此目录下。
5. 执行 python setup.py install
6. 将python2.7.12/Scripts加入环境变量
7. **安装zmq**

**重新**打开一个cmd，执行pip install zmq即可。新开cmd是因为：安装pip的cmd命令窗口不能找到刚刚更新的pip环境变量。

1. **安装python的redis库**

打开cmd，执行pip install redis 或者easy\_install redis (easy\_install 在安装pip时会被同时安装)

**三．爬虫具体实现：**

**\*需要详细说明的是Spiders.py:**

主要由两个类实现功能：URLSpider、DownloadSpier

**URLSpider:**

1. 负责下载网页页面并保存到本地
2. 分析页面中的全部链接，分类成网页链接和文件链接，并加入到URLSpider和DownloadSpier的任务队列中。
3. 下载网页，将记录写入conf文件
4. 细节要求：能够做链接过滤，保证任务队列中的链接是域内链接。

**DownloadSpider:**

1. 从自身的任务队列中获取任务
2. 对任务进行分类处理
3. 下载文档或者图片

**一． 7个功能实现文件：**

1. Spiders.py: 爬虫的具体功能实现
2. SpiderControler.py: 设置爬虫参数，启动爬虫
3. SpiderTaskReceiver.py: 接收Distributor分发的任务
4. SpiderTaskDistributor.py: 接收任务分发给Receiver
5. Sender.py: 与检查端通信
6. Task\_pb2.py: 爬虫与检查方之间的通信数据
7. DistributorMessage\_pb2.py:任务分发与任务接收端之 间的通信数据

**二．2个json配置文件：**

1. distributor\_setting.json：

内容：检查端ip、port、数据库redis的ip、port、

任务列表的名称、保存路径

2.receiver\_setting.json：

内容：任务分发端的ip、port

**三．2个proto格式文件**

1. DistributorMessage.proto：distributor发送给receiver的消息内容及格式的设置

2. Task.proto：爬虫发送给检查端的消息内容及格式的设置

**四．2个测试文件：**

1. Checker\_Emulate\_Test.py：仿真检查端，用于测试自查
2. Shower\_Emulate\_Test.py：仿真展示端，向redis写任务

**五．3个辅助工具文件**

1. Build\_Json\_File.py：生成json配置文件

2. Default\_Encoding\_Test.py：测试系统的默认编码

3. Find\_Path.py：用于拼接json配置文件的绝对路径

**六．3个说明文档：**

1. task\_descriptor.txt：展示端向redis写入任务的具体内容

2. Task设计说明.doc：介绍Task.proto的详细设计思路

3. 本文档：整个系统的概要设计和对爬虫的设计介绍

**如何运行**：

1. 独立运行模式

可以直接运行SpiderControler.py ，参数在此文件内部设置

打开cmd执行 python SpiderControler.py

实现的功能：下载网页、图片、附件

1. 整体运行模式
2. 检查端运行，等待爬虫传递任务
3. 根据实际情况修改json配置文件 (上文已经介绍)
4. 打开cmd，执行python SpiderTaskDistributor.py
5. 打开cmd，执行python SpiderTaskReceiver.py
6. 展示端运行，将任务写入redis数据库
7. 补充：步骤1、3、4、5的启动顺序允许任意打乱，因为采用的是阻塞模式，步骤2必须在3、4之前
8. 模拟测试

消除检查端和展示端不在线或未开发完成的影响，模仿检查方和展示方使程序照常运行，实现单独测试

(注意：模拟测试同样需要在本地或远程计算机预先安装redis数据库)

1. 打开cmd，执行python Checker\_Emulate\_Test.py
2. 新开cmd，执行python Shower\_Emulate\_Test.py
3. 根据实际情况修改json配置文件
4. 打开cmd，执行python SpiderTaskDistributor.py
5. 打开cmd，执行python SpiderTaskReceiver.py
6. 补充：步骤1、2、4、5的启动顺序允许任意打乱，

允许这四个文件运行在四台不同的电脑上，也可以都运行在同一台电脑上。同时，SpiderTaskReceiver.py可以同时在多台电脑上运行，每台电脑会独立运行一个单独的任务，实现简单的分布式爬虫。