一种可视化爬虫技术的分享

石恩名 shikanon@live.com

广州优亿科技有限公司

Background

Pyspider

Scrapy 1.1

Requests

對比
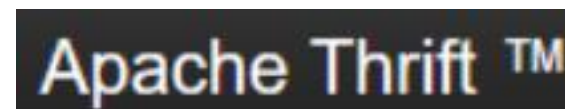
Beautiful Soup

好心塞！

More...

Demand

We need...

Demand

- Simple

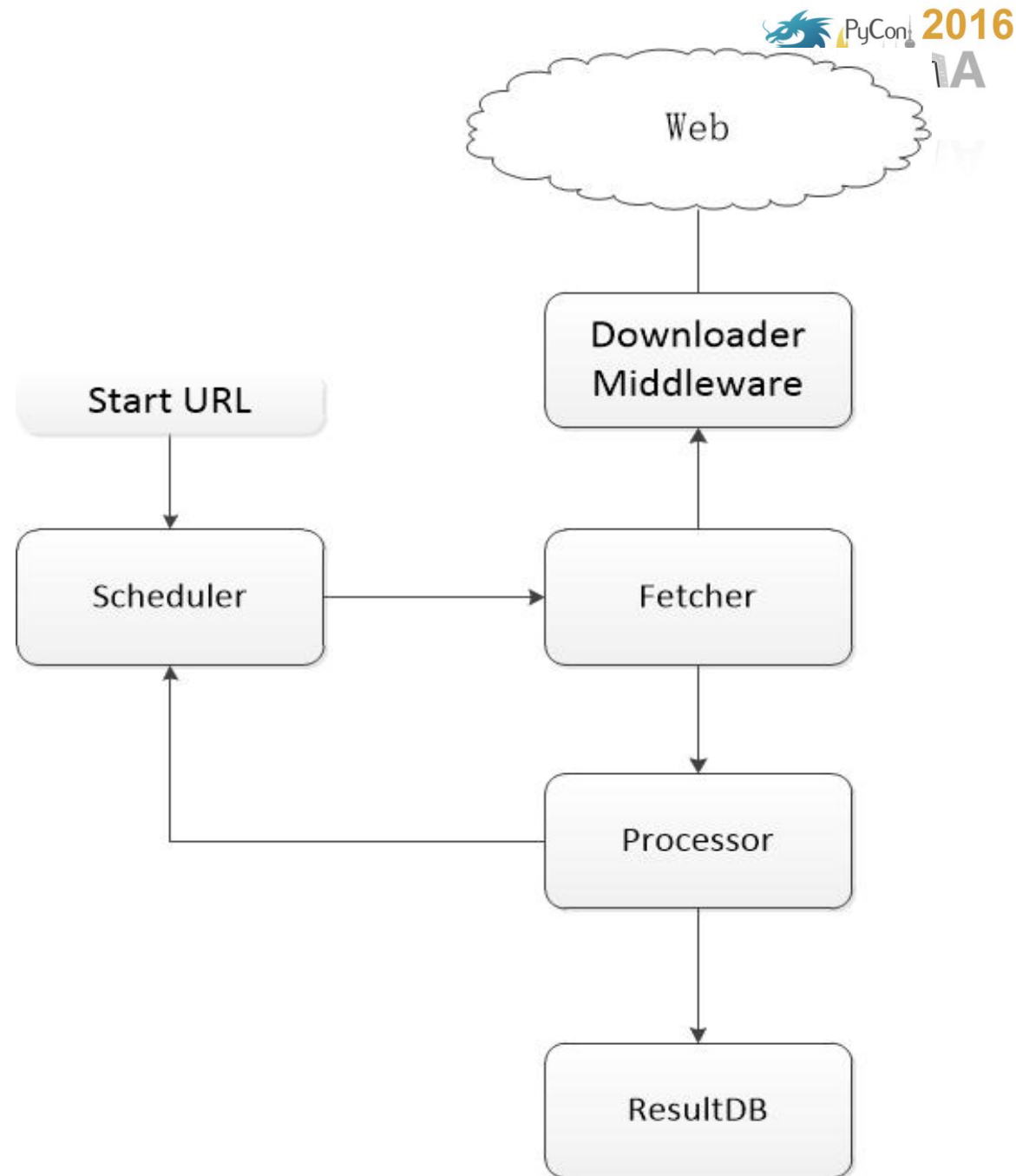- Convenient

- Stable

- High performance
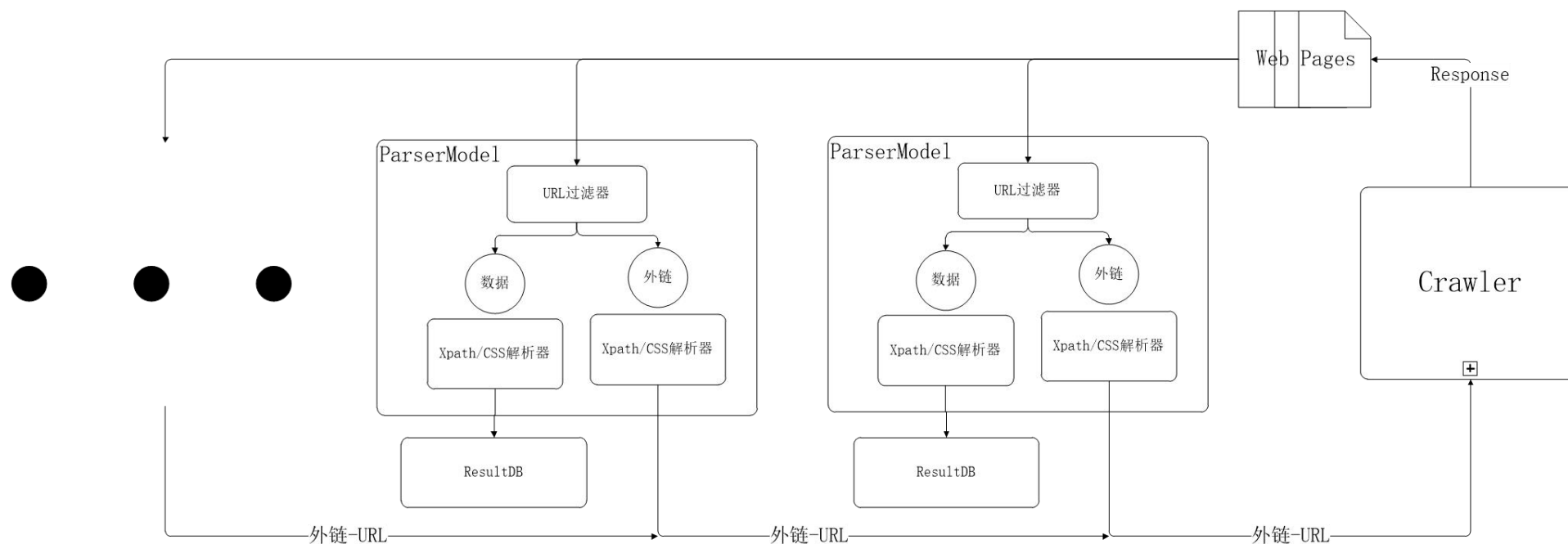
前端
技术栈

后端
技术栈

Crawler

- Scheduler:
  - Bloom filter
- Fetcher:
  - Tornado、PhantomJS
- DownloaderMiddleware:
  - Proxy、Cookies、User-Agent
- Processor:
  - ParserModel、cx-extractor
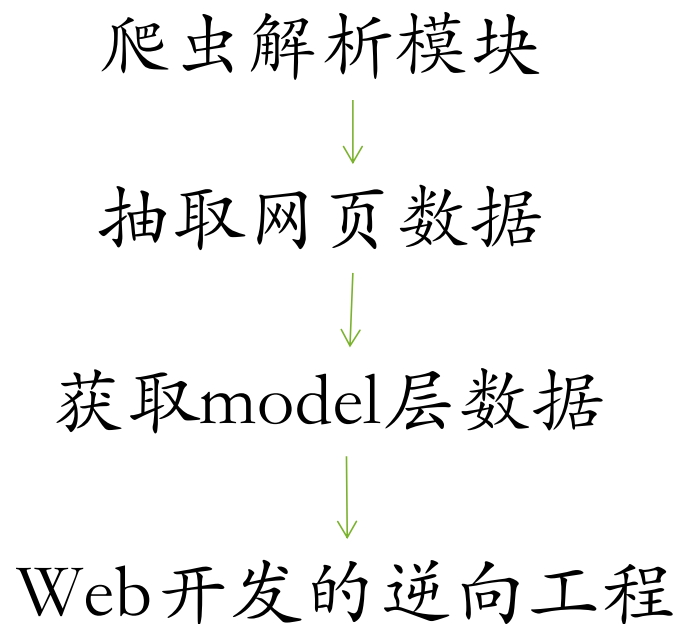- Database:
  - Redis、MongoDB

Crawler Visual Logic

- Start URL (入口)

- Parser Config (解析规则文件)

- Crawler Config (采集周期、JS模拟、Cookies等)

- 常见的Web开发模式：MVC、MVP、MTV、MVVM ......

爬虫解析模块

↓

抽取网页数据

↓

获取model层数据

↓

Web开发的逆向工程

- 自古爬虫和Web开发是一家...

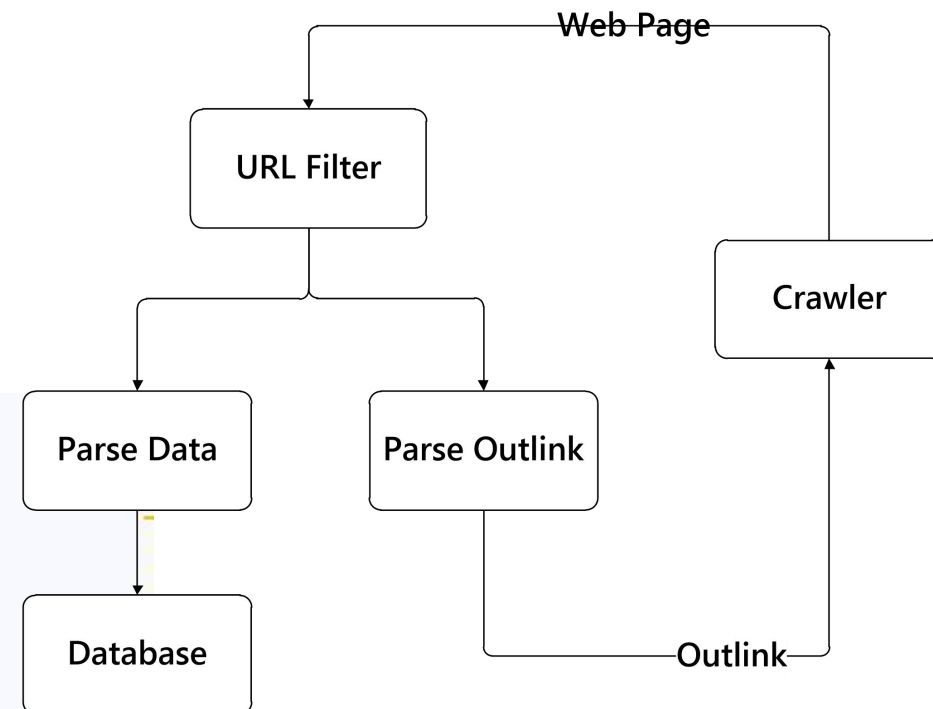# ParserModel

```
489              self.python_script, url) for url in urls]
490         return urls, outlink_attrs
491
492     def parser_model(self, xml):        ← 导入配置文件
493         '''
494         解析模块，将xml-pattern文件解析成配置文件
495         '''
496         try:
497             soup = BeautifulSoup(xml, "xml")
498         except Exception:
```

```
503
504         # 解析
505         for website in websites:
506             fields_result = []
507             outlinks_result = []
508             # URL匹配,不匹配则跳过            URL匹配
509             if len(self.url_filter(website)):
510                 continue
511             # 数据解析
512             data_object = website.find("data-object")
513             if data_object:
514                 fields = data_object.findAll("field")
515                 fields_result = [self.get_field(field) for field in fields]        数据解析
516             # 链接解析
517             outlink = website.find("outlinks")
518             if outlink:                                                           外链解析
519                 entities = outlink.findAll("entity")
520                 outlinks_result = [self.get_outlink(entity) for entity in entities if self.get_outlink(entity)]
521             yield fields_result, outlinks_result
522
```

Web Page

URL Filter

Crawler

Parse Data

Parse Outlink

Database

Outlink

ParserModel

- ## URL Filter

  - regular match、HTML DOM Similarity

- ## Parse Data

  - CSS selector、XPath parser、regular match expression、character slice、text extractor...

- ## Parse Outlink

  - CSS selector、XPath parser、URL normalization、Javascript extractor

URL Filter

Django路由规则
匹配

```
8    urlpatterns = [
9        url(r'^projects/$', project.projects),
10       url(r'^project/(\d+)/$', project.project_detail),
11       url(r'^project/(\d+)/templates/$', project.templates),
12       url(r'^project/(\d+)/template/(\d+)/$', project.template_detail),
13       url(r'^project/(\d+)/tasks/$', project.tasks),
14       url(r'^project/(\d+)/img/(\d+)/$', project.img),
15       url(r'^project/(\d+)/performance/', project.performance),
```

一个逆向的MVC框架...

daily.zhihu.com/story/8819156
daily.zhihu.com/story/8822772

逆向

http://daily.zhihu.com/story/\d+

基于**HTML DOM结构树**的**相似性**判别

```
1  <!doctype html>
2
3
4
5  <html lang="zh-CN">
6  <head>
7  <meta charset="utf-8">
8  <meta http-equiv="X-UA-Compatible" content="IE=edge,chrome=1">
9  <title>让孩子参与照顾临终老人，会不会太残酷？</title>
10 <meta name="apple-itunes-app" content="app-id=639087967, app-argument=zhihudaily://story/8794980">
11 <meta name="viewport" content="user-scalable=no, width=device-width">
12 <link rel="stylesheet" href="/css/share.css?v=5956a">
13
14
15 <script src="http://static.daily.zhihu.com/js/modernizr-2.6.2.min.js"></script>
16 <link rel="canonical" href="http://daily.zhihu.com/story/8794980"/>
17 <base target="_blank">
18 <script type="text/javascript">
19 if (localStorage && localStorage.getItem('hideDownloadBanner') !== 'true') {
20 document.documentElement.className += ' show-download-banner';
21 }
22 </script>
23 </head>
24 <body>
25
26 <div class="global-header">
27 <div class="main-wrap">
```
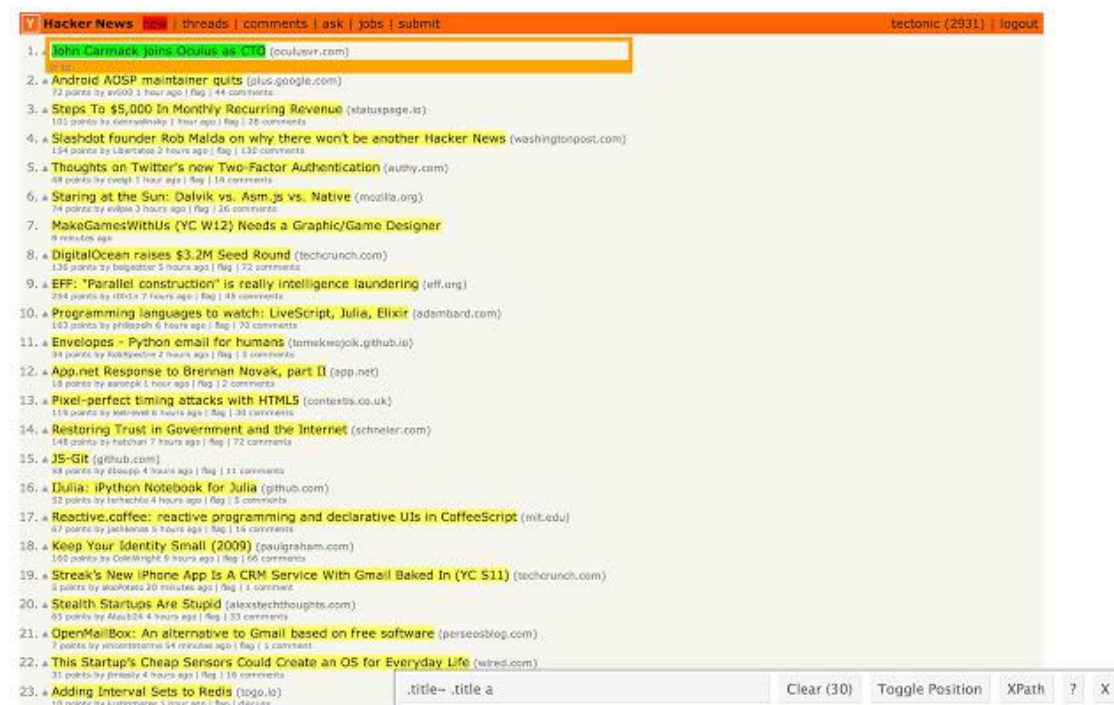
```
1  <!doctype html>
2
3
4
5  <html lang="zh-CN">
6  <head>
7  <meta charset="utf-8">
8  <meta http-equiv="X-UA-Compatible" content="IE=edge,chrome=1">
9  <title>水是生命，卫生是尊严，那这水费，该怎么收才合理？</title>
10 <meta name="apple-itunes-app" content="app-id=639087967, app-argument=zhihudaily://story/8792845">
11 <meta name="viewport" content="user-scalable=no, width=device-width">
12 <link rel="stylesheet" href="/css/share.css?v=5956a">
13
14
15 <script src="http://static.daily.zhihu.com/js/modernizr-2.6.2.min.js"></script>
16 <link rel="canonical" href="http://daily.zhihu.com/story/8792845"/>
17 <base target="_blank">
18 <script type="text/javascript">
19 if (localStorage && localStorage.getItem('hideDownloadBanner') !== 'true') {
20 document.documentElement.className += ' show-download-banner';
21 }
22 </script>
23 </head>
24 <body>
25
26 <div class="global-header">
27 <div class="main-wrap">
```

Parse Data

- lxml

- PyQuery

- Thrift

- cx-extractor (Java)

- SelectorGadget (javascript插件)

Parse Data

```
126
127     def xpath_parser(self, expression):
128         '''xpath解析方法'''
129         try:
130             express_result = self.etree.xpath(expression)
131         except:
132             logging.error(u"xml解析文件存在错误!xpath表达式存在错误.")
133             raise AttributeError("xml_file is Error!")
134         return express_result
135
136     def css_parser(self, expression):
137         '''css解析方法'''
138         query_tree = PyQuery(self.etree)
139         try:
140             express_result = query_tree(expression)
141         except:
142             logging.error(u"xml解析文件存在错误!css表达式存在错误.")
143             raise AttributeError("xml_file is Error!")
144         return express_result
145
146     def text_extractor(self, content):
147         '''解析网页正文'''
148         extractor = thriftpy.load(u"../classifier.thrift",
149                                    module_name="classifier_thrift")
150         classifier_client = make_client(extractor.Classifier, '192.168.73.1', 8090)
151         return classifier_client.extractor(content)
152
```

利用lxml的xpath方法解析

CSS选择器调用了PyQuery

cs_extractor的jar包运算结果通过thriftpy通信传递

Parse Outlink

- lxml

- PyQuery

- werkzeug

```
>>> url_fix(u'http://de.wikipedia.org/wiki/Elf (Begriffsklärung)')
'http://de.wikipedia.org/wiki/Elf%20%28Begriffskl%C3%A4rung%29'
```

- URL Filter
  - 机器学习：通过选择两个页面，计算HTML DOM相似度，再通过多个相似页面计算URL正则表达式，构建判别器。

Page A → DOM Similarity → URL 正则表达式 → 判别器

Page B

Prospect
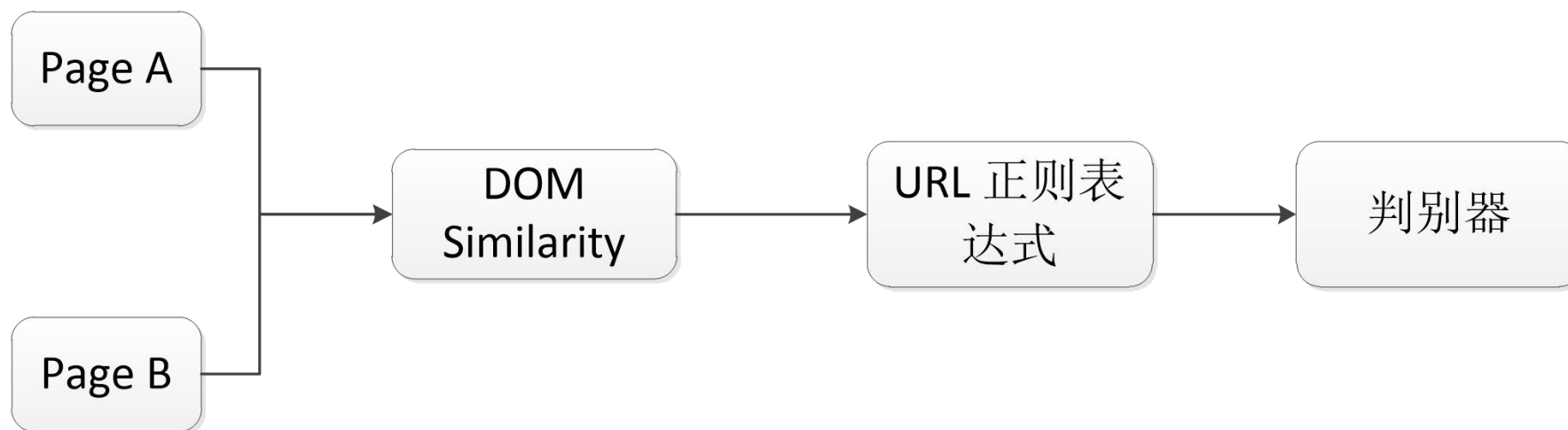
- Parse Outlink
  - 桌面插件：通过对浏览器Hooking截获javascript执行代码块。

点击下一页

↓

__doPostBack('lbNextPage','')

↓

PhantomJS执行
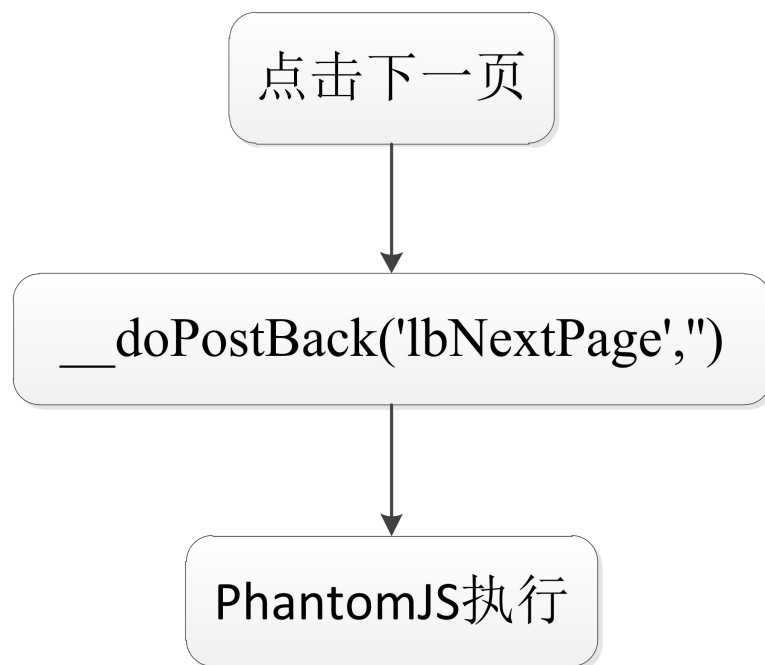
可行性探讨？

Database Choice and Test

- 测试环境：
- 机器：Intel(R) Xeon(R) CPU E5-2620 0 @ 2.00GHz
- 内存：28G

采集速率(page/min)

Product Show

- 网址：http://spiderman.useease.com:9000/

谢谢观看

Luck_Sugar

shikanon

2966950857