

基于搜索引擎 Web API 和模拟请求方式的搜索服务研究

余文斌, 曾夏玲, 周清清, 余敏

(江西师范大学 计算机信息工程学院, 江西 南昌 330022)

摘 要:传统的基于网页方式的搜索引擎服务不能很好地满足用户个性化的需求, 搜索引擎 Web API 概念的提出较好地解决了这一问题。文中讨论了非商业网站如何使用搜索引擎 Web APIs 实现网站搜索服务。为避免搜索引擎 Web APIs 在搜索次数上的限制, 给出一种不使用 Search APIs 而直接通过模拟请求方式得到搜索结果的新方法。

关键词: Web API; SOAP; Ajax; 模拟请求; 搜索服务

中图分类号: TP391.3

文献标识码: A

文章编号: 1673-629X(2008)03-0114-04

Research of Search Service Based on Search Engine Web API and Simulating Request

YU Wen-bin, ZENG Xia-ling, ZHOU Qing-qing, YU Min

(College of Computer Information Engineering, Jiangxi Normal University, Nanchang 330022, China)

Abstract: The traditional way of search engine services based on webpage fail to meet the personalized demand of the user well, the concept of search engine Web API provides a better solution to the problem. Discusses how to use search engine Web APIs to implement search service of the non-commercial website. To avoid the limit of the search number, gives a new method to get search result by using the way of simulating request.

Key words: Web API; SOAP; Ajax; simulating request; search service

0 引言

随着 Internet 日新月异的发展, 人们的生活越来越多地依赖于网络上的海量信息。那么, 如何从海量信息中找到想要的信息就变得至关重要。在各大搜索巨头的努力下, 基于关键词的搜索已经取得了令人满意的结果。但是对于小型非商业网站, 也通过自主研发的方式开发出类似功能的搜索服务, 无疑是给网站带来了资金和技术方面的困难。所以, 在遵守相关授权合约的情况下利用已有搜索引擎^[1]的搜索结果完成网站的搜索服务, 不失为一个好的解决办法。

文中基于这一目的, 讲解了利用现有搜索引擎开发站内搜索的两种方法: 利用搜索引擎方提供的多种 Search APIs 完成自身的搜索; 为了避免 Search APIs 在

搜索次数上的限制, 直接采用模拟请求方式得到结果网页, 并通过进一步的页面分析得到需要的结果。

1 搜索引擎 Web API 服务及使用

1.1 Web Service

Web Service 是连接在互联网上的各种服务, 它用一种新的方式让公司和个人共享信息。简单来说, 它就是一个应用程序, 向外界暴露出一个能够通过 Web 进行调用的 API, 开发者只需将它们的请求通过接口直接发送给在公司网站后端运行的程序就可以得到想要的结果。Web Service 是完全基于标准的技术, 它的通信标准是独立于平台、组建模型和编程语言的^[2,3]。无论应用程序是使用何种编程语言而写, 运行在何种操作系统之上, 一个应用程序应该可以很方便地访问另外的应用程序。

搜索引擎 Web API 就是一种 Web Service 服务, 它允许用户在客户端调用接口函数完成查询工作, 查询条件和结果被作为接口函数参数在服务器端和客户端之间传递。目前使用的比较多的搜索引擎 Web APIs

收稿日期: 2007-06-01

基金项目: 国家自然科学基金资助项目(60363002); 江西省分布计算工程技术研究中心开放基金资助项目

作者简介: 余文斌(1983-), 男, 江西南昌人, 硕士研究生, 研究方向为信息安全、数据挖掘; 余敏, 教授, 硕士生导师, 研究方向为信息安全、分布式系统与移动计算技术。

主要有 Google Search APIs 和 Yahoo! Search APIs。这里主要以 Google Search APIs 为例讲解,有关 Yahoo! Search APIs 详细内容可以参阅 Yahoo! Developer Network^[4]。

Google Search APIs 提供的检索服务经历了两个阶段,在第一阶段提供的是基于 SOAP 架构的 API 服务,第二阶段提供的是基于 Ajax 技术的 API 服务。

1.2 基于 SOAP 框架的 Google Search APIs 服务

1.2.1 SOAP

SOAP(Simple Object Access Protocol,简单对象访问协议)是在非集中、分布式环境中交换信息的轻量级协议^[5]。它是基于 XML 的协议,包括三个部分:封套(envelope)定义了消息内容和处理的框架、一套编码规则用来表达应用定义数据类型的实例以及表达远程过程调用和响应的协定。它采用了已经广泛使用的两个协议:HTTP 和 XML,HTTP 用于实现 SOAP 的 RPC 风格的传输,而 XML 是它的编码模式。

1.2.2 使用 Google SOAP Search APIs 服务

为了方便更多用户能够使用到 Google 所提供的服务,Google 针对搜寻机制提供十分完整的 API,通过使用 API 来将它搜索的服务整合到自己的应用程序中。目前的 Google Web API 是符合 Web Services 的架构,只要依照 WSDL(Web Services Description Language)^[6]描述性文档的定义,以 SOAP 的方式呼叫取得搜寻结果即可。这样,只要程序语言本身可以撰写出 SOAP Client 就可以使用。若采用 Java 为开发语言,Google Web API 提供已经封装好的 SOAP Client 链接库,通过简单的程序就可以获取 Google 的服务。

Google Web API 中提供以下五个类^[7]: GoogleSearch, GoogleSearchDirectoryCategory, GoogleSearchFault, GoogleSearchResult, GoogleSearchResultElement。GoogleSearch 类主要封装了通过 SOAP 对 Google Web APIs 的访问以及 cached 页面的访问函数;GoogleSearchDirectoryCategory 类用于记录网页目录类别;GoogleSearchFault 类提供了异常处理的功能;GoogleSearchResult, GoogleSearchResultElement 则提供了对搜索结果的进一步操作功能。下面通过代码示例来演示它的使用:

```
GoogleSearch s = new GoogleSearch();
//设置开发者的 API keys
s.setKey(clientKey);
//设置查询关键字
s.setQueryString(QueryKey);
.....
//设置当前查询的起始下标
s.setStartResult(start);
```

```
GoogleSearchResult r = s.doSearch();
GoogleSearchResultElement[] re = r.getResultElements();
...//对查询结果进行处理并相应输出
```

通过上述示例代码的阅读,可以发现获取 Google 的强大服务,仅需要完成类似的简单编程即可实现,首先初始化一个 GoogleSearch 对象,然后设置使用者申请的账户密码以及查询关键词,即可进行查询。获取到每次的查询结果后,根据不同的需要对结果进行相应处理。

1.3 基于 Ajax 的 Google Search APIs 服务

1.3.1 Ajax 简介

Ajax 是 Asynchronous JavaScript and XML(以及 DHTML 等)的缩写,由 HTML、JavaScript™ 技术、DHTML 和 DOM 组成,它将原来笨拙的 Web 界面转化成交互性的 Ajax 应用程序^[8]。

Ajax 的核心是 JavaScript 对象 XMLHttpRequest,该对象是微软公司为了满足开发者的需要,在 1999 年的 IE5.0 浏览器中率先推出的。它允许客户端脚本执行 HTTP 请求,并解析 XML 服务器响应。它是一种支持异步请求的技术,XMLHttpRequest 为运行于浏览器中的 JavaScript 脚本提供了一种在页面之内与服务器通信的手段。页面内的 JavaScript 可以在不刷新页面的情况下从服务器获取数据,或者向服务器提交数据。简而言之,XmlHttpRequest 使您可以使用 JavaScript 向服务器提出请求并处理响应,而不阻塞用户。

1.3.2 使用 Google Ajax Search APIs 服务

Google 在 2006 年的 12 月 5 号正式停止了 SOAP Search API key 的申请,所以在此之后,如果没有预先申请 SOAP Search API key 的程序员就不能使用 SOAP Search 服务了,取而代之的是 Google 提供的 Google Ajax Search APIs。

Google Ajax Search APIs 提供了大量的搜索服务,包括:web, blog, news, book 以及 map 搜索。那么它的相关类大致包括上述几种搜索的 Searcher 类以及 Result 类,这里通过 web 搜索示例来了解它的基本使用:

```
<script
src="http://www.google.com/uds/api? file = uds.js& v =
0.1&key=your key"
type="text/javascript"></script>
<script type="text/javascript">
var gWebSearch;
function OnLoad()
{
//Initialize the web searcher
gWebSearch = new GwebSearch();
```

```

//Set Search resultSet size.
gWebSearch. setResultSetSize ( GSearch. LARGE_ RESULT-
SET);
//Set callback function
gWebSearch. setSearchCompleteCallback(null, OnWebSearch);
//Set Query key
gWebSearch. execute("mark");
}

function OnWebSearch()
{
if (! gWebSearch. results) return;
var searchresults = document. getElementB
yId ("searchresults");
searchresults. innerHTML = "";
var results = "";
for (var i = 0; i < gWebSearch. results. length; i++) {
var thisResult = gWebSearch. results[i];
results += "<p>";
results += "<a href = \" " + thisResult. url + "\">" + this-
Result. title + "<V a><br V>";
results += thisResult. content + "<br V>";
results += "<span class = \"url \">" + thisResult. url + "<
V span>";
if (thisResult. cacheUrl)
{
results += "<a class = \"cached \" href = \" " + thisRe-
sult. cacheUrl + "\">Cached <V a>";
}
results += "<V p>";
}
searchresults. innerHTML = results;
}
</script>
</head>
<body onload = "OnLoad()">
<div id = "searchresults"></div>
</body>

在整段代码中, 没有发现 Ajax 核心对象 XMLHttpRequest 的存在。但是搜索结果显示的确没有页面重新装载过程, 具备了 Ajax 应用程序的标志之一, 即无需重新装载页面, 直接显示内容更改。那么, XMLHttpRequest 在哪里呢? 另外除了 getElementById () 方法, DOM 和页面操作又在哪里? 实际上, 这一切都包含在下面几行代码中:
<script
src = "http://www. google. com/uds/api? file = uds. js&v =
0.1&key = your key"
type = "text/javascript"></script>

```

通过调用 Google 提供的 uds. js 文件, 就可以很容

易地获得 Google 的服务了。其中 key 需要用户在 Google 申请一个 Ajax API key。另外代码中的设置回调函数部分也可以看到 Ajax 的影子, 通过它的使用, 可以在发送查询请求后, 继续完成页面上的其它工作而不是等待服务器的响应。等服务器方返回查询的结果, 反方向调用回调函数, 从而实现了 Ajax 的异步。

2 模拟请求方式获取搜索服务

为了让搜索资源能有效分享给更多的人, 搜索公司提供的 API 均有不同程度的限制。例如: Google API 针对每个申请者限制每日仅能呼叫 1000 次, 而且每次的查询结果仅回传前 1000 笔, 回传则以最多 10 笔为一个单位。因此, 为了不受此约束可以采用模拟请求的方式获取搜索服务, 如图 1 所示。

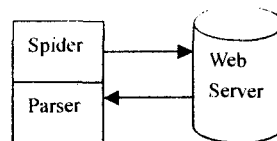


图 1 模拟请求方式搜索服务

如图所示, 先通过一个 Spider 对象向 Web Server 发送一个查询请求, 查询完成后即返回一个包含查询结果的 Html 页面, 然后将结果传给 Parser 对象, 对页面进行处理得到查询结果中标题、内容摘要以及 cache 页面链接。用户可以针对自己的需要对 Parser 后的结果再加工, 满足用户个性化需求, 例如下面的代码示例:

```

<? php
include "Crawler. class. php";
include "Parser. class. php";
//write the keywords here for search $ keywordsGot = "Cricket";
echo "Search Results for <b>". $ keywords. "</b><br>";
$ pageNeed = 1; //enter here how many pages you want to crawl
for( $ n = 0; $ n < $ pageNeed; $ n++)
{
echo "In Page ---<b>". ($ n + 1). "</b><br>";
$ spider = new Crawler( $ keywordsGot, $ n);
$ parser = new Parser( spider. getResult());
$ parser -> getDivs();
$ links = $ parser -> getLinks();
$ titles = $ parser -> getTitles();
$ descriptions = $ parser -> getDescriptions();
for( $ i = 1; $ i < (count( $ links)) + 1; $ i++)
{
echo "<div><a href = \" ". $ links[ $ i]. "\">". $ titles
[ $ i]. "</a><br>";
echo $ descriptions[ $ i]. "</div><br>";
}
}

```

1? >

至此,基本的模拟请求式搜索服务模型已经完成。但是基本模型还存在以下两个方面问题:

1)网络服务具有不稳定性。如果将请求搜索 Web Server 局限在一家,必然出现偶尔的搜索失败。

2)搜索内容的有限性。根据 2005 年搜索网站 Dogpile.com 联合美国匹兹堡大学和宾夕法尼亚州立大学研究人员所进行的一项调查结果^[9]显示,Ask Jeeves, Google 和 Yahoo 三大搜索引擎的搜索结果大相径庭。研究人员对随机抽取的 10316 个关键词的搜索记录进行了比较,结果发现在这三大搜索引擎总数为 336232 个的搜索结果中只有 10712 个相同,即只有 3% 相同。另外在某两个搜索引擎的结果中只有 12% 的搜索结果相同,而其余 85% 的搜索结果则是各搜索引擎独有的。

针对上述问题,网站的搜索服务推出了两项扩展功能:

(1)普通搜索,默认搜索 Google,仅当出现搜索失败时尝试其他的搜索服务。

(2)全面搜索,将百度、Google 以及 Yahoo 的搜索结果去同存异,丰富我们的搜索。

实现这两项服务,需要对之前的搜索结果处理程序 Search 类、Parser 类以及 Spider 类进行重构。这里采用 Strategy 模式,对其进行扩展,如图 2 所示,其中 GParser, BParser, YParser 分别是根据 Google, Baidu, Yahoo 网页格式生成的 Parser 类,而 GQUrl, BQUrl, YQUrl 则是根据三大网站的请求格式生成的搜索关

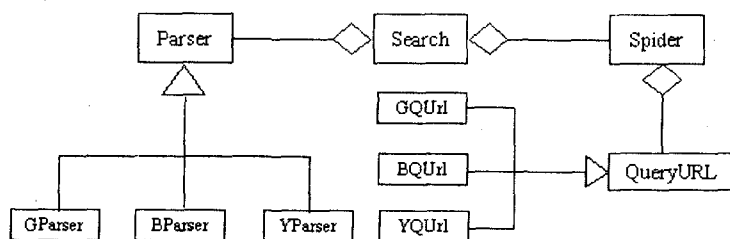


图 2 扩展的模拟请求式搜索服务

键词的 URL。

3 结束语

文中根据小型网站的搜索需求,给出了网站搜索服务的两种实现方式——基于搜索引擎 Web API 和模拟请求方式的搜索服务实现。针对每种实现方法给出了相关的示例,并根据真实的网络情况给出了两种扩展的模拟请求式搜索服务。通过使用扩展的搜索服务,提高了传统搜索引擎的查准率和查全率,较好地满足了用户的查询需求。

参考文献:

- [1] 张卫丰,徐宝文. Web 搜索引擎框架研究[J]. 计算机研究与发展,2000,37(3):376-378.
- [2] 杨涛,刘锦得. Web Services 技术综述——一种面向服务的分布式计算模式[J]. 计算机应用,2004,24(8):1-4.
- [3] 唐飞龙,李明禄,曹健. 一个 Web 服务事务处理模型:结构、算法和事务补偿[J]. 电子学报,2003,12(12A):2074-2078.
- [4] Yahoo! Developer Network Home[EB/OL]. 2002-10. <http://developer.yahoo.com/>.
- [5] Mitra N. Simple Object Access Protocol (SOAP) 1.2. W3C working draft[DB/OL]. 2002-06. <http://www.w3.org/TR/WCAG20/>.
- [6] Christensen E. Web Services Description Language (WSDL) 1. IBM/Microsoft Joint Working Document[DB/OL]. 2002-10. <http://www4.ibm.com/software/developer/library/wwsdl.html?dwzone=web/>.
- [7] Google Web API Reference[EB/OL]. 2004-10. <http://www.google.com/apis/>.
- [8] Garrett J J. Ajax: A New Approach to Web Applications[EB/OL]. 2005-02. <http://www.adaptivepath.com/publications/essays/>.
- [9] A Research Study by Dogpile.com: Different Engines, Different Results[EB/OL]. 2005-06. <http://comparesearchengines.dogpile.com/OverlapAnalysis.pdf>.

(上接第 113 页)

范大学出版社,1989.

- [3] 曾天雄,孙月红. 马克思主义哲学教程[M]. 北京:北京工业大学出版社,2004.
- [4] 陈昌曙. 自然辩证法概论新编[M]. 沈阳:东北大学出版社,2004.
- [5] 乐晓波,汪琳. 面向对象的 Petri 网建模技术的研究[J]. 计算机工程,2002,28(5):86-88.
- [6] 乐晓波,唐贤英. 基于 Petri 网建模的资源调度蚁群算法[J]. 计算机技术与发展,2006,16(1):44-46.

- [7] 郝东,蒋昌俊. 基于 Petri 网于 GA 算法的 FMS 调度优化[J]. 计算机学报,2005,28(2):201-208.
- [8] 周卫东,杨加敏. 一种 Petri 网结合遗传算法的优化方法及应用[J]. 山东大学学报:工学版,2005,35(4):59-67.
- [9] 维尔·杜兰特. 哲学的故事[M]. 呼和浩特:内蒙古人民出版社,2005.
- [10] 乐晓波,陈黎静. Petri 网应用综述[J]. 长沙交通学院学报,2004,20(2):51-55.