第 34 卷 第 11 期 计 算 机 工 程 2008 年 6 月
Vol.34 No.11 Computer Engineering June 2008
·开发研究与设计技术· 文章编号：1000—3428(2008)11—0258—02 文献标识码：A 中图分类号：TP393

Web Services 在黄页搜索引擎中的应用

贺 皓，王正刚，杨义传，胡运发
(复旦大学计算机与信息技术系，上海 200433)

摘 要：提出一种将 Windows 平台上的动态链接库文件封装为.NET 平台下的 Web Services 并使用 Linux 平台下 Java Web 程序调用的方法。该方法已成功应用于中国电信公司黄页搜索引擎系统。该文讨论了整个系统的架构、Web Services 的封装方式及调用方式，利用分布式调度程序提高了系统整体性能。

关键词：Web Services 技术；搜索引擎；分布式调度

Application of Web Services in Yellow Page Search Engine

HE Hao, WANG Zheng-gang, YANG Yi-chuan, HU Yun-fa
(Department of Computing and Information Technology, Fudan University, Shanghai 200433)

【Abstract】 This paper proposes a method that packs the windows DLL as a Web Services on the .NET platform and invokes it from Java Web application. This method has been used in the China Telecom Yellow Page Search Engine project successfully. The system architecture, the packing and invoking method of the Web Services are discussed. It uses a distributed scheduling procedure to enhances the overall performance.

【Key words】 Web Services technology; search engine; distributed scheduling

Web Services 是分布式计算领域最新的应用技术，实现了面向服务的架构，具有良好的封装性和广泛的适用性，强调开放的标准协议规范，采用通用的数据格式。其软件资源的服务接口完全公开，解决了软件跨平台、跨语言和跨防火墙访问等方面的问题，可对现有应用系统进行整合。Web Services 的上述特征对软件的开发和应用具有重大影响。近年来搜索业务已成为互联网业务的重中之重，中国电信黄页公司为了增强其核心竞争力，依托自身原有的大量黄页信息，欲推出专有的黄页搜索引擎。本文针对黄页搜索引擎中的实际问题，提出了相应的 Web Services 解决方案。

1 黄页搜索引擎系统架构描述

整个黄页搜索引擎系统的架构如图 1 所示。

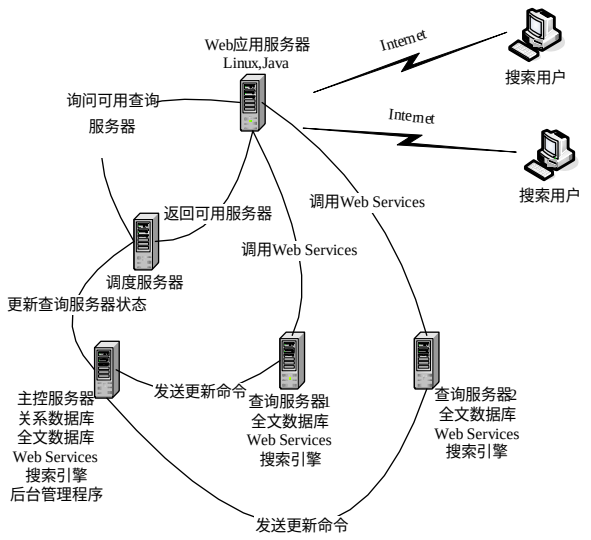

图 1 黄页搜索引擎的基本架构

系统的核心搜索引擎部分使用互关联后继树技术[1-2]。该搜索引擎被封装为Windows平台下的动态链接库(DLL)文件。而根据黄页公司的实际需求，需要用Linux平台上的Java Web应用程序制作系统用户界面。因此，产生了 2 个平台、 2 种语言之间的调用问题，可通过Web Services解决此问题。

整个系统主要分为 Web 应用服务器、调度服务器、主控服务器和查询服务器。Web 应用服务器负责为最终用户提供搜索界面；调度服务器负责与 Web 应用服务器和主控服务器进行通信，完成查询调度分配功能；主控服务器上有关系数据库、后台管理程序及一套查询服务器所包含的全文数据库、Web Services 和搜索引擎程序，主控服务器完成数据的更新、与查询服务器同步全文索引、向调度服务器发送各个查询服务器的状态及建立并更新全文索引等功能；每个查询服务器(可以根据性能需要设置多个查询服务器)上都有一套相同的全文数据库及封装为 Web Services 的搜索引擎，负责向 Web 应用服务器提供查询接口。

2 Web Services 的封装

2.1 Dll 函数的调用

Web Services 是用.NET 平台上最主要的开发语言 C#编写的，为了从 C#中调用 Dll 函数，必须有一个声明，在 C#中使用 DllImport 关键字。DllImport 关键字的作用是告诉编译器需要调用的函数的入口点，并将函数打包在一个类中，这个类可以任意取名。也可以将这些 DllImport 直接放到主类中，代码如下：

```
[WebService(Namespace="http://www.yellowpage.com.cn/IRSTW
```

基金项目：国家自然科学基金资助项目(60473070)
作者简介：贺 皓(1979－)，男，硕士研究生，主研方向：分布式检索，全文检索；王正刚、杨义传，硕士；胡运发，教授、博士生导师
收稿日期：2007-08-12 **E-mail**：hehao@fudan.edu.cn

```
ebService/")]
    [WebServiceBinding(ConformsTo = WsiProfiles.BasicProfile1_1)]
    public class IRSTWebService : System.Web.Services.WebService
    { …
      [DllImport("SHYellowPageDLL.dll")]
    static extern int firstSearch(string str,int flag, int type);
       …  };
```

其中，IRSTWebService 是主类；firstSearch()函数是包含在SHYellowPageDLL.dll 中的一个函数。使用上述代码可以将该函数加入到主类中，可供后面的代码调用。

2.2 一个实际的 Web Services 对外服务接口

下面以一个主要的 Web Services 接口(一次查询接口)为例，介绍实际开发中需要注意的问题。

SearchResult 是一个主要的数据结构，用于包含搜索结果，其大致内容如下(相对实际开发中的定义略有简化)：

```
public class SearchResult
{    public int resultsCount;   //搜索的结果数
     public double searchTime; //搜索时间
     public int countInName;   //在公司名字中命中的个数
     …
     public Record[] results;   //实际存放每个搜索结果的数组}
```

一次实际的搜索 search 代码片段如下：

```
[WebMethod,
System.Web.Services.Protocols.SoapRpcMethod(Use = System.Web.
Services.Description.SoapBindingUse.Literal)]
        [XmlInclude(typeof(SearchResult))]
    public SearchResult search(string str,string searchFlag, int
startIndex, int maxPerPage,int classCount)
    {
            SearchResult sr = new SearchResult();
            if (getDBState() !=1)
            {    sr.resultsCount = -1;
                 return sr;
            }
            lock (m_object)
            {    …
int dwResult = firstSearch(str, fir, 1);
                 …  }
return sr;   }
```

在上述代码中，SoapRpcMethod 用于指定客户端调用 Web Services 所使用的通信协议，本文使用 SoapRpc 方式是为了能和 Java 所写的客户端进行通信，也是 Web Services 所使用的标准通信协议；SoapBindingUse. Literal 是指定 soap 使用的绑定模式，实践表明其他模式不能与 Java 客户端正常通信；[XmlInclude(typeof(SearchResult))]的作用是指明本 Web Method 的返回结果类型 SearchResult 需要进行序列化，以便通过网络传送到 Java 调用客户端，在 Java 客户端再经过反序列化就可以直接调用 SearchResult 中的搜索结果；lock (m_object)的作用是对 search 方法加锁，实现互斥调用，防止 2 个以上的调用同时进入 search 的临界区而导致调用错误，使用这个锁机制可以保证在单机上的每个搜索调用都能按顺序完成相应请求。

3 Web Services 的调用

完成Web Services封装后，需要考虑如何在Java程序中调用它。Java平台下用于Web Services开发的库较多，本文使用Apache下的Axis开发库[3]。

3.1 Axis 简介

Axis 框架来自 Apache 开放源代码组织，是基于 Java 语言的最新 SOAP 规范(SOAP 1.2)和 SOAP with Attachments 规范(来自 Apache Group)的开放源代码实现。很多流行的开发工具都使用 Axis 作为其实现支持 Web Services 的功能，例如 JBuilder 及著名的 Eclipse J2EE 插件 Lomboz。Axis 的最新版本是 1.4，可从 http://ws.apache.org/Axis/index.html 下载。

3.2 Java 调用实例

使用 Axis 中的工具可以使标准 WSDL(Web Service Definition Language)自动生成相应的 Java 类，只须在自己的程序中引入这些自动生成的类就能方便地使用它们。

下面是调用代码片段：

```
String endPoint
="http://10.20.6.2/irstwebservice/service.asmx";
Call call = (Call) service.createCall();
                call.setTargetEndpointAddress(endPoint);
//创建调用类，设置实际的调用链接
call.addParameter("str",
org.apache.Axis.encoding.XMLType.XSD_STRING,
javax.xml.rpc.ParameterMode.IN);
…
//设置调用参数的类型
QName qname=new QName("http://www.yellowpage.com.cn/
IRSTWebService/", "SearchResult");
…
call.setSOAPActionURI("http://www.yellowpage.com.cn/IRSTWe
bService/search");
//设置调用的服务接口
SearchResult result = (SearchResult) call.invoke(new Object[]
{str,searchFlag, startIndex, maxPerPage, classCount});
//调用服务接口得到相应的搜索结果
```

最终的Java Web应用程序的用户界面参见中国电信黄页公司的本地搜(http://www.locoso.com)和商易搜(http://www.e118114.cn)。

4 分布式调度

本文搜索引擎已在单机 6×10^6条数据上测试，可达每秒 100 个的并发请求。如果需要更高的性能指标就要增加查询服务器，需要分布调度程序。本节介绍调度程序的主要功能及实现方法，一些主要思想参考了文献[4]。

4.1 主要功能

调度服务器负责与 Web 应用服务器和主控服务器进行通信，完成查询调度分配功能。

4.1.1 与主控服务器的通信

当主控服务器完成数据索引更新，需要与某台查询服务器进行同步更新时，主控服务器向调度程序发出该查询服务器正在更新的信息，调度服务器在下次分配可用查询服务器时就不会将正在同步更新数据索引的查询服务器分配出去。当同步更新结束后，主控服务器又向调度程序发出该查询服务器同步更新结束的信号。这样调度服务器就可将该查询服务器重新列为可分配对象。

4.1.2 与 Web 应用服务器的通信

当有查询请求发生时，Web 应用服务器先向调度程序发出询问命令，询问当前可用的查询服务器，调度程序通过调度方法将当前可用的一个查询服务器 IP 告知 Web 应用服务

（下转第 272 页）