基于 dom4j 转换 XML 为 XHTML 页面的方法

周 强,李 宇,许雁冬
(中国科学院 国家科学图书馆,北京 100190)

摘 要:跨库检索系统的SRU接口返回的检索结果是XML文件流。IE浏览器可以解析该文件流,根据XSLT文件,自动转换为XHTML文件流,显示检索结果。但是,Firefox,Google Chrome浏览器却无法解析这个XML文件流,它们显示的是非标准格式的文本文字,用户无法查看检索结果。为了使这些浏览器能正常显示检索结果,采用dom4j的应用开发接口,应用XSLT文件,把XML文件流转换为XHTML文件流,从而使检索结果能在Firefox,Google Chrome浏览器上正常显示。

关键词:可扩展标记语言;可扩展样式表转换语言;可扩展超文本标记语言
中图分类号:TP31 **文献标识码**:A **文章编号**:1673-629X(2010)01-0043-03

A Way of Creating XHTML Page from XML by Introducing dom4j

ZHOU Qiang, LI Yu, XU Yan-dong
(National Science Library, Chinese Academy of Sciences, Beijing 100190, China)

Abstract: The returning search results displayed by SRU interface of cross-database search system are the XML document stream. IE browser can resolve the file stream, according to XSLT document, the XML document stream will automatically translated into XHTML document stream, can show search results. However, Firefox, Google Chrome browser was unable to resolve this XML document stream, which shows the non-standard text format text, users can not view the search results. In order to make these browsers display search results, by using dom4j API, according to XSLT documents, translated XML document stream into a XHTML document stream, so search results can normally display in Firefox, Google Chrome browser.

Key words: XML; XSLT; XHTML

0 引 言

跨库检索系统是中国科学院国家科学图书馆开发的集成检索系统,集成了30多个全文数据库、5个文摘数据库、4个电子图书库和多个图书馆公共目录数据库,约有100多个数据库。

跨库检索系统的结果采用SRU接口进行封装,链接在国家科学图书馆的主页的"找文章"上。

常用的浏览器有IE(Internet Explorer),Firefox(Mozilla Firefox),Google Chrome等。

Mozilla Firefox,中文名为火狐,是由Mozilla基金会(谋智网络)与开源团体共同开发的网页浏览器。Firefox是从Mozilla Application Suite派生出来的网页浏览器,从2005年开始,每年都被媒体PC Magazine选为年度最佳浏览器。根据Net Applications的统计,Firefox全世界的浏览器市场份额突破20%关口,仅次于IE。

SRU是一个面向网络环境的信息检索协议,从Z39.50发展而来,目的是通过提供通用的框架结构,整合对各种网络资源的访问规范,使分布式数据库之间能够协同工作。SRU由美国国会图书馆负责维护管理,2004年2月发布了SRU1.1版本。SRU查询结果均采用基于特定元数据规范(如DC)的XML编码[1]。由于SRU实现了Web查询的标准化、查询结果的结构化,因而具有良好的开放性、易用性,近年来,在集成检索服务领域得到了广泛应用。XML是Extensible Markup Language的缩写,全称是可扩展标记语言。通过XML,可以规范地定义结构化数据,使网上传输的数据和文档符合统一标准。用XML表述的数据和文档,可以很容易的让所有程序共享。它的数据存储格式不受显示格式制约,因而XML广泛应用在软件系统的相关文件中,日益成为因特网上的标准数据存储格式和交换格式[2,3]。

XSLT (extensible stylesheet language transformation),是可扩展样式表转换语言,它是一种用来转换

收稿日期:2009-05-26;**修回日期**:2009-08-05
作者简介:周 强(1971-),男,北京人,软件设计师,中国科学技术大学软件工程硕士,研究方向为网络信息系统、信息检索。

XML 文档结构的语言。根据 W3C 的规范说明书,设计 XSTL 的目的是帮助把 XML 文档转换为 XHTML 等文本格式文档[4]。

通过 XSLT,可以从输出文件添加或移除元素和属性,也可以重新排列元素。

XHTML(extensible hyper text markup language),是可扩展超文本标记语言。2000 年底,国际 W3C 组织公布发行了 XHTML1.0 版本。XHTML1.0 是一种在 HTML4.0 基础上优化和改进的新语言,目的是用于 XML 开发应用。XHTML 结合了 XML 的强大功能及 HTML 的简单特性,它的可扩展性和灵活性将适应未来网络应用更多的需求[2]。

跨库检索系统的 SRU 接口返回的是 XML 文件流。IE 浏览器可以解析该 XML 文件流,根据 XSLT 文件,自动转换为 XHTML 文件流,显示检索结果。Firefox,Google Chrome 浏览器却无法解析这个 XML 文件流,显示出的是一些非标准格式文本文字,并且没有排版,用户无法查看检索结果。这就是用 Firefox,Google Chrome 浏览器显示检索结果存在的问题。

为了使 Firefox,Google Chrome 浏览器能正常显示检索结果,文中采用 dom4j API 写程序,把 XML 文件转换为 XHTML 文件。这个 XHTML 文件,可以在 Firefox,Google Chrome 浏览器正常显示。

1 XSLT 转换 XML 为 XHTML

1.1 关于 dom4j 的介绍

dom4j 是 dom4j.org 出品的一个开源 XML 解析包,用于 XML,XPath 和 XSLT 解析,它应用于 Java 平台,采用 Java 集合框架,完全支持 DOM[5~7],SAX 和 JAXP[3]。dom4j 是 Java XML API,它是开源代码软件。dom4j 性能优异、功能强大而且易于使用。dom4j 的最大特点是使用了大量接口,使用接口和抽象类提供了更多的灵活性。

dom4j 主要接口和类如下:

Attribute	Attribute 定义了 XML 的属性
Document	定义了 XML 文档
Element	Element 定义 XML 元素
SAXReader	SAX 方式解析 XML 文件
Transformer	XML、XSLT 转换为文本的主类
TransformerFactory	XML、XSLT 转换为文本的主类的工厂类

1.2 SAX

解析 XML 就是将 XML 文件转换成程序可以理解的对象。Java 提供了两种解析 XML 的方法:SAX 和 DOM[3]。这两种方法是目前主要的 XML 解析方法。文中采用 SAX 方法。

SAX 是 Simple API for XML Parsing 的缩写,它采用事件驱动的方法,当程序解析一个 XML 时,根据读到的 XML 元素生成事件。SAX 是事件驱动的,它从头到尾遍历整个文档,当它遇到一个语法结构时,它会通知运行它的程序,这些是通过事件处理接口 ContentHandler,DTDHandler 和 EntityResolver 中的回调方法实现的[8~10]。

1.3 转换过程的分析

首先,把 XML 文件转换为字符串对象,这样便于处理。

```
String xmlResult = sb.toString();
```

接着,调用 dom4j 的应用开发接口,根据 XSLT 文件,把 XML 转换为 XHTML 文件。

```
xml2htmlString xmlTransHtml = new xml2htmlString();
String sru_sheet="searchRetrieveResponse_html.xsl";
String htmlStream = xmlTransHtml.getHtmlString(xmlResult, sru_sheet);
```

searchRetrieveResponse_html.xsl 就是转换用的 XSLT 文件,下面列出该文件的主要内容。

```
.....
<xsl:variable name="position" select="../../srw:recordPosition"/>
<xsl:variable name="mytitle" select="dc:title"/>
<xsl:variable name="p" select="dc:title_pic"/>
<xsl:variable name="myauthor" select="dc:creator"/>
<xsl:variable name="mykeyword" select="dc:subject.keywords"/>
<xsl:variable name="myabstract" select="dc:description.abstract"/>
```

以上代码,定义变量,从 XML 文件中取出对应元素的值。

```
.....
<td width="5%"></td>
<td width="6%">
  <xsl:if test="contains($p,'green')">
    <img src="/metasearch_SRU/images/green.jpg" alt="即查即得" border="0"width="15" height="15"/>
  </xsl:if>
  <xsl:if test="($position &gt; 0)">
    <b>
      <xsl:value-of select="$position"/>//显示该记录的序号
    </b>
  </xsl:if>
</td>
```

以上代码,根据 XML 文件中 dc:title_pic 元素的值来判断该记录是否为"即查即得"的,接下来显示该

记录的序号。可以看出,这段代码采用了 XHTML 标签。

```
……
<xsl:value-of select="$mytitle" /> //显示标题
<xsl:value-of select="$myauthor"/> // 显示作者
<xsl:value-of select="$mykeyword"/> // 显示关键词
<xsl:value-of select="$ myabstract "/> // 显示摘要
……
```

以上的代码,略去了 XHTML 标签。

类 xml2htmlString 的方法 getHtmlString() 代码如下:

```
……
SAXReader reader = new SAXReader();
//创建 xml 文件字节流
ByteArrayInputStream bs = new ByteArrayInputStream (xml-
String.getBytes());
//解析 XML 文件,在内存建立 xml 对象,得到 Document 对象
(root)
Document document = reader.read(bs);
Document retDoc = styleDocument(document, stylesheet);// 转
为 XHTML
String retDocStr = writeToString(retDoc); // 输出 XHTML 文
件
……
```

图 1 是 styleDocument()的过程图示。

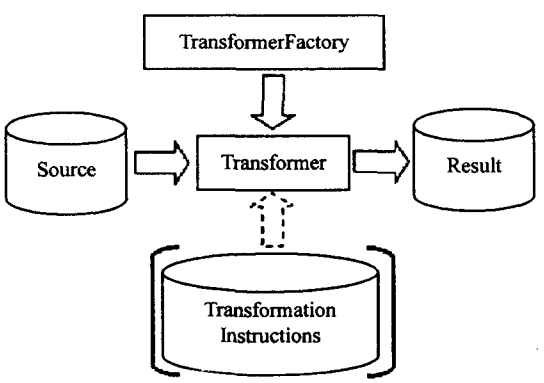

图 1　XSLT 转换 XML 的过程

方法 styleDocument() 的代码如下:

```
public Document styleDocument (Document document, String
stylesheet) throws Exception {
    TransformerFactory factory = TransformerFactory.newIn-
stance();
    Transformer transformer = factory.newTransformer (new
StreamSource(stylesheet));
    DocumentSource source = new DocumentSource(document);
    DocumentResult result = new DocumentResult();
    //转换为 XHTML,下面这行代码是核心的代码
transformer.transform(source, result);
    Document transformedDoc = result.getDocument();
    return transformedDoc;
}
```

可以看到,这段代码是把 XML 文件转换为 XHTML 文件的关键部分,结合过程图示可以形象理解这段代码。

下面是方法 writeToString() 的代码:

```
protected String writeToString(Document document) {
    StringWriter sw = new StringWriter();
    org.dom4j.io.OutputFormat format = org.dom4j.io.Output-
Format.createPrettyPrint();
    format.setEncoding("utf-8");//设置字符编码为 utf-8
    format.setXHTML(true);//设置输出文件格式为
XHTML
    HTMLWriter writer = new HTMLWriter(sw, format);
    try {
      format.setExpandEmptyElements(false);
      writer.write(document);
      writer.flush();
      } catch (Exception e) {
      e.printStackTrace();
      }
      return sw.toString();
  }
```

最后,显示该 XHTML。

```
out.println(htmlStream);//在 jsp 中,用 out 对象输出文件
```

2　结束语

跨库检索系统的 SRU 接口返回的检索结果是 XML 文件流。这 XML 文件流,应用 XSLT 文件,IE 浏览器中可以正常显示检索结果,但是 Firefox,Google Chrome 浏览器检索结果界面是一些非标准格式文本文字,用户无法查看检索结果。

用户希望能够在 Firefox,Google Chrome 浏览器显示,为了满足用户这个需求,采用 dom4j 的应用开发接口,应用 XSLT 文件,把 XML 文件流转换为 XHTML 文件流,该 XHTML 文件流,可以在 Firefox,Google Chrome 浏览器正常显示,用户可以查看检索结果。

现在,跨库检索系统的 SRU 检索,修改好的代码正在运行,检索结果显示正确、运行状态稳定,得到了使用者的好评。

参考文献:

[1] 李春旺,王小梅,王　昉,等.基于 SRU 的集成服务平台设计与实现[J].现代图书情报技术,2007,23(10):12-15.

[2] 左伟明.即用即查——XML 数据标记语言参考手册[M].北京:人民邮电出版社,2007.

(下转第 49 页)

表 3 中,列 Multi Label Correct 表示在传统的准确率评估中,记为错误的实例数,而在文中提出的 EMOSIML 评估方法中认为是正确分类的,而且这些元组均同时属于两个类标签。对于数据集 Corral、Diabetes 和 Glass2,multi label correct 对应的元组在各训练集中所占的比率较大,所以用 EMOSIML 评估方法得到的准确率比 Accuracy 评估方法得到的准确率相对较高。而数据集 Heart 和 Post operative 中, multi label correct 对应的元组在各训练集中所占的比率较小,所以两种评估方法得到的准确率差异较小。

对于以上的五个数据集,在利用朴素贝叶斯分类模型进行分类时,由于数据集中存在元组同时属于多个类别标签,如果使用传统的准确率评估方法,multi label correct 这一列所标示数目的多标签元组,在对每个数据集的分类评估过程中都是被判为错误分类的,所以利用传统的准确率评估方法评估得到的分类器准确率相对较低,而 EMOSIML 评估方法能够很好地解决 SIML 分类评估问题,实验证明文中提出的准确率评估方法较合理。

4　结束语

文中首先对单实例多标签、多实例单标签和多实例多标签分类问题进行了简单的介绍,然后介绍传统的准确率评估方法。由此对 SIML 分类问题提出了一种新的评估方法 EMOSIML,该评估方法的思路是:只要分类器将一个对象分给它所属多个类标签中的任何一个,分类结果都是正确和合理的,再次介绍了该评估方法的计算方法,最后通过实验证明 EMOSIML 评估方法的有效性和合理性。试验所用到的数据集类别数均为两类,使用该评估方法评估两个以上类别的数据集构建的分类模型,也是值得研究的。同时,对 SIML、MISL 和 MIML 分类问题分类算法的设计和评估将是值得研究的一个方向。

下一步的工作:首先利用贝叶斯算法对多类别多标签数据集构建分类器,并在 EMOSIML 的基础上进行改进评估方法,以适合评估多类别多标签分类问题。其次在现有的多标签分类算法的基础上,提出一种新的多标签分类算法。

参考文献:

[1] Hanjiawei, Kamber M. Data Mining Concepts and Techniques [M]. [s. l.]: Morgan Kaufmann publishers, 2000.

[2] Schapire R E, Singer Y. BoostTexter: A boosting - based system for text categorization[J]. Machine Learning, 2000, 39(2 - 3): 135 - 168.

[3] Dietterich T G, Lathrop R H, Lozano - Perez T. Solving the multi - instance problem with axis - parallel rectangles[J]. Artificial Intelligence, 1997, 89(1 - 2): 31 - 71.

[4] Zhou Z H, Zhang M L. Multi - instance multi - label learning with application to scene classification[M]//In Advances in Neural Information Processing Systems 19. Cambridge, MA: MIT Press, 2007: 1609 - 1616.

[5] Zhang M L, Zhou Z H. M^3MIML: A maximum margin method for multi - instance multi - label learning[C]//In: Proceedings of the 8th IEEE International Conference on Data Mining (ICDM'08). Pisa, Italy: [s. n.], 2008: 688 - 697.

[6] Zhang M L, Zhou Z H. ML - kNN: A lazy learning approach to multi - label learning[J]. Pattern Recognition, 2007, 40 (7): 2038 - 2048.

[7] Schapire R E, Singer Y. Improved boosting algorithms using confidence - rated predictions[J]. Machine Learning, 1999, 27 (3): 297 - 336.

[8] 程泽凯,林士敏. 文本分类器的准确性评估方法[J]. 情报学报 2004, 23(5): 631 - 636.

[9] 秦　锋,杨　波,程泽凯. 分类器性能评价标准研究[J]. 计算机技术与发展, 2006, 16(10): 85 - 88.

[10] 秦　锋,罗　慧,程泽凯,等. 一种新的基于 AUC 的多类分类评估方法[J]. 计算机工程与应用, 2008, 44(5): 194 - 196.

[11] Boutell M R, Luo Jiebo, Shen Xipeng, et al. Learning multi - label scene classification[J]. Pattern Recognition, 2004, 37: 1757 - 1771.

(上接第 45 页)

[3] 侯要红,栗松涛. Java XML 应用程序设计[M]. 北京:机械工业出版社, 2007.

[4] Burke E M. Java 与 XSLT[M]. 高　伟,英　宇,译. 北京:中国电力出版社, 2003.

[5] Busatto G, Lohrey M, Maneth S. Efficient memory representation of XML document trees[J]. Information Systems, 2008, 33(4 - 5): 456 - 474.

[6] Wang Fangju, Li Jing, Homayounfar H. A space efficient XML DOM parser[J]. Data & Knowledge Engineering, 2007, 60 (1): 185 - 207.

[7] 蔚晓娟,冉　静,李爱华,等. 基于 DOM 的 XML 解析与应用[J]. 计算机技术与发展, 2007, 17(4): 86 - 88.

[8] 陈　娟,李　晖,鱼　雷. XML 文档快速解析技术研究[J]. 计算机技术与发展, 2007, 17(10): 40 - 42.

[9] 陈　奇. XSLT、XPath 和 DOM 的应用研究[J]. 计算机工程, 2003, 29(3): 14 - 16.

[10] 蔡　剑,景　楠. Java 网络程序设计:J2EE(含 1.4 最新功能)[M]. 北京:清华大学出版社, 2003.