文章编号：1005-8451（2014）07-0039-05

基于Solr的分布式铁路科技资源整合与检索实践

李雪山
（中国铁道科学研究院 科学技术信息研究所，北京 100081）

摘 要：根据铁路科技信用与能力评价的实际需要，针对铁路科技资源存储、利用现状，基于Solr开源搜索平台，提出了分布式铁路科技资源整合与检索解决方案，设计了检索框架，阐述了具体操作方法，进行了实际应用。

关键词：Solr；分布式；资源整合
中图分类号：U29 ： TP39 文献标识码：A

Solr-based practice and retrieval of distributed railway technology resource integration

LI Xueshan
(Scientific and Technical Information Research Institute, China Academy of Railway Sciences, Beijing 100081, China)

Abstract: According to the actual needs concerning the credit and capability evaluation of railway technology and the status quo with regard to the storage and utilization of railway technology resources, this paper, based on the solr open-source information retrieval platform, proposed the distributed solution to the railway technology resource integration and retrieval, designed the systematic framework, elaborated the specific operation methods and applied them in practice.

Key words: Solr; distributed solution; resource integration

铁路科技信用与能力评价，是指按规范的指标体系和科学的评估方法，对被评价对象（如单位和个人）的科研行为、科研成果等全面了解、分析的基础上，作出有关其科研能力、信用可靠性、安全性程度的估量。科技信用与能力评价有利于规范科研行为、避免科研失信、提高科研效率，降低科技投入风险。

铁路科研行为、成果信息广泛存在于被评价对象科研活动中，如立项申报、投标、课题实施、结题、科技奖励、论文（专著）发表、专利申请授权及成果转化情况等。而这些科研活动信息分散于不同的系统或网络应用中，存在标准不统一、数据结构不统一、异构平台和异构应用等问题。而要对被评价对象的科技信用和能力作出科学、全面的评价，则应首先对以上科技资源进行有效整合、充分利用。

1 资源整合方案选型

数据资源整合应基于已有系统或应用，在不影响其正常运行的基础上，对其部分或全部数据进行抽取和有效利用。目前，搜索引擎技术不仅可以使得用户快速获得信息，且已成为数据资源整合的一个重要技术手段。基于搜索引擎的数据整合方案已被越来越多的企业或技术人员研究应用。经笔者调研，其主要应用模式有以下几种 ：

（1）企业自己开发索引工具和软件，对信息进行的索引、检索，达到数据整合的目的。此模式存在软件源码、接口不开放，功能拓展和推广应用困难。

（2）基于 Lucene 封装实现信息索引。该模式在 Lucene 前期，其配套应用（Compass、Solr）还未推出前，有着较广泛的应用，但存在工作量大、

收稿日期：2013-12-19
基金项目：中国铁路总公司科技研究开发计划项目（2011Z011-A）。
作者简介：李雪山，副研究员。

扩展性差、实际应用困难等问题。

（3）调用 Google、百度的 API 实现信息索引。该模式对第三方搜索引擎具有较强依赖性，无法满足后期业务扩展需要。

（4）基于 Compass+Lucene 实现信息索引。该模式适合对数据库驱动的应用数据进行索引，是替代传统的 like ‘ %expression% ’ 来实现对 varchar 或 clob 等字段的索引。该模式对于实现站内搜索是一种值得采纳的方案，但在分布式处理、接口封装上尚需要用户进行一定程度的封装。

（5）基于 Solr 实现信息索引。该模式提供了较为完备的解决方案，封装及扩展性均较好。

综上，基于铁路科技信用与能力评价的现状，探究利用 Solr 对异构异源数据进行整合，提出了实现方法，并进行了实践。

2 Solr搜索引擎

2.1 Solr简介

Solr 是 Apache 软件基金会的一个开源子项目，它是一个高性能的、采用 Java5 开发的、基于 Lucene 全文搜索库的企业搜索服务器。提供了强大的全文检索、高亮显示、分面搜索、动态集群、数据库整合、分布式检索、索引复制及丰富的文档（如 Word，PDF 等）的处理和地理信息搜索等功能 [1]，并提供了完善的功能管理界面。

2.2 Solr与Lucene

Lucene 也是 Apache 软件基金会的一个子项目，是一个开放源代码的全文检索引擎工具包，它本身不是一个完整的搜索程序，只是搜索程序的核心和搜索模块，可嵌入到各种应用中实现针对应用的索引、检索功能 [2]。

Solr 是基于 Lucene，并对 Lucene 的功能进行封装和扩展后而形成的企业级搜索引擎。Solr 实现了 Lucene 服务器化。Solr 和 Lucene 的区别主要为 ：Lucene 本质上是搜索库，需要进行二次开发才能集成到具体的应用中，而 Solr 是基于 Lucene 的独立应用程序 ；Lucene 专注于搜索底层的建设，而 Solr 专注于企业应用，不仅封装了 Lucene 接口，实现了索引库的读写，还可进行动态集群、数据库整合、分布式检索、索引复制等企业级应用操作。即，Solr 是 Lucene 面向企业搜索应用的扩展 [3]。

2.3 Solr特性[4]

（1）易用性。Solr 简化了 Lucene 具体应用，使用户无需或简单编写代码就可实现其企业级应用。利用 Solr，用户在客户端用 POST 方法向服务器发送请求，即可完成索引 ；Solr 支持从数据库、Web 页面和文本中直接导入数据，进行索引 ；Solr 还可根据需要修改配置文件，完成字段定义、是否被索引、是否存储、中文分词器、默认检索字段、检索方法等配置工作。

（2）异构性。Solr 的一个突出特点是提供了对异构系统的数据整合方案，在动态集群、分布检索、索引复制、检索结果排序、查重、显示等方面均提供了完整的解决方法。

（3）易集成性。Solr 是一个 Web 应用，它支持 PHP、Java、Perl、C# 等多种客户端调用其搜索和索引。客户端和服务端之间基于 HTTP 协议进行通信，客户端可以创建 HTTP 请求，然后解析 response 成各语言能识别的对象或结果，这样实现了 Solr 与多系统、多语言环境的集成。

3 检索架构设计

目前，铁路科技信用与能力评价所需数据分布在不同系统或应用中。如科研立项、实施及结题信息存在于科研系统，招投标信息存在于科技招标系统，铁路科技成果鉴定、评审信息存在于成果管理系统中，奖励信息则来源于国家科技部及铁道学会网站，专利信息则来源于国家知识产权网站等。基于以上实际，笔者设计了检索框架。该框架主要分为索引库建立与检索两部分，简述如下。

3.1 索引建立

如图 1 所示，本文根据数据来源不同，采取了不同的数据采集、整合策略。因科研管理、招投标、成果管理等系统为笔者所在课题组开发，拥有系统源代码，并负责运营维护，在建立索引时，课题组在这些系统上部署了 Solr 索引模块，实现了数据整合。具体过程如下 ：

（1）分析各系统要采集的数据信息（数据表及字段），在 Schema.xml 对采集字段进行元数据定义 ；

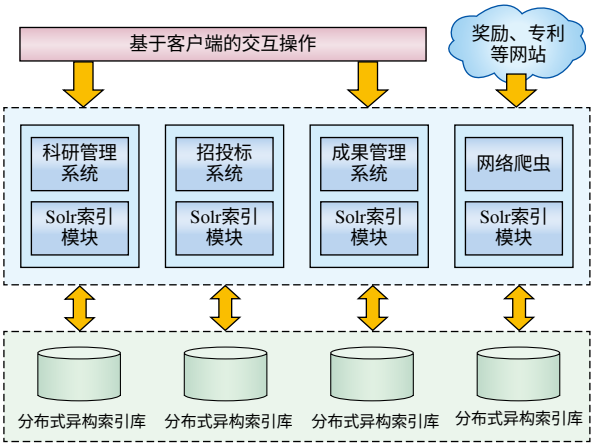

图1　分布式索引检索框架

（2）基于各系统业务数据库，对已有数据进行批量导入，建立分布式异构索引库 ；

（3）基于元数据定义，修改各业务系统相关代码，对数据的增、删、改等操作增加了 Solr 索引功能，实现了索引数据与具体业务数据的同步。

对于国家奖、铁道学会奖、知识产权等非课题组管理的网站，利用 Web-Harvest 在通过对目标网站网页结构进行分析的基础上，提取了数据，并通过 Solr 建立了索引，最终形成了分布式异构索引数据库群。

3.2　数据检索

Solr 通过分布 (Distributed) 和复制 (Replication) 策略，实现了分布式数据的检索 [5]。Solr 分布式检索特性可将分布在多个服务器上的资源进行分别索引，再利用片（Shards）技术，将相同的检索请求同时发送到集群内任意服务器进行检索，最后将整合后的检索结果返回到最初的调用服务器。此种特性使得分布式异构资源整合变得较为容易实现。检索过程如图 2 所示。

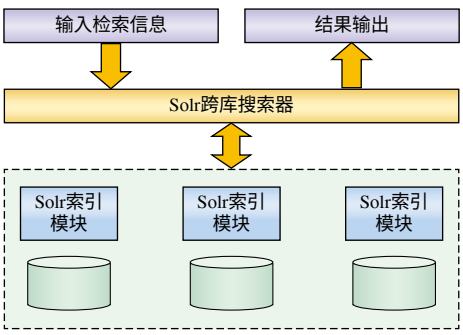

图2　数据检索流程

其中，Solr 跨库搜索器，主要接受用户搜索查询请求，并将用户请求转换为 Solr 内部语法格式后，向分布式 shards 发送查询请求，并对查询结果进行排序、查重、过滤后返回给用户。Solr 跨库搜索器实现较为简单，可部署于任意服务器上，其示例代码如下 ：http ：//localhost/solr/select?shards=172.20.0.62:8083/solr, 172.20.0.65:8083/solr&q= 铁路 &facet=true&facet.field=name。其中，172.20.0.62:8083/solr 与 172.20.0.65:8083/solr 为两个分布式索引服务器地址。

4　检索实现

4.1　Solr安装配置

Solr 的运行，需先安装在 JDK 和 Servlet 容器（如 tomcat），然后下载 Solr 安装文件（本文使用 4.4 版本），解压后拷贝 dist 目录下的 solr-4.4.0.war 文件到 tomcat 的 webapps 目录下 , 并重命名为 solr.war。设定工作目录为 D:\solrhome\solr，并将 example 下 multicore 中配置文件拷贝到其中。打开 webapps 下 Solr 网站中的 web.xml 文件，在其中加入 ：

```
<env-entry>
    <env-entry-name>solr/home</env-entry-name>
    <env-entry-value>D:\ solrhome\solr</env-entry-value>
    <env-entry-type>java.lang.String</env-entry-type>
</env-entry>
```

以指定工作目录的位置。然后访问 http://localhost:8080/solr 出现 Solr 的系统管理界面，则配置成功。

4.2　中文分词配置

英文以空格作为分隔符，而中文词语之间没有分隔，在建立中文搜索引擎时，首先需要对中文进行切词。目前，此类工具较多如 IKAnalyer、Paoding、mmseg4j 等。本文使用 mseg4j-1.9.1 进行了切词。将 mseg4j 下载并解压后将其 dist 下的 jar 包拷贝到 tomcat\webapps\solr\WEB-INF\lib 目录中。最后，修改 Schema.xml 文档中的内容，在 <types> 标签中添加如下内容 ：

```
<fieldType name="text_zh" class="solr.TextField" positionIncrementGap="100">
```

```
        <analyzer>
    <tokenizerclass="com.chenlb.mmseg4j.solr.
MMSegTokenizerFactory" mode="complex" />
          </analyzer>
        </fieldType>
```

实现了对中文分词器的配置。

4.3 元数据定义

在数据索引前，需先在 Schema.xml 文件中对要索引的字段进行定义，具体包括字段（fields），唯一标识符（uniqueKey），默认检索字段（defaultSearchField），默认搜索设置（solrQueryParser）等。代码片段如下：

```
    <fields>
       <field name="id" type="string" indexed="
true" stored="true" required="true" />
        <field name="name" type="text_zh"
indexed="true" stored="true" />
       <field name="content" type="text_zh"
indexed="true" stored="true" />
        …
       <field name="_version_" type="long"
indexed="true"  stored="true"/>
     </fields>
    <uniqueKey>id</uniqueKey> // 唯一标示符
设置
    <defaultSearchField>name</defaultSearch-
Field>
    <copyField source=“ name ”dest=“ all ” >
    <solrQueryParser defaultOperator="OR"/>
```

其中 <fields> 节点具体定义了要索引字段的配置，name 是字段名、type 是分词器，indexed 是否被索引，stored 是否存储。copyField 是将所有的字段复制到一个字段中，以便进行统一检索，solrQueryParser 配置了默认检索参数之间的逻辑关系，可为 OR，也可为 AND。

4.4 数据批量导入与索引库建立

对已有的数据，如铁路科研立项、结题、成果鉴定等数据，本文利用 DataImportHandler 进行了直接数据库导入。其在 Solr 的主要配置如下：

在 solrconfig.xml 文件中加入：

```
    <requestHandler name="/dataimport"
        class="org.apache.solr.handler.dataimport.
DataImportHandler">
        <lst name="defaults">
         <str name="config">db-data-config.xml</
str>
        </lst>
      </requestHandler>
```

此步启用了批量数据导入模块，并指定了数据库配置文件 db-data-config.xml。

在 db-data-config.xml 中加入：

```
    <dataSource driver="com.microsoft.sqlserver.
jdbc.SQLServerDriver"
        url="jdbc:sqlserver://localhost:1433;Datab
aseName=bky"
         user="sa"  password="sa"/>
```

配置了数据库名，访问用户名密码等信息。

在 db-data-config.xml 中加入：

```
       <entity name="project" deltaQuery="select
ID from PROJECT where to_char (CHANGEDATE
'yyyy-mm-ddhh:mi:ss' ) > '${dataimporter.
last_index_time}'"
        query="select ID,PROJECT_NAME,MAIN
_CONTENT from PROJECT"
    deltaImportQuery="select  from PROJECT
where ID ='$ {dataimporter.delta.ID}'" >
```

此部分为执行导入配置了具体数据表及字段。deltaQuery 和 deltaImportQuery 为执行增量导入时的数据库查询语句，选择 CHANGEDATE 的时间大于上次执行导入的时间的条目，实现了增量导入，避免了全部导入重复内容浪费的时间。Query 未执行完全导入时执行的数据库查询语句。

4.5 查询、添加与删除索引

Solr 提供了基于 Java 的 API，即 SolrJ。SolrJ 对 HTTP 链接和 XML 命令进行了封装，为使用 Java 代码处理 Solr 更加方便，简化了索引创建、搜索、排序和分类等操作。查询主要代码如下：

```
    HttpSolrServer solrServer= new HttpSolr-
Server（URL）；
    SolrQuery query = new SolrQuery();
    query.setQuery(":");
    QueryResponse rsp = solrServer.query( query );
    SolrDocumentList docs = rsp.getResults();
```

（下转 P47）

作流程的基础上，以其安全性和受限活性为验证目标，建立整个系统交互的时间自动机网络模型，作为验证的基础。根据 CTC 分界口临时限速系统约束提出功能和性能等系统性质，利用 UPPAAL 验证工具，验证了各条性质均得到满足，从而说明了系统的安全性和受限活性，为系统的设计和开发提供了一定的依据。

参考文献:

[1] 古天龙 . 软件开发的形式化方法 [M]. 北京 : 高等教育出版社，2005 ：5-67.

[2] 吴永刚，陆慧娟 . 基于时间自动机的实时系统建模及验证 [J]. 计算机时代，20116（1）：2-3.

[3] 刘传会，张广泉 . 一种基于时间自动机网络的实时系统形式化验证方法 [J]. 苏州大学学报，2008，24（1）：35-40.

[4] 袁　磊，王俊峰 .CTCS-3 级列控系统临时限速建模与验证 [J]. 西南交通大学学报，2013，48（4）：710-712.

[5] OLDEROGER, DIERKSH. Real-time systems[M].London: Cambridge University Press，2008: 137-146.

[6] 康仁伟 . 基于时间自动机的 CTCS-3 级列控系统建模方法与验证研究 [D]. 北京 : 北京交通大学，2013 ：40-53.

责任编辑　徐侃春

（下转 P42）

```
for ( Object obj:docs) {
    SolrDocument doc=(SolrDocument)obj;
    String name = (String ) doc.getFieldValue
("name");
}
```

通过 SolrJ 需先连接 HttpSolrServer，定义 SolrQuery，添加查询语句 setQuery()，然后通过 QueryResponse 类型的对象读出查询结果 ；添加索引时，先创建 SolrInputDocument 对象，通过 addField() 方法添加相应内容，最后执行 add() 以及 commit() 即可 ；在建立连接的基础上执行 deleteByQuery () 方法，并 commit() 即可删除索引。

5　结束语

Solr 作为一种开源的搜索引擎，具有功能强大、易实施、易应用，灵活性、可扩展性强等优点，为数据资源整合、索引、检索提供了一套较为简单的模式。将其应用在网站索引、检索及数据资源集成检索等系统中具有明显优势，前景广阔。本文根据实际需求，基于 Solr 提出了分布式铁路科技资源整合与检索解决方案，设计了系统框架，并进行了实践，取得了较好的效果。

参考文献：

[1] Apache Solr[EB/OL]. http://lucene.apache.org/solr/，2014-01-07.

[2] 管建和，甘剑峰 . 基于 Lucene 全文检索引擎的应用研究与实现 [J]. 计算机工程与设计，2007（1）：489-491.

[3] netoearth. Apache Solr 介绍 [EB/OL]. http://blog.netoearth.com/html/201104/apache-solr- 介绍 .htm,2014-1-4.

[4] 张建勇，廖　凤，刘小兵，陶超全 . 集群与负载均衡技术在国际科学引文数据库服务系统中的应用研究 [J]. 现代图书情报技术，2010（6）：25.

[5] 马凤娟 , 吴鹏飞 . 基于 solr 的异构资源集成检索框架设计与实现 [J]. 现代情报，2012（8）：133-135.

责任编辑　徐侃春