

基于事件驱动的异构数据交换系统的研究

袁 林, 张新家, 李 菲

(西北工业大学 自动化学院, 陕西 西安 710129)

摘 要:为解决异构数据的交换问题,提出了一个基于事件驱动和 Web Services 的异构数据交换系统,系统利用数据中心及适配器相结合的框架来解决异构问题。数据中心提供接入系统的管理及数据的转发功能;适配器提供了驱动事件的管理、异构数据的转换、异构系统的屏蔽等功能,是系统的核心部分。事件驱动机制提高了系统的效率和灵活性,而 Web Services 技术使系统各部分的通信跨越了防火墙的障碍。经实例测试系统可以有效解决异构数据源之间的差异性,为数据源之间数据灵活的交换提供了有效的工具。

关键词:数据交换;事件驱动;Web Services;XML

中图分类号:TP311

文献标识码:A

文章编号:1673-629X(2011)12-0100-05

Research of Heterogeneous Data Exchange System Based on Event-Driven Mechanism

YUAN Lin, ZHANG Xin-jia, LI Fei

(College of Automation, Northwestern Polytechnical University, Xi'an 710129, China)

Abstract: To solve the problem of heterogeneous data exchange, it introduces a heterogeneous data exchange system based on event-driven mechanism and Web Service. This system provides a framework which consists of a center part and adaptors. The center manages all the subsystems while the adaptors contribute for the management of events and the transformation of heterogeneous data, the adaptors play an important role in the system. Event-driven mechanism improves the efficiency and flexibility of the system, while the Web Services technology makes the communication go through the firewall. Exchange data among different operating systems and data sources efficiently by this platform.

Key words: data exchange; event-driven mechanism; Web Services; XML

0 引 言

随着企业业务的不断扩展,软件系统变得越来越复杂,越来越多的系统和公司被牵连进来,但各个系统的开发都是独立进行的,数据存储及管理系统也各异,这为需要进行数据交换的各方带来了许多麻烦^[1]。本系统采用数据交换中心结合适配器的系统结构,可有效解决存储异构及管理系统异构的问题并能满足数据的同步、异步交换及一对一、一对多的交换,同时事件驱动的方式增加了数据交换的灵活性,交换双方可通过设定事件来触发数据的交换。当前,对于异构数据的交换已有很多研究^[2-4],各个方案都有自身的优点与不足^[5],文中提出的方案能够有效地解决数据交换中的关键问题。

1 系统的设计思想

1.1 群组思想

在现有的大部分研究中,各个异构数据库利用 XML 文件创建局部模式,通过模式之间映射来解决异构问题,模式映射有两种情况,其一是两两映射, n 个模式将产生 $n(n-1)$ 个映射文件;其二是构建全局模式,以全局模式为中介向目标模式转换^[6]。前者工作量大且产生很多无效动作。后者在实践中很难实现,因为不可能找到一种全局模式适合所有的数据格式。文中提出群组的思想来简化数据格式的转换,在交换系统中由提供共享数据的节点申请建立群组,并注册所要共享的数据模式,其他数据节点根据需要申请加入群组,之后用户手动匹配映射表,配置过程简单且精确,可以消除生成全局模式的复杂性。当数据接收节点收到数据时,依据映射表进行数据格式转换并进行数据库更新。

1.2 事件驱动

事件驱动是程序按照事件发生的次序执行而不是

收稿日期:2011-05-16;修回日期:2011-08-20

基金项目:国家自然科学基金(60803158)

作者简介:袁 林(1986-),女,硕士研究生,研究方向为 web 服务、数据交换;张新家,副教授,研究方向为计算机网络、图形图像、软件重构等。

按照编程时就定义好的顺序执行,当某个事件发生时程序将找到相应的事件处理程序来处理事件^[7]。事件驱动使数据交换实现异步及松耦合且使源节点不再阻塞于接收目的节点的回应。事件的发生即设定条件的满足,条件包括时间周期、数据库条目个数上限等,当该条件满足时才触发数据交换动作,系统提供的条件表达式工具可满足用户的各种需求。

2 系统模型

数据交换系统主要完成异构的关系数据库之间数据的交换,系统的结构如图 1 所示。

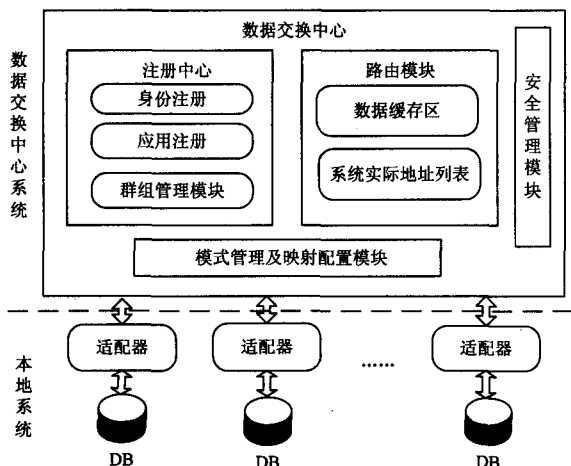


图 1 系统的整体结构

系统由交换中心系统和本地系统组成,数据交换中心主要负责用户的登陆、注册、权限管理、数据转发等功能;本地系统的核心为适配器,主要提供数据中心交互的接口、数据的打包与解析、数据库的交互等功能。该系统的网络结构如图 2 所示。

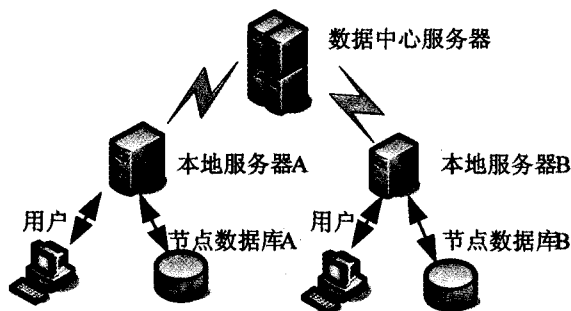


图 2 系统的网络结构

交换中心系统放置在专门的服务器上,需要进行数据交换的各个数据节点分别有自己的本地服务器,该服务器为各数据节点的适配器提供工作环境。用户利用浏览器与本地服务器及数据中心服务器交换信息完成各项配置,从而实现各数据节点间的正确交换。

2.1 交换中心结构

2.1.1 注册中心

注册中心的实现主要依靠 LDAP 目录服务^[8]。

LDAP 将用户的注册信息进行存储并将系统的服务信息(应用注册中可公开的信息)发布给公众,从而让用户可以选择是否需要加入该系统及需要系统为其提供哪些服务。系统中,注册信息来源于两个部分:身份注册及应用注册。身份注册的数据主要是 IP 地址、单位名称、单位描述、联系方式、web services 服务接口信息等,应用注册主要包括异构数据库的相关信息,如数据库的各个表信息以及表的各个字段信息(如:字段类型、字段长度、字段名称等)。中心提供各种注册界面接收用户的输入信息。

2.1.2 模式管理及映射配置

模式即数据源节点公布的共享数据描述,类似数据库中的视图,是基于数据表的信息集合,主要包括字段信息、数据表的关联关系、驱动事件设置等。群组是基于模式建立的,一个模式对应一个组,组成员即为目的数据节点。源节点需向数据中心注册自己所要共享的数据表,根据注册的数据表可配置数据模式,配置成功后系统为模式分配 ID 并自动建立群组,群组号即为模式 ID。用户首先浏览系统提供的服务(所有可见的数据模式),根据自己的需求选择模式并加入该模式对应的组,之后进行数据映射关系的配置,映射配置是解决数据异构的关键步骤,主要解决数据类型的匹配、字段的对应、数据单位的转换等异构问题^[9-11]。

2.1.3 路由模块

该模块完成了数据转发的功能。数据交换开始后,源节点将共享数据发往数据交换中心,中心将数据进行缓存,根据数据中携带的群组 ID 号确定各个目的节点,从路由表中读出实际地址进行数据的转发。群组成员都成功接收数据后,中心将缓存中的数据删除。

2.2 适配器结构

2.2.1 web services 接口部分

该部分主要解决数据存储形式的异构。适配器主要通过 web services 接口与交换中心进行交互:

(1) receiveTransactScript(), 该方法接收模式配置文件,解析文件查看驱动事件设置信息,若为周期模式或其他条件模式则将其存入系统数据库并将事件信息交由驱动事件管理部分进行处理,若为实时交换模式则提取模式中的数据表及其字段信息,生成 SQL 查询语句,执行该语句从节点数据库提取数据。系统利用 JDOM 技术将提取出的数据以预先设定的格式生成树型结构,之后将其转换成字符串类型并作为参数传递给数据交换中心的 web services 接口中。该接口主要为源节点提供服务。

(2) receiveDestActionScript(), 该方法用于接收映射配置文件并将其保存在系统数据库,其 ID 与对应模式 ID 相同,目的节点依据此 ID 提取映射表,随后解析

共享数据。

(3) receiveDataSet(), 该方法用于接收共享数据。当源节点将数据发布到交换中心后, 中心调用该接口进行数据的传递。

2.2.2 驱动事件管理部分

事件驱动是数据交换的起点, 设定的事件发生则触发数据交换动作开始, 该部分在数据的源节点发挥作用。源节点可发布多个数据模式, 对同一模式又可设定多个事件条件, 事件的发生具有很大的不确定性, 为确保事件触发的准确性与实时性事件的管理必不可少。驱动事件管理部分主要完成事件的登记、启动、注销等操作, 利用线程池技术来协调管理事件。

2.2.3 数据库交互部分

该部分主要用于屏蔽数据库管理系统的异构性。本系统由 java 语言编写, 数据库连接部分基于 JDBC 接口规范, 具有跨平台特性也适用于 internet 上的异构环境。利用 JDBC 提供的标准 API, 对于主流的各类关系数据库的连接只需更改配置信息, 这些信息包括数据库连接 URL、用户名、密码、驱动程序。

3 适配器的实现

3.1 事件驱动

事件驱动模块主要包括三个要素: 事件源即事件的产生方, 系统中的事件包括定时器超时或条件表达式成立等; 监听器接收事件源发来的通知并将它们交给事件处理方法进行处理; 事件处理方法是系统在事件发生后做出的反应, 对于本数据交换系统该方法一般为数据的提取与发送动作, 若用户有进一步的设置则按照设置进行处理, 如数据发送后对源数据表的操作等。事件驱动的处理过程如图 3 所示。

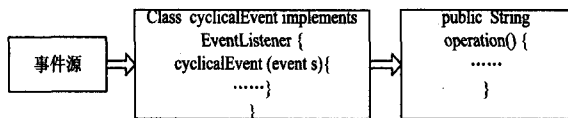


图 3 事件驱动处理过程

cyclicalEvent 为接收周期事件的监听器类, 实现于 EventListener, 该类创建了一个监听器, 接收事件并调用处理函数 operation() 进行事件处理。

3.2 数据库操作

数据库操作的关键部分为数据库的连接、数据结果的封装, 以数据表元数据信息的提取为例, 利用 JDOM 提取信息并转换为 XML^[12] 的过程如下:

1) 连接数据库:

```
Class.forName( driver );
```

```
Connection ct = DriverManager.getConnection( url, username, password );
```

2) 实例化 DatabaseMetaData 对象, 获取数据表结

构信息:

```
DatabaseMetaData dbmd = ct.getMetaData();
```

```
ResultSet keyRs = dbmd.getPrimaryKeys( null, null, tableName );
```

```
ResultSet colRs = dbmd.getColumns( null, null, tableName, null );
```

3) 将返回的结果生成 Document 文档:

```
Element dataRoot = new Element( "MetaData" );
```

```
Element classNode = new Element( "tableClass" );
```

```
Element tableNode = new Element( "table" );
```

```
tableNode.setAttribute( "name", tableName );
```

```
.....
```

```
Document doc = new Document( dataRoot );
```

XML 文件的结构如下:

```
<MetaData>
```

```
<table name=" ">
```

```
<columns>
```

```
<primaryKey> </primaryKey>
```

```
<item name=" " type=" " size=" " nullable=" " />
```

```
.....
```

```
</columns>
```

```
</table>
```

```
.....
```

```
</MetaData>
```

MetaData 表示文件为元数据信息, columns 标签下是具体的字段信息, 每一个 item 代表一列, 字段信息包括数据类型、长度、是否为空、主键信息。

共享数据的提取与封装过程与此类似, 其 XML 文件的结构如下:

```
<DataSet DataSwitchTrans_ID=" ">
```

```
<row>
```

```
<result></result> .....
```

```
</row> .....
```

```
</DataSet>
```

DataSwitchTrans_ID 为模式 ID, 此 ID 表示了模式、群组、映射表的对应关系, 每个 row 代表一条记录, result 的内容为具体字段值, 其排列顺序与模式配置文件中的字段对应。由于各部分数据传输都利用 XML 格式的数据, 解析时利用 JDOM 提供的 SAXBuilder 解析器, 将 XML 文件解析为 Document 文档, 进而提取各节点的数据。

3.3 数据匹配算法

映射表中的动作类型包括“Put”、“Update”、“Delete”分别对应 SQL 语句中的“insert”、“update”、“delete”操作; 操作语句基于 SQL 语句并嵌入表达式来表示映射关系, 基本表达式有: result(x, y)、replace(x, z)。前者为简单的字段数据类型转换, x 表示目的数据类型, y 表示模式文件(即源数据)中字段的序号; 后

者可表示较复杂的数据映射, x 仍表示目的数据类型, z 表示子表达式的 ID 号, 子表达式以 $\text{result}(x, y)$ 为运算单元, 利用算术、比较、逻辑等运算符来完成较复杂的数据映射。

以字段合并为例介绍数据匹配算法。假设映射表动作类型为“Put”; 操作语句为“insert table (colA) values (replace(s, 1))”, 即将表达式 $\text{replace}(s, 1)$ 所代表的内容插入 table 的 colA 列中; ID 为 1 的子表达式为 $\text{result}(s, 1) + \text{result}(s, 2)$, 表示将模式文件中第 1 项与第 2 项字段的值合并, 算法如下:

- 1) 根据共享数据中的 DataSwitchTrans_ID 项, 提取映射表, 将其解析并提取其中的操作语句, 检测操作语句中是否有 $\text{result}(x, y)$ 项, 若有则执行步骤 2), 若无则跳至步骤 3);
- 2) 利用字符串操作将 $\text{result}(x, y)$ 中的参数提取出来, x 取值为 s 表示数据类型为 String。根据 y 的值从共享数据中提取对应项的内容并转换成 String 类型, 利用 java 提供的 $\text{replace}(\text{oldStr}, \text{newStr})$ 函数将 $\text{result}(x, y)$ 项替换为实际需要插入的数据, 检测是否有 $\text{replace}(x, z)$ 项, 有则执行步骤 3), 无则跳至步骤 5);
- 3) 提取 $\text{replace}(x, z)$ 项中的参数, 根据 y 的值获得子表达式信息, 执行步骤 2) 将 $\text{result}(x, y)$ 项用实际数据替换, 将“+”号用空格替换;
- 4) 利用步骤 3) 得到的数据将 $\text{replace}(x, y)$ 项替换;
- 5) 输出可直接使用的 SQL 语句。

4 数据交换流程及实例

4.1 数据交换流程

数据交换中心首先将数据模式配置文件发往数据源节点, 源节点解析配置文件并启动驱动事件, 由驱动

事件触发数据交换的过程, 如图 4 所示。源节点、交换中心、目的节点通过交互完成整个交换流程, 标号 (1)、(2)、(3) 表示一次数据交换过程。

4.2 数据交换实例

假设 A 节点为源节点, 注册表 1 如下:

表 1 tableA

ID(int)	FirstName(String)	LastName(String)
00001	Bjarne	Stroustrup

数据节点 B 为目的节点, 注册表 2, 此表为空:

表 2 tableB(a)

UserID(int)	Name(String)

节点 A 配置数据模式, 其中的关键部分为:

```
<Action ActID="0" SQL="select tableA. ID, tableA. First-
Name, tableA. LastName" TN="tableA">
<EventItem>
<Event tp="1" TimeVal="6" TimeUnit="1"/>
</EventItem></Action>
```

Action 标签内为共享数据的具体描述, ActID 为中心分配的模式号, “SQL”项说明该模式共享 tableA 的所有列数据且“ID”、“FirstName”、“LastName”字段序号分别为 0、1、2; EventItem 标签下为驱动事件的具体设定, tp=“1”表示事件类型为周期事件, TimeVal 为周期大小, TimeUnit 为时间单位: 0、1、2 分别表示时、分、秒, 此事件为每隔 6 分钟进行一次数据发送。模式配置完成后中心将其存入数据库并发送对应脚本文件到节点 A, 节点适配器接收该文件并解析从中获取驱动事件信息, 启动定时器等待驱动条件的满足。节点 B 申请加入 A 节点共享模式所对应的组, 配置映射表, 等待接收数据。

当驱动事件满足, A 节点根据模式配置文件中的

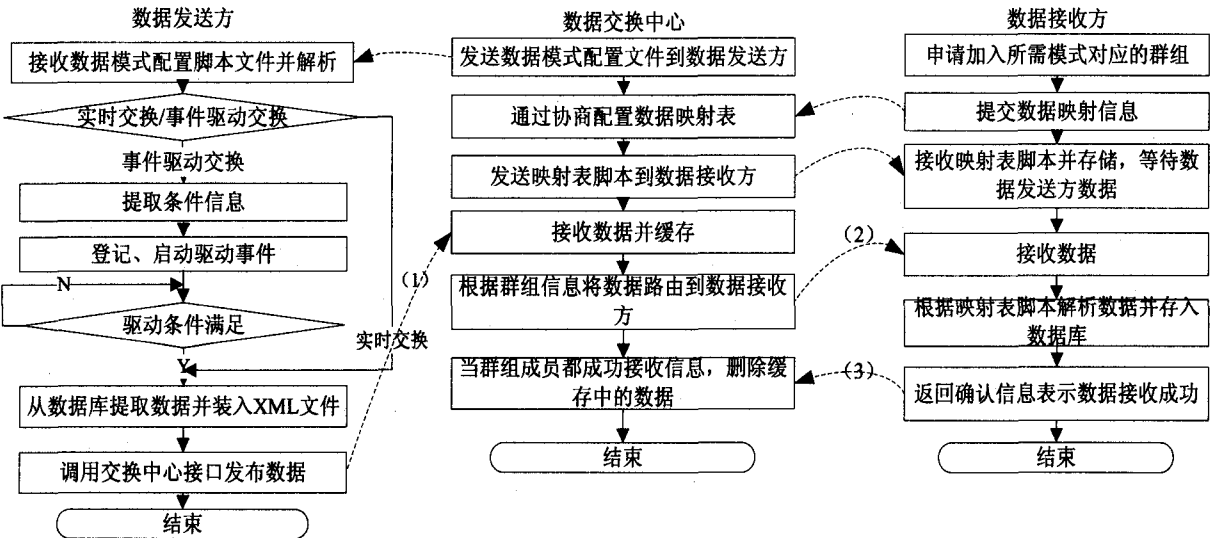


图 4 系统总体交换流程

信息,提取数据并封装成 XML 格式的共享数据,发往给数据交换中心,交换中心将数据进行缓存并转发,路由模块调用 B 节点的 web services 接口传输数据。B 节点接收数据,根据其携带的 DataSwitchTrans_ID 值提取对应的映射表,根据表 3 中的数据匹配算法进行数据转换,利用算法输出的 SQL 语句进行数据库操作。交换过程结束后 tableB 的内容如下:

表 3 tableB(b)

UserID(int)	Name(String)
00001	Bjarne Stroustrup

5 结束语

文中详述了系统的结构以及适配器的实现技术,利用实例说明了数据交换的过程,系统能够为异构的数据交换提供有效的解决方案,数据交换的准确性与效率较高。下一步的工作是设计更好的数据流程,进一步提高系统的效率。

参考文献:

[1] 唐俊伟. 基于 Web 技术的数据交换平台的研究与设计[D]. 西安:西北工业大学,2006.

[2] 袁磊,李帅. 基于 WebServices 的异构系统信息交换的实现[J]. 计算机技术与发展,2008,18(12):8-10.

[3] 马云峰,王源. 基于 XML 和 WebService 的异构数据集成研究与实现[J]. 计算机技术与发展,2010,20(11):42-

(上接第 99 页)

中不难看出,改进前团队的生产率平均不超过 5,并且 Sprint 结束后还有部分任务完不成。改进后,不但按时完成任务而且生产率基本在 5 之上。

3 结束语

公司的文化、管理模式不同,开发者的经验不同等等都决定了 Scrum 方法不能被按部就班地套用。Scrum 宗旨是提倡灵活,因此在实际运用中的灵活改进并不有悖于 Scrum 的原则。相反一味的借鉴模仿只会形似而神不似,导致适得其反的结果。“敏捷”本身就意味着适时地变化,应该在敏捷和不敏捷之间寻找一种合理的折衷方案。在实践中对敏捷模式进行适当的改进和灵活的运用,能够帮助我们更好地抓住敏捷开发方法的精髓,最终达到预期的目标和效果。

参考文献:

[1] Schwaber K, Beedle M. Agile Software Development With Scrum[M]. [s.l.]:Prentice Hall, 2001.

[2] Leffingwell D. 可伸缩敏捷开发:企业级最佳实践[M]. 李

46.

[4] 章明,许青松,沈锡军. 基于 XML 的数据交换共享平台模型[J]. 清华大学学报(自然科学版),2003,43(1):105-107.

[5] 李亚楠,刘连忠,贾毓星. 数据交换研究[J]. 计算机技术与发展,2008,18(2):5-8.

[6] 王艳敏,谢强,丁秋林. 基于本体和 WebServices 的数据交换平台[J]. 计算机技术与发展,2010,20(5):112-116.

[7] Philip G C. Software design guidelines for event-driven programming[J]. The Journal of Systems and Software,1998,41(2):79-91.

[8] 欧阳辉. 基于 LDAP 协议的异构数据源集成与信息共享关键技术研究[D]. 广州:华南师范大学,2007.

[9] Trinh Q, Barker K, Alhajj R. RDB2ONT: A Tool for Generating OWL Ontologies From Relational Database Systems[C]. [s.l.]:AICT/ICIW,2006.

[10] Xu Jiuyun, Li Weichong. Using Relational Database to Build OWL Ontology from XML[C]//International Conference on Computational Intelligence and Security Workshops. Washington, DC, USA:IEEE,2007.

[11] El-Khatib H T, Williams M L, MacKinnon L M. Using a distributed approach to retrieve and integrate information from heterogeneous distributed databases[J]. Computer Journal, 2002,45(4):381-394.

[12] 史立军,侯红,祁方民. 基于 Schema 的 XML 文档到关系数据库的转换[J]. 计算机技术与发展,2008,18(10):71-74.

冬冬,译. 北京:电子工业出版社,2009.

[3] 杨璇. 基于敏捷方法的科研管理系统的设计与实现[J]. 计算机技术与发展,2008,18(12):182-185.

[4] Pollice G. 敏捷起源以及敏捷创始人的传记[EB/OL]. 2009. <http://www.scrumcn.com/agiledev/html/?169.html>.

[5] Martin R G. 敏捷软件开发—原则、模式与实践[M]. 邓辉,译. 北京:清华大学出版社,2003.

[6] Kniberg H. 硝烟中的 Scrum 和 XP—我们如何实施 Scrum[M]. 李剑,译. 北京:清华大学出版社,2011.

[7] 钟杨,刘业政,马向辉. 小团队结对编程实践研究和重构[J]. 计算机技术与发展,2007,17(11):160-163.

[8] 张克强. 让管理层害怕的 8 个敏捷理解[EB/OL]. 2011. <http://www.scrumcn.com/scrumpc/html/?254.html>.

[9] 贝克. 测试驱动开发[M]. 孙方,译. 北京:人民邮电出版社,2008.

[10] 张扬,黄厚宽. 测试驱动开发及开发实践[J]. 计算机技术与发展,2006,16(5):75-77.

[11] 柳纯录. 系统集成项目管理工程师教程[M]. 北京:清华大学出版社,2009.

[12] 青瑞. 软件工程之全程建模实现[M]. 北京:机械工业出版社,2010.