

# 基于 EAI 消息平台的异构系统数据同步

蒋 溢, 丁 优, 熊安萍

(重庆邮电大学计算机科学与技术学院, 重庆 400065)

**摘 要:** 针对电信企业异构信息系统的数据同步与共享问题, 提出一种基于企业应用集成(EAI)消息平台的数据同步方法。利用 EAI 技术对异构系统进行无缝集成, 借助消息平台接收并解析异构系统的数据消息, 触发 ETL 组件处理更新数据, 实现异构系统数据同步。测试结果表明, 该方法具有较好的实时性, 并能有效降低异构系统的资源消耗。

**关键词:** 企业应用集成; 异构系统; 消息平台; 可扩展标记语言; 数据同步

## Heterogeneous System Data Synchronization Based on EAI Messaging Platform

JIANG Yi, DING You, XIONG An-ping

(College of Computer Science and Technology, Chongqing University of Posts and Communications, Chongqing 400065, China)

**【Abstract】** Aiming at the problem that the heterogeneous systems for telecommunications companies data to be shared real-time and synchronization, this paper presents heterogeneous system data synchronization method based on Enterprise Application Integration(EAI) messaging platform. It uses EAI technology to seamlessly integrate heterogeneous systems, message platform, sends a message to trigger the data Extraction Transformation Loading(ETL) components to handle data update by accepting and parsing heterogeneous system, heterogeneous system to achieve real-time data synchronization. Test result proves that this method not only realizes the sharing of data synchronization, and also reduces the resource consumption of the heterogeneous systems effectively.

**【Key words】** Enterprise Application Integration(EAI); heterogeneous system; messaging platform; eXtensible Markup Language(XML); data synchronization

DOI: 10.3969/j.issn.1000-3428.2011.21.018

### 1 概述

随着电信企业业务的不断发展增加,企业内部数据分散、出现信息孤岛不利于资源的有效利用。为此,大部分企业采用数据仓库解决方案或加入操作数据存储(Operational Data Store, ODS)<sup>[1]</sup>平台。虽然这2种方法能够实现数据的统一利用,但难以保证数据的实时性、有效性以及跨系统的数据交换和跨系统的数据应用需求,因此,必须建立起统一的数据交换平台<sup>[2]</sup>,通过统一的数据服务层实现异构系统的数据共享及同步,从而对交换数据进行一致性控制。

本文结合可扩展标记语言(eXtensible Markup Language, XML)及面向服务的数据同步方法<sup>[3]</sup>,并借助企业应用集成(Enterprise Application Integration, EAI)消息平台较好地实现了异构信息系统共享数据的准实时同步。

### 2 异构系统数据同步

根据目前企业情况,从数据集成的发展过程,出现的数据源同步方法有模式集成方法<sup>[3]</sup>、数据仓库方法、基于XML的数据同步方法<sup>[3]</sup>和面向服务的同步方法等。基于XML的数据同步方法利用XML作为交换数据的公共语言,屏蔽了底层数据库的物理位置差异和结构差异。面向服务的同步方法将异构数据源封装为Web服务,向服务中心注册,应用端可以向注册中心发送查找请求,获得数据服务。

结合XML与面向服务的数据同步方法在对异构系统数据处理方面的优点,基于电信操作数据平台和EAI消息平台统一控制和管理异构系统的共享数据,并通过消息机制驱动ETL(Extraction Transformation Loading)组件的执行,促使

ETL及时、准确地进行增量或全量数据处理。

### 3 基于 EAI 消息平台的异构系统

#### 3.1 异构系统架构

EAI平台的高层视图是“三总线”架构<sup>[4]</sup>,即企业服务总线(Enterprise Business Bus, ESB)、企业信息总线(Enterprise Information Bus, EIB)以及业务流程总线(Enterprise Process Bus, EPB),本文主要利用EAI消息平台的ESB和EIB的功能,保证异构系统数据的准实时同步。基于EAI消息平台的异构系统集成架构如图1所示。

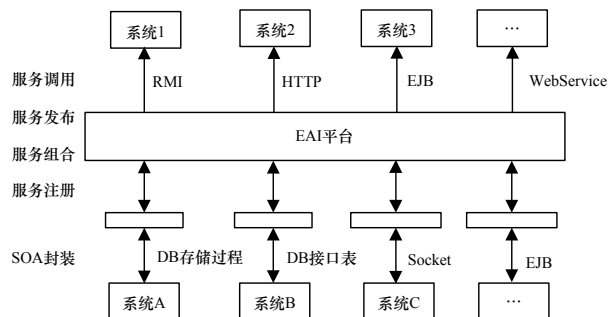


图1 基于 EAI 消息平台的异构系统集成架构

**基金项目:** 重庆市自然科学基金资助重点项目(2008BA2017); 重庆市信息产业发展专项基金资助项目(200811004)

**作者简介:** 蒋 溢(1969—), 男, 高级工程师, 主研方向: 企业信息化, 异构信息系统; 丁 优, 硕士; 熊安萍, 副教授

**收稿日期:** 2011-05-12 **E-mail:** jiangyi@cqupt.edu.cn

从图 1 可以看出,所有的异构系统都利用面向服务的技术以服务的方式通过服务组合和服务发布并注册在 EAI 平台,实现异构系统的集成,并通过服务调用和服务组合机制实现系统之间的无缝连接。当 EAI 平台需要系统信息或通知系统事件时,可以调用这些封装的服务进行平台和系统之间的交互。在此种交互方式的基础上,可以实现数据传递、同步更新和信息交互等应用。

### 3.2 基于消息的异构系统数据同步处理机制

为有效保证企业共享核心数据在各异构系统中的准实时同步,减少外围系统和 EAI 平台的接口设计的复杂性,并根据系统的数据更新要求,本系统在实施过程中采用基于 EAI 消息平台的消息驱动方式保持异构信息系统的数据同步。这里需要一个完整的消息处理机制进行消息管理和处理。

如图 2 所示,CRM、计费源系统和 ODS 共享库通过 EAI 消息平台进行集成,为了简化系统间的消息接口,由 EAI 消息平台监控各业务系统的数据更新,通过 ODS 再向其他数据应用系统提供统一的客户视图。

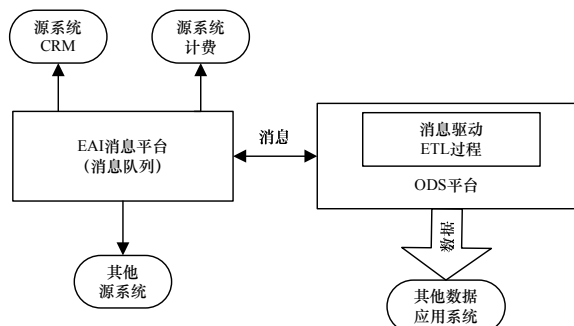


图2 基于 EAI 消息平台的异构系统数据同步

在该体系结构下, EAI 消息平台扫描各业务系统的数据更新,如果发现系统发生数据更新,则 EAI 生成相应的消息并封装成一个 XML 格式的消息文件,并将该消息文件置入消息队列,通过消息调度向 ODS 发出更新请求消息,而 ODS 平台部署的消息引擎会解析消息并完成更新数据的装载,从而将更新数据同步到 ODS 平台,通过 ODS 向其他应用系统提供统一的数据视图,达到异构系统数据同步的目的。这种数据同步处理流程如图 3 所示。

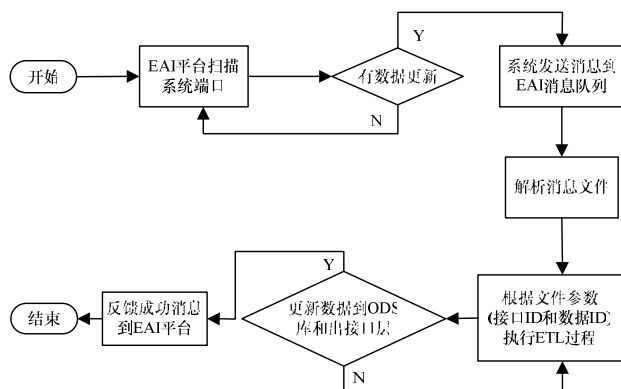


图3 数据同步处理流程

在图 3 中, EAI 消息平台发送的消息有 2 种格式,只包含命令的消息和包含命令和数据的消息。EAI 平台会根据进入消息队列的先后顺序并结合优先级设置进行消息调度,通过消息解析程序通知 ETL<sup>[5]</sup>处理程序执行 ETL 过程。

以上 2 种对于系统数据更新消息的处理都能够将数据同

步到 ODS 库中,而在数据进入 ODS 后,考虑到下游系统对更新变化数据的使用,将更新至整合层的数据也更新到出口层。

## 4 基于 EAI 消息平台的数据同步实现

基于 EAI 消息平台的数据同步实现分 2 步进行处理:

(1)通过不同的接口方式(RMI 接口)将外围异构信息系统集成到 EAI 平台,为 EAI 处理消息文件做好信息通道基础。

(2)在第(1)步的基础上利用 EAI 平台的消息队列存放外围系统的数据消息,并对其进行解析处理,触发 ETL 过程的执行,同步数据到 ODS 库中,并且对出口层数据也进行相应的更新。

### 4.1 接口实现

这里以 EAI 和 ODS 系统为例阐述相关实现。经过项目前期工作组的约定,ODS 系统对 EAI 平台提供基于 EJB 规范的接入方式,EAI 平台对 ODS 系统提供基于 RMI 规范的接入方式。ODS 系统与 EAI 平台的系统接口见图 4(以 RMI 接口方式为例)。

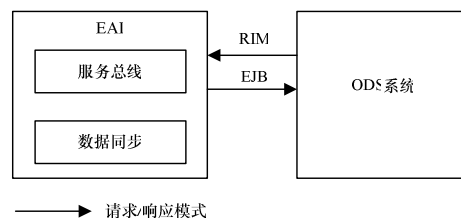


图4 ODS 与 EAI 的系统接口

RMI 接口消息包含 3 个部分:

(1)FunctionCode: 接口标识码,类型为 String。

(2)Header: 包含接口标识,EAI 控制信息,类型为 XML String,XML String 为汉字内码扩展规范(GBK)编码。Header 中包含一部分业务信息,主要用于异常处理。

(3)Body: ODS 系统发送的服务请求数据,或者服务后的返回数据,类型为 XML String,XML String 为 GBK 编码。

RMI 的接口方式:

(1)ODS 系统发送服务请求(到 EAI 平台)。定义 Header 和 Body 的 Schema;Header 内部包含的接口标识码必须与 EAI 平台内部定义的一致,如果不一致则直接返回 ODS 系统,报告数据错误。

(2)ODS 系统接收服务处理结果(从 EAI 平台)。EAI 直接发送返回的结果,以 XML String 方式,不回送 Header 和接口标识码。

通过接口实现了 EAI 平台和 ODS 系统之间的同步通信,同时,使 ODS 系统集成到 EAI 平台。

### 4.2 数据同步的实现

#### 4.2.1 消息文件内容

通过以上接口设计以及 ODS 数据共享平台和 EAI 平台通信接口的设计,为异构数据同步提供了数据处理的通道,在此基础上只需要进行数据的同步处理实现。整个实现过程分为数据信息解析加载、共享数据平台更新以及同步到出口层 3 个步骤。以上步骤的成功执行需要 XML 格式的消息文件支撑,只有处理正确的 XML 参数,EAI 平台才能够调用正确的 ETL 过程执行数据处理(包括数据加载、同步处理等),以及返回正确的处理信息。以下是一个最简单的 XML 文件消息,该消息对规则文件和数据文件的名字和相关主要参数做了记录以便在解析时唯一的确定待处理对象。

```
<? xml version="1.0" encoding="UTF-8" ?>
<SqlldrReq>
  <InterfaceId>423135</InterfaceId>
  <CycleId>201007291028</CycleId>
</SqlldrReq>
```

此 XML 消息文件中定义了 InterfaceId 接口 Id、CycleId 周期这 2 个参数,解析文件后通过 SqlLoader 的方式对数据进行处理。在文件中通过 InterfaceId 的 value 可以确定唯一执行的 ETL 过程,因为在 ETL 过程的入参定义中有一个接口标识与其一一对应;从 CycleId 对应的 value 值可以找到这个周期内的增量数据文件(比如此 XML 对应的增量数据文件名字为: a\_201007291028\_423135.dat 以及处理规则文件: a\_201007291028\_423135.verf),在规则文件里,定义了几个重要的参数,以 a\_201007291028\_423135.verf 规则文件内容进行说明:

```
a_201007291028_423135.dat
13851                293
201007291028 201007291028 44
```

第 1 行的 a\_201007291028\_423135.dat 是此规则文件对应的增量数据文件,是目标执行文件;第 2 行的 13851 和 293 分别是增量数据文件生成的文件大小和总记录条数;第 3 行的 201007291028 和 44 分别是增量数据文件在某个周期内的增量数据以及此周期的时间长度(以秒记)。在规则文件定义这几种参数值后才能使 ETL 过程正确执行和处理增量数据文件。

#### 4.2.2 消息文件处理

在对消息文件参数的说明后,只需要利用 EAI 平台正确调用此规则文件和对应的消息文件就可以成功实现对数据进行 ETL 操作。

在触发这些文件时,通过 EAI 平台定义 Java Socket 通信,首先监控源系统服务器的固定端口,将 XML 文件写入字符串输出流、缓冲各个字符,从而提供单个字符、数组和字符串的高效写入到 ODS。最后通过 BufferedWriter 的 flush() 方法对该流缓冲进行刷新以保证写入效率。核心代码如下:

```
public static String getXML(){
    String xml = "<? xml version='1.0' encoding='UTF-8' ?>
    <SqlldrReq>
    <InterfaceId>423135</InterfaceId>
    <CycleId>201007291028</CycleId>
    </SqlldrReq>";    return xml;
} //获取 XML 文件,为输入数据信息做准备
String xmlStr = getXML();
Socket socket = new Socket("135.0.30.92",15467);
for(;;){
    OutputStream ops = socket.getOutputStream();
    OutputStreamWriter opsw = new OutputStreamWriter(ops);
    BufferedWriter bw = new BufferedWriter(opsw);
    bw.write(xmlStr); bw.flush();}
```

通过对规则文件和数据文件的处理,完成了数据整合最重要的步骤,即是源系统发生变化的增量数据成功的进入到 ODS 平台中。然后需要将外围系统和 ODS 之间进行相应的数据更新即可。

从数据同步处理机制的性能考虑,在项目实施过程中,

需要减少系统与 EAI 平台的接口设计,因此,当更新消息传递到 ODS 平台时,由其完成基于消息驱动的数据更新,而外围数据应用系统则基于 ODS 平台。

## 5 系统运行环境与结果分析

系统运行硬件环境为 HP Unix 系统平台,编译运行环境是 Oracle 9i、jdk1.4、WebLogic8.1.6。在测试系统中,分别对某个表新增 573 条、680 条、988 条和 1 081 条记录,并对其同步处理。按照目标要求其整个流程的处理时间应在 10 s 内。测试结果对线程用时(程序代码实现)和数据加载时间(后台过程记录)进行记录,如表 1 所示。

表 1 实验测试结果

测试数据量	线程用时/s	加载时间/s	总时间/s
573	0.121	0.779	0.99
680	0.121	1.089	1.21
988	0.121	2.989	3.11
1 081	0.121	4.869	4.99

在系统的实际运行中,千条左右的记录数只需要 5 s 之内即可更新到 ODS 共享库和出接口层。在对大量的数据更新进行处理时,时间一般耗费在 ETL 数据处理过程(即线程 Socket 时间和数据入库时间之和),但是整个的过程耗时在理想的数值之间,通过系统影响因素得出时间耗费的公式为:  $T=T_0+T_1$ ,并得到数据同步处理结果,其中:

$$T_1 = \begin{cases} T_{\min} & \text{一个线程socket最短距离} \\ \alpha \times T_{\min} & \text{多个线程socket时间, } 1 < \alpha \leq 4 \end{cases}$$

$$T_0 = \begin{cases} T_{\min} & \text{单数据载入数据库时间} \\ (\alpha + \beta) \times T_{\min} & \text{多个数据载入数据库时间,} \\ & 1 < \alpha \leq 4, 1 < \beta \leq 8 \end{cases}$$

结果显示,与传统的用后台 Job 方式每天或每月定时执行数据同步过程的处理方法相比,基于 EAI 消息平台的数据同步机制在系统性能和实时性要求上都有很大的提高和优化。

## 6 结束语

异构系统数据的实时同步是一个很复杂的处理过程,要考虑的因素较多、模块也较复杂,因此,在系统架构方面要处理的内容相当多。基于 EAI 消息平台的异构信息系统的同步的提出和实现可以很好地解决企业中准实时或单独的数据同步,尤其在对信息系统的集成、增量数据处理方式的选择等方面都有很大程度的提高。

## 参考文献

- [1] 李晓东,杨 扬,郭文彩. 基于企业服务总线的数据共享与交换平台[J]. 计算机工程, 2006, 32(21): 217-218, 226.
- [2] 王 博,李腊元,冯美来. 基于数据同步的增值业务的开发[J]. 计算机工程, 2009, 35(3): 115-117.
- [3] 顾天竺,沈 洁,陈晓红,等. 基于 XML 的异构数据集成模式的研究[J]. 计算机应用研究, 2007, 24(4): 94-96.
- [4] 高 彬,谷建华. 基于 ESB 的实时 ETL 系统的设计与实现[J]. 计算机应用, 2008, 28(4): 822-825.
- [5] 许 力,马瑞新. 基于 SOA 的实时 ETL 的研究与实现[J]. 计算机系统应用, 2007, 16(4): 24-27.

编辑 陆燕菲