A Method of Data Synchronization Based on Message Oriented Middleware and Xml in Distributed Heterogeneous Environments

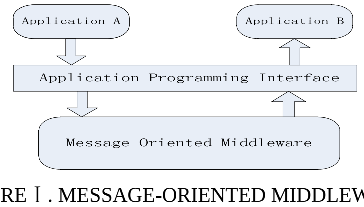
Z.Y. Lu, Z.B. Guo
College of Information Engineering, Qingdao University
Qingdao, 266071, China

Abstract—With the rapid development of information technology, lots of enterprises are being faced with resource share and information exchange under global heterogeneous environment. However, the MOM and XML technology can provide a better solution for the realization of information exchange and resource share of distributed heterogeneous environment. In this paper, we adopt four procedures to realize data synchronization between different databases. 1) get incremental data of source by triggers; 2) convert incremental data into XML file via XML mapping layer; 3) send XML file to destination in a message format; 4) parse data by XML parser.

Keywords-MOM; XML; heterogeneous environment; information exchange;data synchronization; trigger

I. INTRODUCTION

With the rapid development of information and internet technology, it is a vital issue for enterprise to how to achieve heterogeneous data exchange under the network environment. In summery of current research, there are two completely different middlewares can implement the data transfer between different systems of distributed applications[1]. One is based on Remote Procedure Call(RPC) and another is based on Message-Oriented Middleware(MOM). The RPC-based middleware includes Java RMI, CORBA and DCOM, and the MOM includes MQSeries, SonicMQ, RocketMQ and ActiveMQ, etc. However, in the RPC environment, each application system node must always be linked in order to ensure smoothly communication between them. In the MOM environment, online system nodes do not need to require real-time and they also not need to know address with each other. By comparison, MOM-based middleware[2] is more suitable for enterprises. As is shown in FigureⅠ.


Application A
Application B
Application Programming Interface
Message Oriented Middleware

FIGUREⅠ. MESSAGE-ORIENTED MIDDLEWARE.

MOM uses messaging platform-independent mechanism for data exchange and is applied to for distributed systems based on data communications. As the absence of a common standard, a variety of message-oriented middleware[3] systems is difficult to achieve interoperability and seamless connectivity. Java message service (JMS) is proposed by the SUN company. It aims to unify various MOM specification interface and provides a set of specific implementation-independent interface. JMS supports Pub/Sub and P2P message models and provides message filtering and transaction processing mechanisms. As an intermediate data format, XML can facilitate the realization of the interaction between a variety of data sources. In this paper, it realizes the data exchange and loosely coupled distributed applications under communications with the JMS and XML's excellent properties[4-5].

II. DATA SYNCHRONIZATION METHODS AND STRATEGIES

There are two main data synchronization methods, namely complete synchronization and incremental synchronization. Complete synchronization will transfer the data of all departments' database to the central database one time, and update transmitted data synchronously by the central database, which has simple logic but requires a huge network overhead. In contrast, incremental synchronization only needs to transfer the updated data to target database to ensure data synchronization, which can save network overhead and thereby enhance the efficiency of synchronization. In incremental synchronization, the key is how to acquire incremental data.The most commonly used method to obtain incremental data are trigger[6], time stamp[7] and scanning table methods. Each method has its own advantage and disadvantage. For instance, time stamp is easy to realize as its simple principle but waste system resources; and scanning table method has low efficiency for recording large tables.

This paper presents an application-independent database data synchronization system with trigger mode to automatically capture data source changes. In order to complete the data consistency of the different nodes and achieve data synchronization, this paper represents data changes by XML format and recover data by forwarding message through the JMS server[8] to the destination.

III. FRAMEWORK DESIGN

A. *Model Structure*

In order to maintain data consistency between the database the central database and eliminate "information islands", we need to synchronize the data in both databases within a certain time. Because of the platform differences and the diversity of databases, common replication technology becomes weak. It needs a reliable, common data synchronization method. Early adopters of Socket, RMI and other technologies require relatively high performance about network or other aspects that can not fit the actual situation. Here is a data synchronization method based on JMS. It uses asynchronous communication mode, shielding the platform and the underlying communication protocol differences, fitting the data synchronization between heterogeneous databases.

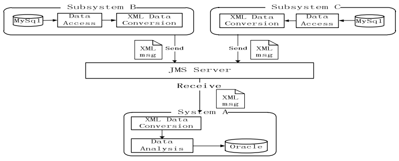
Now supposing the following scenarios: A company’s business scattered around, it needs to update the data around a central database to achieve data consistency in order to fit the needs of business. Now A is the center of the business enterprise, B and C are the enterprise’s subsidiary company businesses. However, the system, network and hardware are different. Therefore, this system uses the JMS deployment, shielding the heterogeneity of the environment, and the data can be transmitted across platforms. JMS server with asynchronous transfer capabilities. After the client B and C send a message to the JMS server, they don’t need to wait the confirmation of the system A. During this period, the client B and C can perform other operations, which can improve the performance of the system. Because there are several subsystems, JMS uses the Pub/Sub mode for data transmission that can be able to complete to-many communication. When the node A needs to update the node B’s data, node A will subscribe a topic in JMS server. Then node B publishes the message to JMS server on this topic, JMS server will send this message to node A to make consistent data update, which is shown as FigureⅡ.

However, due to the use of the database may be inconsistent across that causing the data difference in the data pattern and the traditional way in which data replication is not well suited. Now node A deployment is the enterprise data center using an Oracle database, while node B and C are using Mysql database. Since XML has the separation of content and form, good scalability, cross-platform portability and self-describing advantages, the system can transmit message with XML Text. Because XML is a text essentially, so it can be transmitted in the Text Message instance’s message body. As in the JMS framework above, it can use XML-based model to exchange data between node A and node B, C. The following is the step of switching and transmitting data of the system:

1) Node B get data source from MySql database and through XML Data Conversion to convert data source into XML text.

2) Node B encapsulates the XML text into an XML message, and then sends it to the JMS server.

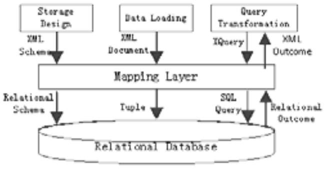
3) After the node A receiving the XML messages from the JMS server, the XML message will be converted to the Oracle data format. And finally, node B’s data store to the node A’s Oracle database successfully.



FIGUREⅡ. JMS IMPLEMENTATION ARCHITECTURE.

B. *The Mapping between Xml and Relational Databases*

As mentioned above, one issue is how to convert data between relational databases and XML. First of all, the paper will figure out the mapping between XML and relational databases as shown in FigureⅢ. The Mapping between them is bidirectional. In the paper[9], it introduces the two mapping methods, namely based on template driven and model driven. The method based on template driven is a shallow mapping, which can only be used for transform data from other types to XML documents. The latter is a deep mapping, and it is a mapping between model and model, which needs a specific model to realize when transferring data from the database into an XML document or transferring data from an XML document to the database. Since the model-driven mapping is based on bidirectional mapping that we employ in this paper, it can realize the data conversion between the relational database and XML document.



FIGUREⅢ. THE STORAGE MODE OF RELATIONAL DATABASE.

C. *Trigger Design*

Triggers are a kind of special mechanism of DBMS and divided into data manipulation language(DWL) triggers and data definition language(DDL) triggers, of which DWL can be activated via the insert, update and delete events, While DDL triggers can be activated by both modes of AFTER and INSTEAD. And this paper employs AFTER mode to activate trigger.

When we insert, update, or delete one or more columns of the table of source data, the table's trigger is activated. And then we write the operation type, primary keys, change of rows and other information into the record table and process state as 0 to acquire the changing data.

Mutative converter listens record table in a definite time. The mutative converter would convert change information of characterization data into an XML file via XML mapping layer after read changeable record table when the data has changed. And the it would send the XML file to the destination database through the JMS server. And the XML file is to be parsed to relational database by XML parser, and the parsed result is updated to destination database, which can complete data synchronization between different databases.

IV. CONCLUSION

In this paper, it provides a platform-independent, asynchronous, loosely coupled messaging with JMS. And it could be used to resolve the problem of XML messages application support layer within the enterprise and across the Internet. And combination of both, it will perform better under heterogeneous distributed application environments and help system with data synchronization and data requested operation well.

REFERENCE

[1] XIONG Hui, LIU Yan-feng, GUO Da-qing. Design and Implementation of Distributed Heterogeneous Database Migration System. Computer Engineering, 34(4), pp. 57-59, 2008.

[2] LIANG Yan-Jie, LIAN, Dong-Ben. Design of Architecture on Transporting Data for Data Exchange Platform Based on MOM. Computer System and Application, 21(4), pp. 12-15, 2012.

[3] Made Sukaras, Nyoman Sarasuartha Mahajaya, Gusti Made Arya Sasmita. Design of Message-Oriented Middleware Based on Social Messenger. IJCSET, 2(4), pp. 1129-1134, 2012.

[4] LÜ Meng, XIA Jie, BI Yu-shen, LI Sa, CHEN Jian-di. Design and research of integrated framework of XML and JMS in heterogeneous environment. Journal of Anhui University Natural Science Edition, 29(6), pp. 23-25, 2005.

[5] Yu, Xiao-peng, CHEN, Jian-xiao, LI, Yong-li. Research and Implementation of Data Consistency for Distributed Database System Baded on Message Queue. Journal of Jilin University(Information Science Edition), 29(3), pp. 259-262, 2011.

[6] Liu Shu, Xu Shuo, Huang Qiquan, Sun Lu, Ding Ran, Wang Lihua. RESEARCH ON A TRIGGER-BASED DATA SYNCHRONISATION SYSTEM AND THE IMPLEMENTATION OF KEY TECHNIQUES. Computer Applications and Software, 29(12), pp. 190-191, 2012.

[7] FANG Wei-qing. Hospital Fixed Assets Management System Based on Time Stamp Data Synchronization. Computer and Modernization, pp. 200-202, 2013.

[8] Yuan, Zhen. A Method of Data Synchronous Updating Based on Message Oriented Middleware. Ordnance Industry Automation, 32(7), pp. 93-96, 2013.

[9] FU Ling-li, JIN Zhi-gang, LIANG Yan-hong. Study of Transformnation between XML and Relational Database. Journal of Adult Education School of Hebei University of Technology, 22(1), pp. 34-36, 2007.