2009 Ninth International Conference on Hybrid Intelligent Systems

Design and Implementation of Data Management Center Based on Web Services

Qiang Guo[1], Haidong Zheng[2], Jie Li[3], Xingwei Wang[4]
School of Information Science and Engineering, Northeastern University,
Shenyang, 110004, Liaoning, China
[1]guoqiang_mike@yahoo.com, [2]heden1970@yahoo.ca

Abstract — **In a large e-government system, different government departments have already created and are going to create a large number of heterogeneous databases. Only using data replication and synchronization technology as the low level infrastructure to maintain the huge system is very difficult to get satisfactory result. A data center constructed with Web Service technique and XML schema can give a good solution to problems with business logic method invocation and transparent data exchange in low layer, and thus achieve better data sharing and integration.**

Keywords - metadata, Web Service source, finding, searching, transparent accessing

I. INTRODUCTION

A huge e-government system is an organic whole, in which different governmental departments are independent each other on the one hand, but also have dependent relationship on information and need data exchange and information sharing on the other hand. However, many sub-systems are constructed at different times. With the development of new technologies, these sub-systems may adopt different development architectures and programming languages. Thus information exchange is very difficult between different departments because some applications may have used different database products or different versions of the same product and the databases are maintained by individual department independently, which results in many information islands. There are three methods to achieve data sharing between the information islands. The first one is to construct a distributed database that is the same as the concentrated database from the point of view of the user operations. The second is data sharing, which has lower coupling. As the databases of different departments are independent, partial data sharing may be achieved. This requirement is from the e-government systems and campus-wide data centers and this method is commonly used currently. The third is to define unified data definition rules, exchange protocols, and data invocation method management so as to create a distributed data center separated into individual e-government departments or to create a centralized physical data center [1]. This paper discusses the third method. For this type of data sharing, the method is to construct a sharing database with all the needed shared data. The data source should guarantee data consistency and the database using the shared data should guarantee the native data are consistent with that of the shared database. This method can realize the consistency requirement for all the shared data, but it cannot implement data resource discovery and retrieval. With the emerging of databases based on Web and the wide applications of Web Service concept, data resource discovery and retrieval have become an urgent problem to solve. This paper presents a method which not only achieves the consistency of shared data but also solves the problem with data resource discovery and retrieval, which makes it easier to create a new database or to construct a new application based on the existing application environment. In the following sections, this paper will discuss this method with the creation of a user login information management center as an example.

II. OVERVIEW OF A DATA MANAGEMENT CENTER

A huge information system of an organization is typically distributed. The aim of the data management center is to provide a centralized management platform for the affiliated departments. The administrator can manage the whole system by this platform. The users can easily search and subscribe data from each database by a unified interface or accomplish some tasks by special applications at the center [2].

A. *The Main Functions*

- Create metadata table. The administrator reads metadata from each heterogeneous database and then stores them with XML format to create a metadata table. Users can easily find and retrieve the data they need by this table [3].
- Subscribe information. The users select the information to subscribe through a user interface. The user is informed by the data management center when the subscribed data are changed.
- When the user creates a new database, the standard web service invocation methods can directly use the new database. The collaborated system does not need to know the manufacturer and the technical details of the database. Thus transparent and seamless integration is achieved [4].

B. *The System Structure*

The system structure of a Web service based data management center is shown in Figure 1. The system uses browser/server multi-layer system framework based on

JAVA platform and a service based on solution for remote data access. Although there are a variety of remote data access technologies, the purpose of this paper is to give a schema based on Web service. Compared with traditional methods, the Web Service based solution has some obvious advantages, for example, simplicity, cross-platform, and cross-firewall. However, Web service is stateless, this means that Web services cannot remember what a user does from one invocation to another. Thus before the invocation of the first Web service is completed, the next Web service cannot be invoked. Obviously, this will restrict the concurrent operations of the database. Therefore, J2EE Session Bean together with the factory mode is used to solve this problem. The factory mode is in charge of creating instances with lifecycle.

C. *System Security*

Security is an important part of a data center system. Web services as the core of the data management center moves data or other objects outside of the firewall for the sake that external entities can use the application programs, which results in new security challenges to the Web service system [5]. The proposed system uses J2EE security mechanism to protect the functions of this system. Because this technology provides a mature software module that includes a large amount of contents, it will not be discussed in more details.

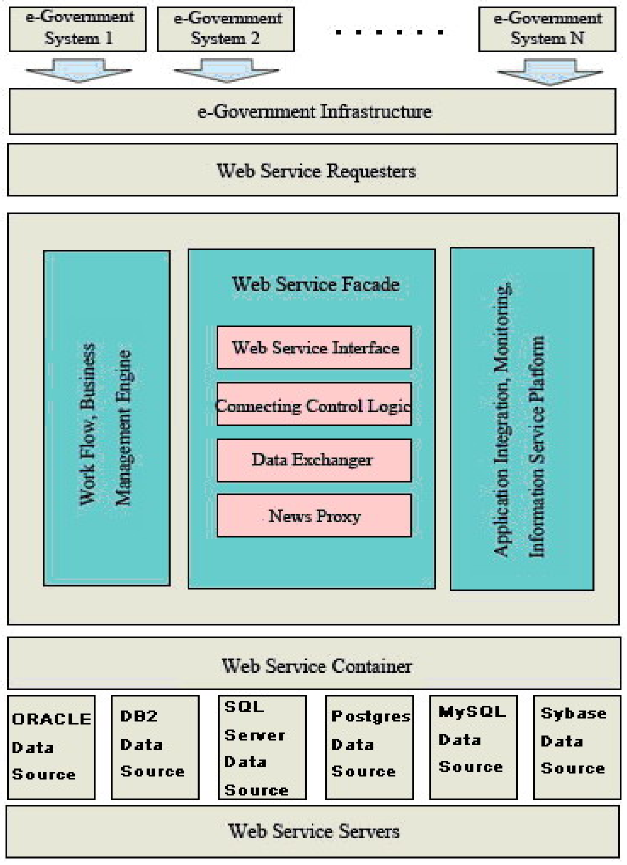

Figure 1. The framework of the data center

III. The Metadata Service

Metadata is the data about other data. Metadata provides a tool and a binding schema for the integration of an information system with distributed, multi-source digital resources info system. Metadata is very important for information publishing and discovery and integrality and consistency maintenance. Using metadata, the data management center can provide effective processing and retrieval. Thus metadata table is the core of the data management center. The administrator can manage the whole system by various applications based on the metadata table, for example, creating the trigger table and modifying the table. The user can subscribe information through metadata and then choose to receive a message or to create a new database [6].

A. *The Structure of Metadata*

In a relational database system, the three-level architecture is used to describe and manipulate data. This architecture is divided into external model, conceptual mode, and internal mode. The three models represent three abstraction levels for data, which leaves the physical organization of data to the DBMS so that the user can logically handle the data and does not need to care the storage and representation of the data. DBMS provides two levels of images between the three models that make data have more logical and physical independence [4].

Based on the data model architecture in RDBMS, the structure of the metadata for the data management center is designed in a hierarchical architecture. The metadata is divided into physical layer, logical layer, and view layer.

- At the physical layer, there are two types of metadata: the metadata for the database system includes the URL, driver, and querying language of the database; and the metadata for the data set, that is metadata for the tables of the database, includes the attributes of the tables and the types of the attributes.
- The metadata for the logical layer aims at special requirements of the data management center, which gets the metadata of physical layer through data model mappings to hide the lower layer difference between different data sources. When the data model changes at the physical layer, the logical layer do not need to be modified. This helps to achieve access transparency.
- The metadata for the view layer provides local information users can view and use, which is the logical representation of the metadata related to a certain application. The metadata for the view layer is usually a subset of the metadata for the logical layer. One metadata for the logical layer can have multiple view layers. Because each user can only see and access the corresponding data, the view layer can guarantee data security.

B. *Implementation of Metadata Service*

The administrator invokes metadata service to get the metadata of each data source. Considering the hierarchical structure of the metadata of each database, XML is used to represent metadata and to create metadata table. The metadata table is the information hub of the entire data management center, which plays an important role to the smooth operation of the system. In order to help the user easily find and use, the data management center must provide a function for accessing the metadata. There are two methods for storing XML files. The first one uses CLOB supported by new RDBMS. JAVA API inserts and retrieves CLOB that only supports document in and document out operations, but does not support the retrieval of part of the documents or nodes in the DOM tree. The second one uses native XML database system, which allows convenient retrieval and modification of the elements created using XML in the metadata table [7].

The XML files are managed by Apache Xindice database management system. Xindice is a database management system of Apache Software Foundation, which provides command-line interface to manage the database and Java API, stores XML files, executes retrieval operations similar to that of database on the nodes, and retrieves part of the document or the entire document.

As an example, in a data management center for user registration, the URL of the registration module is transferred to the metadata service. And then the required metadata is extracted and transformed into an XML formatted string, xmlstring, by invoking the getDatabaseXML. Through the document object model (DOM), the xmlstring is parsed into a DOM document and a logical metadata table is created. The corresponding program is shown in Figure 2.

```
<? xml version= 1.0 encoding= UTF-8 ?>
-<MetadataInfo>
   +<Sehema name= rm>
      - <Schema name=register )
      - <Database name=register )
         - <SystemInfo>
         <Location> 192.168.100.149:1433:register</Location>
         <Driver>con. Microsoft.jdbc.sqlserver.
            SQLServerDriver</Driver>
         </SystemInfo>
         - <Table name= account )
         <Number type= varchar(10) />
         <Name type= varchar(64) ./>
         +< Registration Time type= varchar(20)
         < Host city type= varchar(20)-/>
         +<Account Number type=”int(10) >
         <Keyword type=”int(10)-/>
         </Table>
</MetadataInfo>
```

Figure 2. Program for user registration

The table *MetadataInfo* contains metadata of the database. For each database, there are two elements, *SystemInfo* and *Table*. SystemInfo contains the URL and driver information of the database. *Table* contains the name and type of the attributes of each table. If a user wants to subscribe the information in the table, the attributes of the table should include the information of the user. If this table has subscribed other information, the attributes should include subscription information.

The metadata table presents a uniform virtual view of the resources to applications or the end user. The user can use the metadata table through the interactive interface to find and search the information of each database and can subscribe required data. Thus when changes occur in the subscribed data, the subscriber is informed by the data center of the changes based on the subscription information.

The auxiliary trigger table is created on the basis of the logical layer metadata, as shown in Figure 3.

```
<? xml version =1.0 encoding UTF-8 ?>
-<MetadataInfo>
   <TriggerInfo>
   + <Schema name= FeeInfo>
   </TriggerInfo>
   -<TriggerInfo>
   -<Schema name= FeeInfo>
   <Database name="FeeInfo >
   _+<SystemInfo>
   -<Table name=”FeeInfo”>
   -<Cost type type= int(10) >
   <Subscriber> Cost type
   </Subscriber>
<Subscfber>info.Level pay
</Subscriber>
</Table>
</Database>
</Schema>
</triggerInfo>
</MetadataInfo>
```

Figure 3. The metadata information table

The trigger table contains the subscription information of each database. The trigger table is sent back to the data sources according to the subscription information of the data sources generated by the metadata table. The data sources can create the database trigger based on the trigger table. For example, if a user wants to get the subscription information in the Fee Information table about the name attribute and the charging level, he/she can send this request to the metadata operation through the interactive interface. The metadata operation invokes the method *addSubscriberNode* to insert the subscriber into a proper position.

IV. Database Access Services

In this system, the data access service is mainly responsible for accessing the database. When the data access service is published, there will be a fixed URL with which users can invoke this service through the database access application of the application server.

A method for service generation is provided that the database access application accepts two parameters, the URL and the querying string as shown in the first part of Figure 4.

The codes for accessing the service factory in the client are shown in the second part of Figure 4.

An example for factory creation is shown in the third part. Firstly, the method *createService* is invoked which is a member function of *factory*. The method returns a *locator* variable which can be used to get reference of *dataQuery PortType*.

The last line of Figure 4 shows the code for data access service.

```
URL url=new java,net,URL(args[0])
String queryStr= args[1]

ServiceLocator = new ServiceLocator();
Factory factory=locator.getFactoryPort(url);
ServiceFactory dataQueryFactory = new ServiceFactory (factory);

LocatorType locator=dataQueryFactory.createService();
DataQueryServiceLocator dataQueryLocator = new
   dataQueryserviceLocator();
DataQueryportType dataQuery
   dataQueryL0cat0r.getdataQueryservice(1ocator);

ResuhSet rs=dataQueryexecuteQuery(queryStr);
```

Figure 4. Database access service

V. Data Change Notification Service

The steps for data change notification service are as follows:

- Create a data change table. One data change table is created for each data source. When changes occur in the subscribed data source, the corresponding trigger is started to use the related write procedure in order to write the related SQL statement and the time of the changing into the change table.
- Invoke the data change notification service. When there occurs a new data change in the data change table, the data change notification service is invoked and a new SQL statement is assigned to *ServiceData:java.sql.Data*. Then the data change notification service sends *sqlData* to the daemon process of the data management center which stores the change information in XML format and writes it into a proper location in the metadata table.
- Create the modification table. When the change information is being written into the modification table, the modification table is created and is sent to the corresponding subscribers. The subscribers parse the original SQL statement into the required SQL statement and to modify the corresponding data. When a subscriber receives the modification table, he/she returns an acknowledged signal to tell the system to delete the related element from the metadata table. This change notification service has the features of heterogeneity, universality, and real-timeness.

VI. An Application Example

The access procedure to the data management center can be illustrated by an example, the creation of a new database. A database of new employees of an organization is to be created, the attributes of which include the employee number, name, title, salary, sport item, and sport record. Employee number and name are known for each employee; the title can be obtained from the administration department; salary can be obtained from accounting department; sport item and record have to be entered during the creation. SQL server 2000 is used by the administration and accounting departments and new databases use MySQL database management system.

- The user looks for the metadata of a database in the virtual view and chooses the required attributes through the interactive interface. Here, the salary attribute of the *salaryinfo* table and title attribute of the *titleinfo* table are chosen.
- The request of the user is sent to the new database creation module by interactive interface. This application module creates the three entries of information by accessing the logical metadata table. The three entries of information are shown in Figure 5. The character strings after *DatabaseName* are the username of the database, the password, and the name of the table, respectively. The application module creates a MySQL database named *mydata* and a table named *newdb* according to the first entry of information, and then stores the remaining two entries of information in order to allow later access to the related data source.

```
jdbc:mysql://localhost:1433:DatabaseName=mydata ff ff newdb
   Number VARCHAR 10
   Name VARCHAR 10
   Positions VARCHAR 10
   Salary INTEGER NuLL
   Involved project VARCHAR 10
   Project Score VARCHAR 10

jdbc:microsoft:sqlserver://192.168.100.247:1433:Database Name=xgb
xgb xgb titleinfo Positions

jdbc:microsoft:sqlserver://192.168.100.149:1433:Databasename=jwc
jwc jwc salaryinf salary
```

Figure 5. Table of the entries of information

- The user enters employee number. The new database creation module invokes data access service of each database respectively, and then the data access service gets required data based on the employee number from the databases.
- The data management center gets the data from the databases. Then the new database creation module writes the data into the new database, *mydata*.

VII. Conclusion

The data management center based on Web services is a solution to exchanging data between distributed heterogeneous databases, which uses the service factory concept of Web service and implements three services based on the hierarchical metadata of the information system. The metadata for the description of resource structure, content, and accessing methods is very important to achieving transparently accessing the resources. A three-level metadata model is designed and an example application is implemented. XML is used as the description language and Xindice is used as the database management system. This Web service based data management center can solve the problem with data exchange between different departments and can conveniently create new databases with existing data sources.

Acknowledgment

This work is supported by the National Natural Science Foundation of China under Grant No. 60673159 and No. 70671020; the Key Project of Chinese Ministry of Education under Grant No. 108040; Specialized Research Fund for the Doctoral Program of Higher Education under Grant No. 20060145012 and No. 20070145017.

References

[1] Aaron E. Walsh, "UDDI, SOAP, and WSDL: The Web Services Specification Reference Book", April 9, 2002.

[2] Brahim Medjahed and Athman Bouguettaya, "Customized Delivery of E-Government Web Services", November/December, 2005.

[3] Andreas Tolk and Saikou Y.Diallo, "Model-Based Data Engineering for Web Services", July/August 2005.

[4] J. J. Hanson, ".NET versus J2EE Web Services. A Comparison of Approaches," January 9, 2002.

[5] Gunjan Samtani and Dimple Sadhwani, "EAI and Web Services: Easier Enterprise Application Integration," http://www.webservicesarchitect.com/content/articles/samtani01.asp, June 2006.

[6] Sandip H. Mandera, "Web Services and UDDI: The new wave of E-Business Renaissance," http://www.devx.com/xml/articles/sm100901/sm100901-3.asp.

[7] Scott Seely, "Documenting Your Web Service," http://msdn.microsoft.com/en-us/library/aa480506.aspx, July 2001.