The Research and Design of Object Container Technique

Hai-yan Wu, Xin Meng
Center for Space Science and Applied Research
Chinese Academy of Science
Beijing, China

Hai-yan Wu
Graduate University
Chinese Academy of Science
Beijing, China

Abstract—Object-oriented models cannot fit very well with relational models. This mismatch leads to time-consuming work for designers when saving and storing objects in relational databases. To solve the problem, this paper put forward data class object container method. This object container encapsulates direct access to the database in a base class and provides generic interface services. Objects derive from this base class to form specific data classes, and therefore, the data class container is created. By this means, the coupling between object models and the relational database can be greatly reduced.

Keywords- object/relational mapping; relational database; persistence layer; object container; data class

I. INTRODUCTION

It is a more and more common practice to use object-oriented technique to resolve complex issues and establish data model in the process of sophisticated software system development. Generally speaking, the system uses relational database as platform to define various database tables, and it provides access to multiple program to save objects. The mismatch between the object models and relational models requires the software designer to map the design across them. As the design goes into details and requirements changes, a series of object model modifications come about, which involves all programs using these database tables. Concerning the whole software system, these modifications might cause every program to make a certain amount of changes, just like a “program earthquake”. To minimize the impact, it becomes necessary to study a feasible and reasonable object relational mapping method.

The object container is a “container” that holds objects. The object relational mapping method is used to implement all object related operations, and it is actually setting up a persistent layer above the relational database, which maps the object-oriented data model and relational data model in order to solve the mismatch between them. Moreover, it reduced the coupling between object models and the relational database.

At present, Hibernate[2] is an object mapping engine, supporting JAVA, which uses reflection mechanism. It generates SQL codes when the system is started and does persistent management on the objects[3]. It provides not only the mapping from JAVA class to data tables, but also the query and reset mechanism of the database.

However, in respect of C++ language, there is lack of mature products to sustain persistent object. Thereby, this paper is to study C++ data class object container so as to realize object relationship mapping.

II. ANALYSIS ON OBJECT DATABASE PERSISTENCY METHOD

Object database persistency refers to saving the object in database and completing operations on the object via database.

A simple and common approach, as shown in figure 1, is embedding SQL directly into transaction logic to complete object mapping and saving in the relational database.

The advantage of this method is due to its easiness and high efficiency in coding. However, this method is only appropriate for small program development. The disadvantage is that the database models are tightly coupled with object models in transaction logic, which means that the source codes need to be changed whenever the database model has a minor change, such as altering the name of a column. Accessing the database by writing SQL codes directly in transaction logic will cause great influence on the system software when requirements change, and that will greatly reduce its maintainability and extensibility.

In the process of software development, it is unavoidable to revise object persistency code since the requirements become
more detailed and contents alter. And also, heavy workload and brings risk on credibility of the software if embedding SQL in transaction logic.

Therefore, this brings out the need of a new method of object persistency, that is when the requirements or design changes, only by modifying a small part of the codes, one can achieve relative object persistency changes. In real project, we optimize the object persistency by means of data class object container.

III. DESIGN ON PERSISTENCY LAYER OBJECT CONTAINER

A. Overall concept

Encapsulate the SQL codes used to process (create, delete, modify, and query) objects in an object container base class. Specific objects derive data class from this base class to form data class object container. The principle of data class object container is shown in Figure 2.

![Figure 2. The method of data class object container](image)

Data class object container is based on Windows dynamic link library. Other software and module in the system will store and access objects by means of interface provided by the container. The data class object container accept queries from other software, and the real relational database operation is executed according to the object and relational data mapping, then return the result to the querying software or module.

![Figure 3. Traditional call procedure of object container](image)

Theoretically, the object container could run on any platform that supports COM/DCOM/ADO/OLE DB, but mainly Windows in real practice. Surely, database server platform could be Windows, or UNIX, they both could run object container under the support of ADO/OLE DB. As shown in Figure 3, the external program calls the object container interface to get object data; then the object container will access the relational database server in Windows or Unix platform to complete object relation mapping.

B. Data class object container object relational mapping

The object container is operated on relational data base on which only the relational data is saved. To save and access the object data in application layer component in relational data base via object container, it is necessary to build mapping between object data and data saved in relational base tables. Hence, designing relational database model is the first task to do, that is, the structure of relational table.

The method of data class object container fulfilling relational mapping is to map the object to a relational base table. The object attribute is mapped to a segment in relational base table, others are mapped similarly. The following is an example of a mapped table in Oracle database, which includes text objects, video objects and image objects.

Text object CText, Video object CVideo, and image object CImage are defined in C++ program as follows:

```cpp
class CText // Text Object
{
    public:
        CString m_sTitle; // title of text
        CString m_sKeyword; // keywords of text
        int m_iLength; // size of text
        char* m_pContent; // content of text object
}

class CVideo // Video Object
{
    public:
        CString m_sFormat; // format
        int m_iPlayTime; // length of video object
        CString m_sFrom; // where is it from
        CString m_sRemark; // remark of video object
        BYTE* m_pContent; // content of video object
}

class CImage // Image Object
{
    public:
        CString m_sName; // name
        CString m_sFormat; // format
        int m_iHeight; // height
        int m_iWidth; // width
        BYTE* m_pContent; // content of image object
}
```

Database ER model mapping the text object, video objects and image object to relational database is shown in Figure 4. Considering the amount of video and image data, they are saved in a relative sub-table in order to enhance retrieval performance.
C. Database object container interface design

Data class object container is implemented as a dynamic data component, which is to be used by other intermediate layer and application layer clients via dynamic database interface function. Figure 5 shows the interface provided by Data class object container base class COBJECTContainer. Text object container CTextCon, video object container CVideoCon and image object container CImageCon, are subclasses, all derived from COBJECTContainer, inheriting the interface service in base class. The object container provides all object operation service so that other software and component in the system could call them when needed.

Figure 5. the interface provided by COBJECTContainer

Main function design of each interface:

- **OpenDBCConnection**: This interface has to be called first when using the object container. It implements the startup of a session to a database and the connection to the relational data base. Before startup, object container has to set up the connection with relational database, transaction operation, transaction initiation, submission, and rollback support. At present, the support of object container on transaction operation depend on ADO/OLE DB/ODBC, while support of ADO/OLE DB/ODBC on transaction is based on the connection with relational database, which only allows one transaction carried out at a time per connection.

- **CloseDBCConnection**: This interface has to be called after the using of the object container is stopped. It is to end the database session, and close the connection to the relational database. The object container has to disconnect with the relational data base and submit transaction operations.

- **InsertData**: This interface could be called after the object container is connected with the relational database. It could pass an object or a group of objects for the insert operation in database.

- **SetSortRule**: After the object container is connected with relational database, this interface has to be called before GetFirstData(), DeleteData(), and UpdateData(). It is called to set up the conditions for other operations in the database.

- **DeleteData**: Before calling this interface, SetSortRule() has to be called first after the object container is connected with the relational database, and set up conditions, then call this interface, remove one or a group of qualified records.

- **UpdateData**: This interface could be called after the object container and relational database is connected and call SetSortRule() to update the database.

- **SearchData**: After the object container and relational database is connected and SetSortRule() is called, and before GetFirstData() or DeleteData() is called, this interface should be called to retrieve qualified records accordingly from the database.

- **GetFirstData**: After the object container and relational database is connected, SetSortRule() and Searchdata() are called, this interface could be called to obtain the first record after query.

- **GetNextData**: After the object container and relational database is connected, SetSortRule(), Searchdata() and GetFirstData() are called, call this interface to obtain the next qualified record.

- **GetErrInfo**: This interface could be called to check the error info if any operation on calling the object container fails.

Beside all the above interfaces, the object container supports direct SQL query as well. This is an exception of object encapsulation, whose purpose is to provide a means of accessing lower level relational database under certain circumstances.

D. Data class object container application

The design on object container purposed in this paper is applied to a comprehensive information processing system. During system development, there is a special team in charge of developing the data class object container component. It is unnecessary for the software designers to have any knowledge of the structure of the relational database. Furthermore, there is no need to write relative codes to access Oracle database. Thus, more effort could be put into software development.
In this software, the database structure has undergone many modifications due to requirement and design changes. However, the workload on software development has reduced to a certain extent because of the application of object persistency method, hence the efficiency is increased.

IV. CONCLUSIONS

The data class object container is based on C++ language. It encapsulates the direct access to database in a container so that the influence caused by database modifications on the transaction source code is minimized. Moreover, the transparency of database structure to class source code is enhanced, and code maintenance become more convenient. In the future, data class object container could be further used in other system as common a component.

REFERENCES


