Object Relational Mapping in Comparison to Traditional Data Access Techniques

Aditya Joshi, Sanjeev Kukreti

Abstract— Speeding Development, abstraction and portability of data access, data access layer encapsulation, cache management, concurrency support, transaction management and isolation, generating automatic code for basic CRUD (Create, Retrieve, Update, and Delete) gave birth to a term Object Relational Mapping (ORM). It is a data access technique used for making automatic business logic layer, data access layer, views and stored procedure.

This paper gives the insights how object relational model is different from traditional data access technique and what are the research areas of Object Relational Mapping in software engineering.

Index Terms— Object Relational Mapping, object model, relational model, traditional data access techniques, impedance mismatch, performance.

1 INTRODUCTION

OBJECT Relational Mapping is a programming technique that is used to convert incompatible types in object oriented programming language, mainly between the database and objects. You can use object relational mapping framework to retrieve these objects and this framework will take care of converting data between incompatible types most object relational mapping tools depends on metadata, so that the object does not need to know what is going in the database and database does not need to know how the data is structured in the application. Object relational mapping provides clean separation of code in an application, and database and application each work with data in its original form.

We can use object oriented programming language like java and C# which creates a virtual object database. They are object oriented because they use the concept of encapsulation, Inheritance, interfaces, polymorphism. The widely used storage solution is relational databases often called traditional database because they have been used earlier based on proved principles.

The combination of object oriented system and relational database is the most common solution for the object relational mapping which have a need for persistent storage. However, in order to use relational database for storing of objects, a translation mechanism from object to relational data is needed such a mapping is often referred to as object relational mapping.

There are free and commercial packages available to perform object relational mapping some of the popular object relational mapping tools are also available. In object relational mapping two models are used: the object model and the relational model that is inspired by the paper of M Fussell [1].

- *Aditya Joshi is currently pursuing masters of technology program in information technology engineering in Graphic Era University, Dehradun India, PH-091-9557004416. E-mail: aditya.joshi.15@hotmail.com*
- *Sanjeev Kukreti is currently assistant professor in computer science and information technology engineering branch in Graphic Era University, Dehradun, India, PH-091-9997226690. E-mail: sanrit2009@gmail.com*

2 TWO MODELS

2.1 Object Model

Object Modeling is an approach to structure information into entities which is called objects. The modeling language used for the visualization of objects model is called unified modeling language [2] UML. It defines a rich set of modeling and graphical notations to visualize them refer Fig. 1 which is created by UML tool named UMlet. Various other tools are also available for creating UML Diagrams.

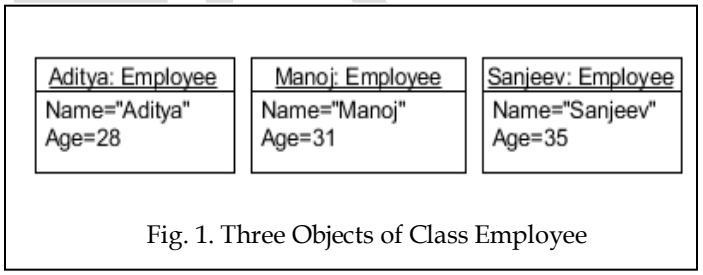

Fig. 1. Three Objects of Class Employee

2.2 Relational Model

The Relational model is developed by E. F. Codd [3] in early 70's and it is based on set theory and predicate logic. The basic concept in relational model is that all concepts are described as predicate and truth statements. It only describes the true things in the real world.

A Relational database is a collection of two dimensional table the data is organized in row and column form and it is called the logical view of the database Fig. 2 illustrates this arrangement.

International Journal of Scientific & Engineering Research, Volume 5, Issue 6, June-2014
ISSN 2229-5518

employee: Employee

Name	Gender	ZipCode	Country
Aditya	Male	248001	India
Manoj	Male	248002	India
Sanjeev	Male	263601	India

Fig. 2. Table View of Relational Variable "employee" of the "Employee"

3 IMPEDANCE MISMATCH

In the previous section two different models are described, a set of conceptual and technical difficulties arises when relational database management system is used by the programme written in object oriented programming language or style is referred as object relational impedance mismatch [4].

3.1 Object Identity

The object and relational model have distinct concept of identity. In the object model identity is the essential part of an object and does not depend on the rest of the model.

In relational model two relation values or rows in relational database are considered to be identical. Object always has unique identity. Two objects in the same state are not viewed to be identical but relations are viewed as a data record with no concept of identity.

3.2 Inheritance

Object of one class may acquire the property of object of another class with the help of inheritance. When object model is transformed to relational equivalent there is no such concept of inheritance.

3.3 Structure, Behavior and Rule of Access

The object model primarily focuses on ensuring that the structure of the model is reasonable whereas a relational model focus on what kind of behavior the resulting runtime system has. Object-oriented methods generally assume that the primary user of the object-oriented code and its interfaces are the developers who develop the application. In relational systems, the end-users view of the behavior of the system is considered as more important.

In the object model all the operations on the objects attribute are performed through methods which are defined in the objects own type. This is called the objects interface and defines how the object can "behave". In the relational model the state can only be accessed or altered by using the relational operators available in the relational model.

4 RESEARCH AREAS

The major Research areas in context of object relational mapping are

- Impedance mismatch (as discussed in previous section)
- Recursive Queries
- Inheritance hierarchies
- Performance of object and object relational database
- Theoretical promises and practical achievements

4.1 Recursive Queries

One of the research fields of interest are recursive queries [5].The first implementations of such queries for SQL has been introduced by Oracle in 1985. However, it was the introduction of recursive Common Table Expressions into the SQL: 99 standards that made the research on this topic more popular. Currently most of the popular DBMS implements recursive queries, but there are no object relational mappers that support such queries. In this paper we propose extending existing ORMs with recursive CTE's support. A prototype of such an extension has been implemented in SQLObject mapper for the Python language. Tests have been conducted with PostgreSQL 8.4 database. Furthermore, recursive queries written using CTEs amount to be too complex and hard to comprehend. Our proposal overcomes this problem by pushing the formulation of recursive queries to a higher abstraction level, which makes them significantly simpler to write and to read.

4.2 Inheritance Hierarchies

We study, in the context of object/relational mapping tools, the problem of describing mappings between inheritance hierarchies [6] and relational schemas. To this end, we introduce a novel mapping model, called M2ORM2 + HIE, and investigate its mapping a hierarchy to relations. We then show that M2ORM2 + HIE also allows expressing further mappings, e.g., where the three basic strategies are applied independently to different parts of a multi-level hierarchy. We describe the semantics of M2ORM2 + HIE in term of how CRUD (i.e., Create, Read, Update, and Delete) operations on objects (in a hierarchy) can be translated into operations over a corresponding relational database. We also investigate correctness conditions.

4.3 Performance of Object and Object Relational Database

Work to compare the performance of object and object relational database systems based on the object's complexity [7], [8], [9]. The findings of this research show that the performance of object and object relational database systems are related to the complexity of the object in use. Object relational databases have better performance compared to object databases for fundamental database operations, with the exception of insert operations, on objects with low and medium complexity. For objects with high complexity, the object relational databases have better performance for update and delete operations.

4.4 Theoretical Promises and practical achievements

This paper tries to point out some of the promises, quarrels, achievements, and perspectives of the forced marriage between the relational and object-oriented data models, from both theoretical and implementation perspectives. As practical O-R achievements, the latest SQL standards (SQL: 1999-2008) and Oracle implementation are discussed [10].

International Journal of Scientific & Engineering Research, Volume 5, Issue 6, June-2014
ISSN 2229-5518

5 TOOLS

There are a number of ORM tools available for .NET applications [11]. Before Microsoft introduced Entity Framework, the open source NHibernate was probably the dominant ORM tool. NHibernate is ported from Hibernate, a Java ORM tool that has been available for years. But because Microsoft now bundles Entity Framework with the .NET Framework and incorporates extensive support for it in Visual Studio, Entity Framework has become the dominant ORM in the Microsoft development world.

Some of the various tools are:

- ADO.NET Entity Framework
- Base One Foundation Component library
- Business Logic tool kit
- DataObjects.NET
- Dapper
- EntitySpaces
- MyBatis
- LINQ to SQL
- NHibernate
- nHidrate
- Persistor.NET
- EasyObjects.NET

6 FUNCTIONALITY

It is used for application's business logic layer and data access layer, includes support for transaction, null value handling and standard CRUD operation, support for dynamic query provider.

6.1 Traditional Data Access Techniques

Compared to traditional techniques of data exchange between an object oriented language and a relational database, ORM reduces the amount of written code. The disadvantage is the high level of abstraction that is what is actually happening in the implementation code with the use of ORM tool.

6.2 Common Tasks

There are some basic tasks done by the object relational mapping the queries are quite different from the traditional way of accessing the data The Comparison of Queries are in the Table below

TABLE I
BASIC QUERIES USING OBJECT RELATIONAL MAPPING IN COMPARISION TO TRADITIONAL DATA ACCESS QUERIES

Object Relational mapping Queries	Traditional data access Queries
Employees emps=new Employees(); int EmpID;	Create Table in Database
emps.LoadAll();	Select * from Employees
emps.LoadByPrimaryKey(EmpID)	select * from Employees where EmployeeID=1

emps.AddNew();
emps.FirstName="Aditya";
emps.HireDate=DateTime.Now();
emps.Save();
empID = emps.EmployeeID; (emps returns new key value)
	insert into Employees (FirstName, HireDate) values ("Aditya", GetDate())

emps.MarkAsDeleted(); emps.Save();	delete from Employees

emps.Where.EmployeeID.Value=empID;
emps.Query.Load();
emps.MarkAsDeleted();
emps.Save();
	delete from Employees where EmployeeID=1

emps.Where.EmployeeID.Value=empID
emps.Query.Load();
emps.LastName = "Sanjeev";
emps.Save();
	update from Employees set LastName= 'Sanjeev' where EmployeeID=2;

6.3 Data Binding

The data binding task is shown in table below for both by ORM and Traditional data access technique

TABLE II
DATA BINDING TO GRIDVIEW

Employees emps = new Employees(); emps.LoadAll(); GridView.DataSource= emps.DefaultView; GridView.DataBind();	SqlConnection conn = new SqlConnection(conectionstring); conn.Open(); SqlCommand command = new SqlCommand("SELECT * FROM Employees); SqlDataReader dr = command.ExecuteReader(); GridView.DataSource = dr; GridView.DataBind();

7 BENEFITS

There are a number of benefits to using an ORM for development of data based applications and here are four:

- Productivity
- Application Design
- Code Reuse
- Application Maintainability

7.1 Productivity

The data access code is usually an important portion of a complex application, and the time needed to write that code can be an important portion of the overall development of the application. When using an ORM tool, the amount of code is improbable to be reduced, but the ORM tool generates all of the data access code means business logic layer and data access layer automatically based on the data model you define,

International Journal of Scientific & Engineering Research, Volume 5, Issue 6, June-2014
ISSN 2229-5518

very quickly.

7.2 Application Design

A good ORM tool designed by very skilled software architects will implement effective design patterns that allow you to use good programming practices in an application development. This can help support a clean separation of concerns and independent development that allows parallel, concurrent development of application layers, business logic layer and data access layer.

7.3 Code Reuse

If you create a class library to generate a separate DLL for the ORM-generated data access code i.e. BLL and DAL, you can easily reuse the data objects in a number of applications. This way, each of your applications that use the class library need not have data access code at all.

7.4 Application Maintainability

All of the code generated by the ORM is probably well-tested, so you usually don't need to worry about testing it on large extent. You need to make sure that the what the code have what you need, but a widely used ORM is likely to have code banged on by many developers at all skill levels. Over the long term, you can refactor the database schema or the model definition without affecting how the application uses the data objects means you do not need to know how application is using data Objects.

CONCLUSION

The drawback of using an ORM is performance. It is sure that the code generated by the ORM for accessing data is very complex than you write for an application. This is because ORMs are designed to handle a variety of data access scenarios, far more than any single application is ever likely to use. Complex code result to slower performance, but a skillfully ORM is likely to generate well-tuned code that minimizes the performance impact. Besides, in all but the most dynamic applications the time spent interacting with the database is a relatively small portion of the time the user spends using the application. Nevertheless, we have never found a case where the small performance hit wasn't worth the other benefits of using an ORM. You should surely test it for your data and applications and make sure that the performance is acceptable to the people.

REFERENCES

[1] Mark L. Fussell, "foundations of object-relational mapping," *White Paper, ChiMu Corp*, 1997.

[2] Richard Soley, "Object Management Group," *Unified modeling language (uml) specification*, version 2.0, 2006.

[3] E.F. Codd, "Derivability, redundancy and consistency of relations stored in large data banks," *IBM Research Report*, 1969.

[4] Ireland, C., Bowers, D., Newton, M., Waugh, K., "A classification of object-relational impedance mismatch," unpublished. (Unpublished manuscript).

[5] Burzanska, M., Stencel, K., Wisniewski, P.,"Pushing Predicates into Recursive SQL Common Table Expressions," *In: Grundspenkis, J., Morzy, T., Vossen, G. (eds.) ADBIS 2009. LNCS*, vol. 5739, pp. 194–205. Springer, Heidelberg (2009).

[6] Cabibbo, L, "Objects Meet Relations: On the Transparent Management of Persistent Objects" *In: Persson, A., Stirna, J. (eds.) CAiSE 2004. LNCS*, vol. 3084, pp. 429–445. Springer, Heidelberg (2004).

[7] Vanzyl, P., Derrick, G., Kourie and Boake, A.,"Comparing the Performance of Object Databases and ORM tools"*Proceeding of South African Institute for Computer Scientists and Information Technologists*, pp. 1-11 (2006).

[8] Bhatti, S., Abro, Z., Rufabro, F.: "Performance evaluation of java based object relational mapping tool". Mehran University Research Journal of Engineering and Technology 32(2), 159–166 (2013).

[9] Aleksandra Gruca, Przemysław Podsiadło, "Performance Analysis of .NET Based Object–Relational Mapping Frameworks," In10th International Conference, BDAS 2014, Ustron, Poland, May 27-30, pp. 40-49. Springer International Publishing (2014).

[10] Agarwal, S., "Architecting Object Applications for High Performance with Relational Databases", *OOPSLA Workshop on Object Database Behavior, Benchmarks, and Performance*, 1995, available at: http://wwwdb.stanford.edu/pub/keller/1995/high-perf.pdf

[11] Cvetković, S., Janković, D.: "A comparative study of the features and performance of ORM tools in a .NET environment". In: Dearle, A., Zicari, R.V. (eds.) ICOODB 2010. LNCS, vol. 6348, pp. 147–158. Springer, Heidelberg (2010).