Value Object를 이용한 EJB 엔티티빈의 성능에 관한 연구

°최은희, 이남용
숭실대학교 컴퓨터학과

A Study of the Performance on EJB Entity Bean with Value Object

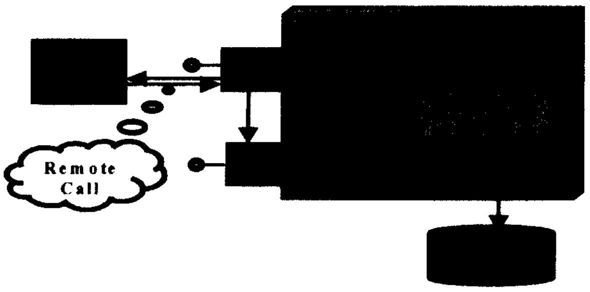
°Eun-Hee Choi, Nam-Yong Lee
Dept. of Computing, Soongsil University

Abstract

In an EJB 1.1 specification, every method call made to the Enterprise Java Bean, is potentially remote call. Such remote invocations use the network layer regardless of the proximity of the client to the bean, creating a network overhead. Especially, because entity bean is more notable performance fall by remote call than session bean, frequency of use on Session Bean in work-site operations is much more than Entity Bean. We focus on how to improve the performance on the entity bean with Value Object, which is one of J2EE patterns suggested by Sun Microsystems. We presents related design-issues for performance testing, the testing results compared with original entity bean and our findings.

1. Introduction

In an EJB 1.1 specification, every method call made to the business service object, is potentially remote call, be it an entity bean or a session bean (see [Figure 1]). Such remote invocations use the network layer regardless of the proximity of the client to the bean, creating a network overhead. As the usage of these remote methods increases, application performance can significantly degrade. Therefore, using multiple calls to get methods that return single attribute values is inefficient for obtaining data values from an enterprise bean [1]. Especially, because entity bean is more notable performance fall by remote call than session bean, we focus on how to improve the performance on the entity bean.



[Figure 1] EJB Entity Bean Architecture

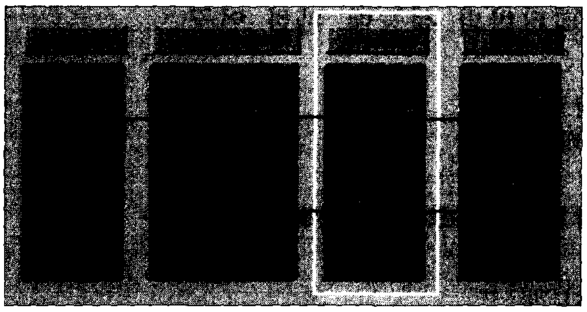
We examine how value object, one of patterns suggested by Sun Microsystems, improve the performance on the entity bean. Also, becaus client requires the data from bean and database for display, and other read only types of processing, the copy of the value object made in clien computer will reduce the number of remote call.

The remaining sections are organized as follows: Section 2 discusses related research. Section 3 presents related design-issues for performanc testing. Section 4 describes the testing results compared with origina entity bean. Section 5 presents our findings and further researchable areas.

2. Related Research

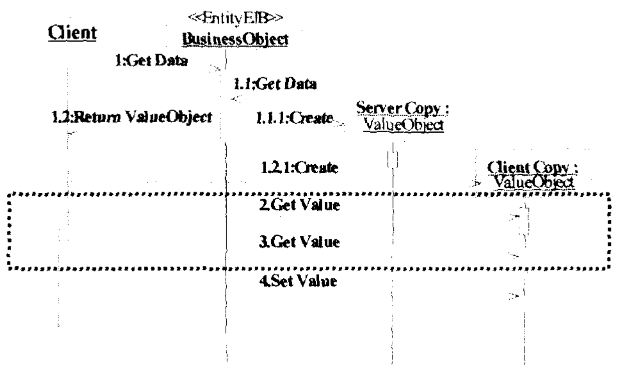
2.1 Value Object

Value Object is One of J2EE patterns suggested by Sun Microsystems. which is a plain serializable java class that represent a snapshot of some server side data (see [Figure 2]). The value object contains and encapsulates bulk data in one network transportable bundle. Typically, a value object is defined as public, thus eliminating the need for get and set methods.



[Figure 2] EJB Architecture with Value Object

The client makes a single remote method invocation to the enterprise bean to request the value object instead of numerous remote method calls to get individual attribute values (see [Figure 3]). Because the value object is passed by value to the client, all calls to the value object instance are local calls instead of remote method invocations. The value object not only carries the values from the Entity Bean to the client, but also can carry the changes required by the client back to the Entity Bean.



[Figure 3] Sequence Diagram of Value Object

3. Design-Issues for Testing

The Scenario for testing is organized as follows: In Internet Medical Prescription System, the *Doctor* (actor) must offer the required profile to become an IMPS member. The required profile items consist of ID, Name, Password, Address, E-mail, Medical License, etc. *DoctorValue* is the value object that accepts all attribute values of *Doctor* entity bean. When *Doctor* Entity Bean is deployed and Client request to inquire a specific information of *Doctor* Table, we experiment Total Time for Query Execution according to Time Measurement Operation in Client code (see [Figure 4]).

Through this case studies, we want to examine the relationships where come from, querying Inquire SQL Query in a single table. Also we examine the relationship between the data size of a value object and the number of remote calls of Entity Bean.

```
long startTime = System.currentTimeMillis();

// . . .
// Client's Business Logic
// . . .

long stopTime = System.currentTimeMillis();

System.out.println("Average Ping = " +
  Float.toString((float)(stopTime-startTime)+"sec");
```

[Figure 4] Testing Code of Client

Kinds of situation for testing are organized as follows: ⓐRelationship between *Doctor11* CMP Entity Bean and *Doctor12* CMP Entity Bean with *DoctorValue1* Value Object, ⓑ Relationship between *Doctor21* BMP Entity Bean and *Doctor22* BMP Entity Bean with *DoctorValue2* Value Object, ⓒ Relationship between *Doctor12* CMP Entity Bean with *DoctorValue1* Value Object and *Doctor22* BMP Entity Bean with *DoctorValue2* Value Object.

4. Testing Results

4.1 Testing Result – Outline Table

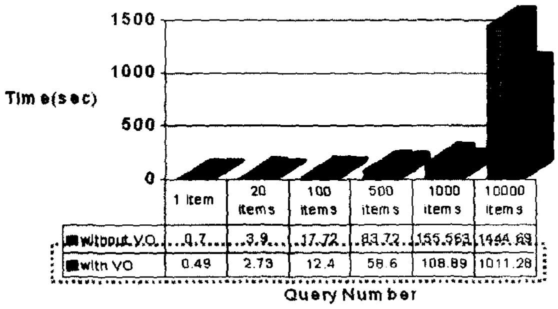
(unit: sec)

Type	CMP Entity Bean		BMP Entity Bean	
Table Type	without VO	with VO	without VO	with VO
1 item	0.7	0.49	0.56	0.42
20 items	3.9	2.73	3.12	2.32
100 items	17.72	12.4	14.18	10.54
500 items	83.72	58.6	66.98	49.81
1000 items	*155.563*	*108.89*	124.45	92.56
10000 items	1444.69	*1011.28*	1155.75	859.59

* *VO : Value Object*

[Table 1] Testing Result – Outline Table

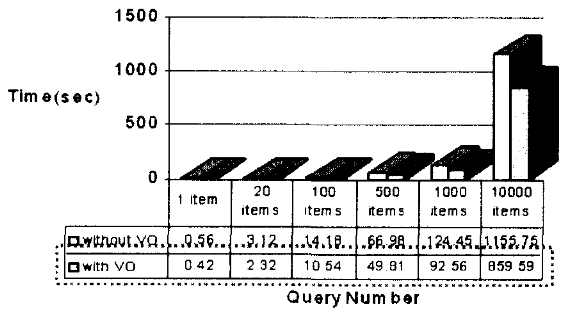
4.2 Testing Result – CMP Entity Bean



	1 item	20 items	100 items	500 items	1000 items	10000 items
■ without V.O.	0.7	3.9	17.72	83.72	155.563	1444.69
■ with VO	0.49	2.73	12.4	58.6	108.89	1011.28

[Figure 4] CMP Entity Bean with VO

4.3 Testing Result – BMP Entity Bean



	1 item	20 items	100 items	500 items	1000 items	10000 items
◻ without VO	0.56	3.12	14.18	66.98	124.45	1155.75
◻ with VO	0.42	2.32	10.54	49.81	92.56	859.59

[Figure 5] BMP Entity Bean with VO

4.4 Testing Result – CMP with VO vs. BMP with VO

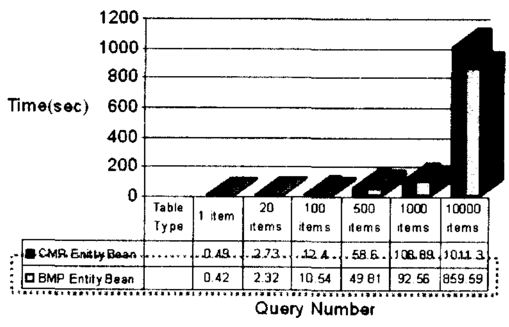


Table Type		1 item	20 items	100 items	500 items	1000 items	10000 items
■ CMP Entity Bean		0.49	2.73	12.4	58.6	108.89	1011.3
◻ BMP Entity Bean		0.42	2.32	10.54	49.81	92.56	859.59

[Figure 6] CMP with VO vs. BMP with VO

5. Findings and Further Researchable Areas

5.1 Findings

5.1.1 Improved the performance of Entity Bean

Because value object acts as a data carrier and reduces the number of remote network method calls, it improves the performance. As testing results, in both CMP Entity Bean and BMP Entity Bean, it takes less the total time for the client query execution with Value Object than without Value Object. Also, in *Doctor22* BMP Entity Bean with Value Object, it takes less the total time for the client query execution than *Doctor21* CMP Entity Bean with Value Object. Consequently, we can expect to improve the performance of Entity Bean with Value Object.

5.1.2 Transfers More Data in Fewer Remote Calls

One method call returns a greater amount of data to the client than each 'get method'. As testing results, original *Doctor11* Entity Bean contains 15 'get methods'. On the other hand, *Doctor21* Entity Bean with Value Object contains 1 'get methods' required to call *DoctorValue1* Value Object. Original *Doctor11* Entity Bean contains 1 Field of *Doctor* DB Table, but *Doctor21* Entity Bean with Value Object contains 15 Fields of *Doctor* DB Table. Consequently, we expect to transfer More Data in Fewer Remote Calls with Value Object.

5.1.3 Simplifies the design of Entity Bean and Remote Interface

Entity bean provides a getData() method to get the value objec containing attribute values. Value Object technique can eliminate some o 'get methods' to implement the entity bean and remote interface

5.2 Further Researchable Areas

We have successfully used the Value Object for improving th performance on Entity Bean, including CMP Entity Bean and BMP Entit Bean.

One direction for further research is to continue gathering experience b applying Performance Testing between multiple Entity Beans wit multiple Value Objects, Performance Testing between Entity Bean an Session Bean, etc. In work-site operations, especially, frequency of use on Session Bean is much more than Entity Bean by reason of distinguished performance. Therefore, it is important to compare these two types of bean

Another important direction of research is to address the Performance Difference between Entity Bean for Read Transaction and Entity Bean for Update Transaction.

Finally, we consider applying the Testing Results of Entity Bean with Value Object in the Enterprise JavaBeans Specification, Version 2.0. In EJB 2.0 specification, because Entity Beans typically are dealt with by local call rather than remote call, the use of Value Object pattern may be meaningless.

Reference

[1] Deepak Alur, John Crupi, Dan Malks, *Core J2EE Patterns*, p.261~290, 2001

[2] Sun Microsystems, *Enterprise JavaBeans Specification*, version 2.0, 2001

[3] Ed Roman, *Mastering Enterprise JavaBeans*, 2000

[4] http://theserverside.com/

[5] http://www.devx.com/upload/free/features/entdev/