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.

Microsystems, improve the performance on the entity bean. Also, because 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 client 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 performance testing. Section 4 describes the testing results compared with original 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 1] EJB Entity Bean Architecture

We examine how value object, one of patterns suggested by Sun
[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.

![Diagram](image)

[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 IMS patient. The required profile items consist of ID, Name, Password, Address, E-mail, Medical License, etc. DoctorValue is the value object that accepts all attributes 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).

![Diagram](image)

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.

![Diagram](image)

[Figure 4] Testing Code of Client

Kinds of situation for testing are organized as follows: 1. Relationship between Doctor1 BMP Entity Bean and Doctor12 BMP Entity Bean with DoctorValue1 Value Object. 2. Relationship between Doctor21 BMP Entity Bean and Doctor22 BMP Entity Bean with DoctorValue2 Value Object. 3. Relationship between Doctor12 BMP Entity Bean with DoctorValue1 Value Object and Doctor22 BMP Entity Bean with DoctorValue2 Value Object.

4. Testing Results

4.1 Testing Result – Outline Table

<table>
<thead>
<tr>
<th>Type</th>
<th>CMP Entity Bean</th>
<th>BMP Entity Bean</th>
</tr>
</thead>
<tbody>
<tr>
<td>Item</td>
<td>0.7</td>
<td>0.49</td>
</tr>
<tr>
<td>20 items</td>
<td>3.9</td>
<td>2.73</td>
</tr>
<tr>
<td>100 items</td>
<td>17.72</td>
<td>12.41</td>
</tr>
<tr>
<td>500 items</td>
<td>83.72</td>
<td>59.66</td>
</tr>
<tr>
<td>1000 items</td>
<td>155.569</td>
<td>108.899</td>
</tr>
<tr>
<td>10000 items</td>
<td>1444.69</td>
<td>1101.28</td>
</tr>
</tbody>
</table>

*VO: Value Object

4.2 Testing Result – CMP Entity Bean

![Diagram](image)

4.3 Testing Result – BMP Entity Bean

![Diagram](image)
Entity bean provides a `getDeta()` method to get the value object containing attribute values. Value Object technique can eliminate some of the 'get methods' to implement the entity bean and remote interface.

5.2 Further Researchable Areas

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

One direction for further research is to continue gathering experience by applying Performance Testing between multiple Entity Beans with multiple Value Objects, Performance Testing between Entity Bean and 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 beans.

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