EJB Persistence Pattern을 이용한 효과 적인 엔티티빈 설계

이돈양°, 이창수, 송영재
경희대학교 전자계산공학과
dylee6211@hanmail.net

Design of Efficient Entity Bean Using EJB Persistence Pattern

Don_Yang Lee°, Chang-Soo Lee, Young-Jae Song
Dept. of Computer Science, Kyunghee University

요약

소프트웨어 산업의 급속한 발전과 더불어 소프트웨어의 생산성의 향상과 재사용 측면이 매우 강조되면서 다양한 방법으로 접근이 되고 있다. 특히 J2EE의 EJB 기반의 컴포넌트 개발은 플랫폼에 독립적으로 운용이 가능한 시스템개발에 큰 영향을 미쳤다. 그리고 디자인 패턴의 개념을 도입함으로써 다시 발생될 수 있는 문제점들을 패턴으로 정리하고 해결점까지 제시하고 있어 소프트웨어의 재사용 측면에 발전을 가져오고 있다. 본 논문에서는 여러 디자인 패턴 중 Dual Persistent Entity Bean 디자인 패턴을 이용하여 엔티티 빈을 설계하는데 BMP와 CMP를 동시에 지원할 수 있는 환경을 이용한 모델을 제시하고 있다.

1. 서론

EJB 기반 컴포넌트 개발은 이미 개발되어 사용가능한 것과 새로운 기능을 가진 컴포넌트를 개발하여 서로 조립하고 재사용하여 빠른 시간안에 성능이 우수한 어플리케이션 및 시스템을 구축하는데 목적을 두고 있다.

EJB는 Sun사에서 제안한 것으로서 Java 2 Enterprise Edition(J2EE)의 기업용 비즈니스 어플리케이션을 개발하고 운용하기 위한 서버측의 컴포넌트 모델이다.[1]

패턴(pattern)의 개념은 건축가인 Christopher Alexander[2]에 의해 1970년에 도입되었다. 그는 일반적인 환경에서 다시 발생할 수 있는 문제들을 패턴으로 정리하였고 이를 해결할 수 핵심적인 답을 제공하여 같은 문제로 발생되는 것들을 해결하는데 사용되었다. 소프트웨어 디자인 패턴은 소프트웨어의 설계와 생산성의 향상에 적용되었다.[3] 이는 소프트웨어 재사용에 유용한 기본형태를 제공하였다.

본 논문에서 소개된 Dual Persistent Entity Bean 디자인 패턴은 EJB 개발자가 엔티티 빈 컴포넌트를 작성할 때 퍼시스턴스를 CMP(Container-Managed Persistence)와 BMP(Bean-Managed Persistence)로 나누어 관리할 수 있는 것을 Superclass와 Subclass로 구분하여 CMP와 BMP를 동시에 지원할 수 있는 엔티티 빈을 작성하는 것이다. 이는 디플로이먼트시 이 두 개중에서 디플로이먼트 디스크립터의 ejb-jar.xml 파일의 설정에 의해 선택되도록 하는 것이다.

Dual Persistent Entity Bean 디자인 패턴을 이용하여 엔티티 빈을 설계하는 것은 BMP에서 CMP로의 변경에서 매우 까다롭고 어려운 방법을 해결함으로써 소프트웨어의 재사용과 생산성 향상에 많은 도움을 줄 수 있을 것이다.

2. 관련연구

2.1 디자인패턴

디자인 패턴(Design Pattern)은 시스템 설계 시 반복되는 문제들을 해결하기 위한 방법의 해결책으로 만들어졌으며, 이는 객체지향 바탕이라 개념화와 다이어그램의 표현이 쉽고 재사용이 간편하다. 또한 디자인 패턴은 하나의 Solution으로, 재사용되거나 새롭게 생성되기도 한다.[4,5,6]

2.2 컴포넌트

컴포넌트는 재사용이 가능한 독립적인 기능을 가진 어플리케이션의 모음으로[5], 컴포넌트는 대형의 어플리케이션 프로그램을 개발하는데 사용되어지는 조립형태의 부품으로 생각하면 된다. EJB기반 컴포넌트(ejb.jar)는 서버측에서 구현되는 리모트 인터페이스(remote interface), 홈 인터페이스(home interface), 빈 클래스(bean class)로 구성되어 하나의 패키지 형태를 유지하고 있다. 그리고 빈(bean)에는 세션빈(session bean)과 엔티티빈(entity bean) 두 종류가 있는데 데이터베이스에 관련된 퍼시스턴스(persistence)의 유무에 따라 타입이 구분되며 <표 1>은 이 두 가지를 비교하여 보여주고 있다.

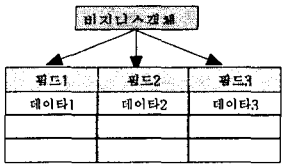
<표 1>

엔티티빈(entity bean)	세션빈(session bean)
공유데이타 생성	공유데이타 이용
클라이언트측 공유가능	클라이언트측 공유불가
데이타베이스생성	데이타베이스생성안함
퍼시스턴스 가짐	퍼시스턴스 갖질않음

그리고 엔티티빈에서는 특정키를 가지고 데이터를 검색하기 위해서 프라이머리 키 클래스(primary key class)를 작성한다.

2.3 엔티티 빈(entity bean)

엔티티빈은 영속적인 저장기술에 대한 비즈니스 객체로 표현되고 있다. 여기서 영속적인 저장기술은 관계데이타베이스(relational database)를 지칭하고 있다. 실질적으로 각 엔티티 빈은 관계 데이타베이스안의 테이블에 연결되어 있으며 빈 인스턴스는 이 테이블의 열 부분에 상호적으로 연결되어 있다.



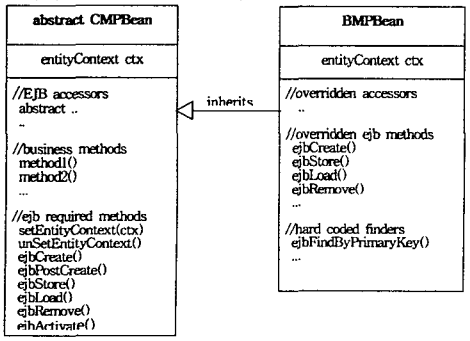
<그림1> 비즈니스객체와 관계데이타베이스

그리고 엔티티빈에서 퍼시스턴스를 관리하는 방법에 따라 CMP(Container-Managed Persistence)와 BMP(Bean-Managed Persistence)의 두가지로 나눈다. CMP는 퍼시스턴스를 EJB 컨테이너가 자동으로 관리하여 프로그램 개발자는 비즈니스 로직에 대한 부분을 전담하고 있다. 데이터베이스와는 독립적으로

운영되어 빈에 대한 응용프로그램간에 있어서 재사용성을 높일 수 있다.
BMP는 CMP와 달리 프로그램 개발자가 직접 퍼시스턴스와 관련된 로직을 명시적으로 작성해야한다. 이는 실제로 데이터베이스에 관련된 SQL문을 처리해야 하므로 CMP보다 복잡하고 까다롭다. 그러나 빈 인스턴스와 데이터베이스와의 관계에서 유연성을 가질 수 있고 복잡하고 다양한 프로그램을 개발하는데 있어서 최적의 빈을 생성할 수 있는 장점을 가지고 있다.

2.4 Dual Persistent 엔티티 빈 패턴

컴포넌트 개발자들을 위한 패턴으로서 Dual Persistent Entity Bean 패턴은 디플로이먼트 기술의 편집으로 한번에 컴파일되고, CMP나 BMP 둘 중의 하나로 디플로이 될 수 있도록 작성하는 것이다[4]. EJB 개발자는 CMP와 BMP 모두를 지원하는 엔티티 빈 컴포넌트를 작성해야 할 때가 있다. 이는 동일한 엔티티빈 컴포넌트에 대해서 CMP와 BMP의 분리된 기반을 제공하는 것이다.
여기서는 비즈니스 로직을 CMP 규약에 만족하는 SuperClass와 BMP의 영속적인 로직을 갖는 SubClass로 구분하여 CMP와 BMP를 동시에 지원하는 엔티티빈을 작성한다. 디플로이먼트 시 둘 중 선택은 디플로이먼트 디스크립터의 설정에 의해 결정된다. <그림 2>은 CMP와 BMP의 이중영속성 엔티티빈의 클래스다이어그램의 형태를 보여주고 있다.[4]

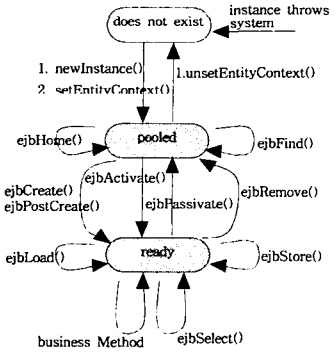


<그림 2>

3. 엔티티 빈 패턴 설계

3.1 컨테이너측 엔티티빈 라이프사이클

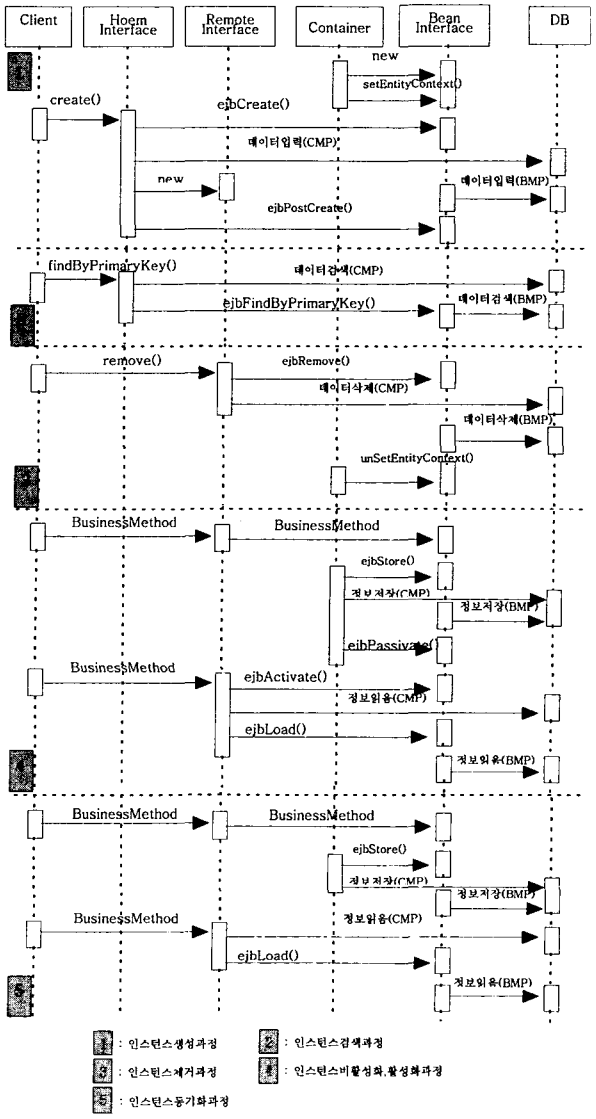
엔티티빈의 라이프사이클은 클라이언트 객체 측에서 본 것과 컨테이너 측에 서 본 것으로 나눌 수가 있다. <그림 3>은 컨테이너에서 메소드의 상태를 알 수 있도록 나타내고 있으며 전체적인 순환과정의 형태를 보여주고 있다.



<그림 3>

3.2 CMP,BMP기반 빈(bean)동작 Sequence Diagram

<그림4>은 EJB서버가 작동하면서 엔티티 빈 인스턴스의 생성과 사용, 제거, 비활성화/활성화, 동기화 과정을 공통적인 작업과 CMP와 BMP의 구분된 작업을 보여주고 있다. 일반적으로 CMP 기반에서는 데이터베이스와의 연결을 컨테이너가 담당하고 있다. <그림2>에서와 같이 CMP 기반에서 ejbLoad()메소드, ejbStroe()메소드, ejbActivate()메소드, ejbPassivate()메소드, ejbRemove()메소드 등은 컨테이너가 자동으로 구현을 해주고 있으나 BMP기반에서는 데이터베이스 연결 뿐만 아니라 위에 해당되는 부분들을 프로그램 개발자가 직접 작성을 해야한다.

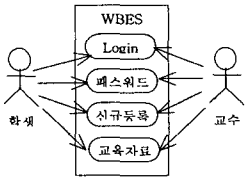


<그림4>

3.3 Dual Persistent 엔티티빈 설계

<그림5>의 UseCase Diagram에 해당되는 <표2>의 빈 프로그램[7] 목록 중 EduBean.java 엔티티 빈 클래스의 비즈니스

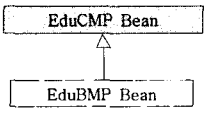
로직을 <그림6>과 같이 CMP에 해당되는 SuperClass와 BMP의 영속적인 로직을 갖고 있는 SubClass로 구분하고 이를 동시에 지원 가능한 엔티티 빈을 작성한다. 둘중의 선택은 디플로이먼트 디스크립터의 설정에 의해 결정할 수 있다.



<그림5>

<표2> 빈 프로그램목록

프로그램종류	이름
리모트인터페이스	EduRmt.java
홈 인터페이스	EduHome.java
CMP 엔티티빈 클래스	EduBean.java
프라이머리키 클래스	EduPK.java
클라이언트입력	EduClient.html
클라이언트 출력	EduClient.jsp



<그림6>

CMP Superclass는 비즈니스 메소드와 abstract 메소드 등 기본적인 것들을 포함하고 있다.Subclass에서는 ejbCreate()와 같이 Superclass에서 간략하게 구현했거나 비워있는 ejbStore(), ejbLoad(), ejbRemove()등을 구체적인 프로그램으로 작성하고 있고 이들은 Persistence에 관련된 메소드 들이다. 그리고 Subclass에서는 setEntityContext(), unSetEntityContext(), ejbActivate(), ejbPassivate() 같은 비즈니스 로직에 해당되는 메소드들은 Superclass에서 상속받기 때문에 다시 작성할 필요가 없다.

3.4 디플로이먼트 작성

CMP와 BMP와 같이 다른 Persistence를 가진 빈을 이용한 컴포넌트의 생성은 디플로이먼트시 ejb-jar.xml 파일을 변경함으로 class를 선택할 수 있다. 즉 <그림7>, <그림8>과 같이 디플로이먼트 스크립터인 <persistence-type> 요소의 변경으로 가능하다.

```
<!-- Edu Deployment Descriptor for CMP -->

<ejb-jar>
 <enterprise-beans>
    <entity>
      <ejb-name>EduDualPersistent</ejb-name>
      <home> EduHome</home>
      <remote>EduRemote</remote>
      <ejb-class>EduCMPBean</ejb-class>
      <persistence-type>Container</persistence-type>
      ...
    </entity>
 </enterprise-beans>
    ...
    <container-transaction>
      ...
    </container-transaction>
</ejb-jar>
```

<그림7> CMP 디플로이먼트 디스크립터

```
<!-- Edu Deployment Descriptor for BMP -->

<ejb-jar>
 <enterprise-beans>
    <entity>
      <ejb-name>EduDualPersistent</ejb-name>
      <home> EduHome</home>
      <remote>EduRemote</remote>
      <ejb-class>EduBMPBean</ejb-class>
      <persistence-type>Bean</persistence-type>
      ...
    </entity>
 </enterprise-beans>
    ...
    <container-transaction>
      ...
    </container-transaction>
</ejb-jar>
```

<그림8> BMP 디플로이먼트 디스크립터

그리고 <그림9>과 <그림10>은 EduCMP Bean Superclass와 EduBMP Bean Subclass 관계에서 상속되어지는 부분의 프로그램작성 부분을 나타낸 것이다.

```
// EduCMP Bean Superclass : EduCMPBean.java

import java.util.*;
import javax.ejb.*;
import javax.naming.*;
import java.sql.*;
import javax.sql.*;

abstract public class EduCMPbean implements EntityBean {
  ...
   public String ejbCreate(num,name,id,sosok,addr,tel)
                throws CreateException
    {
     ...
    }
   ...
   public void ejbActivate() {}
   public void ejbLoad() {}
   public void ejbPassivate() {}
   public void ejbPostCreate(num,name,id,sosok,addr,tel) {}
   public void ejbRemove() throws RemoveException {}
   public void ejbStore() {}
}...
```

<그림9> CMP Bean Superclass

```
// EduBMP Bean Subclass : EduBMPBean.java

import java.util.*;
import javax.ejb.*;
import javax.naming.*;
import java.sql.*;
import javax.sql.*;

public class EduCMPbean extends EduBMPBean implements        EntityBean {
  ...
   public String ejbCreate(num,name,id,sosok,addr,tel)
                throws CreateException
    {
     ...
    }
   ...
   public String ejbFindByPrimaryKey() throws
    ObjectNotFoundException
    { ... }
   public void ejbLoad()
    { ... }
   public void ejbPostCreate(num,name,id,sosok,addr,tel) {}
   public void ejbRemove()
    { ... }
   public void ejbStore()
    { ... }
}...
```

<그림10> BMP Bean Subclass

4. 결론

EJB 컴포넌트의 엔티티 빈 생성에서 재사용을 통한 빈의 조립을 할 수 있다. 이것은 BMP 엔티티 빈들을 CMP 엔티티 빈으로 변경하는 경우인데 프로세스를 작성하는 것이 까다로워 CMP로 재작성하는 것이 쉬운 방법일 수도 있었다. 그러나 Dual Persistent 엔티티 빈 패턴을 사용하여 Superclass와 Subclass로 구분하여 메소드를 관리함으로써 BMP에서 CMP로의 변경을 수월히 함으로써 컴포넌트 기반에서 재사용을 향상시킬 수 있었다. 본 논문에서는 EJB 컴포넌트 생성 및 재사용에서 Dual Persistent 엔티티 빈 패턴을 적용함으로써 최적의 프로그램구현에 그 목적을 두고 있다. 향후 연구로는 실질적인 구현을 통하여 여러 다른 EJB 디자인패턴과 비교와 성능평가에 대한 연구가 이루어져야 할 것이다.

참고문헌

[1] Heineman Councill, "COMPONENT-Based software engineering", Addison wesley, 2001
[2] C, Alexander, A Pattern Language, NewYork : Oxford University Press, 1977
[3] F.Buschmann, R. Meunier, H. Rohnert, P. Sommerland, and M. Stal, "Pattern Oriented Software Architecture, A System of Patterns", Wiley, 1996
[4] Floyd Marinescu, EJB Design Patterns, Wiley, 2002
[5] Krishnan Subramanian, EJB Design Patterns : Designing EJBs for maximum re-rsability, compactness and flexibility
[6] Eric Gamma, Richard Helm, Ralph Johnson, John Vlissides, "Design Patterns : Element of Reusable Object-Oriented Software", Addison-Wesley, 1995
[7] 이돈양, 송영재, "EJB기반 웹 컴포넌트 모델분석 및 설계에 관한 연구",대한전자공학회, 2002,하계학술대회 논문집