Enterprise JavaBeans를 사용한 웹기반 도서 대출 관리

서봉근* · 김윤호**

A Web based Book Lending Management System using Enterprise JavaBeans

Bong-kun Seo* · Yun-ho Kim**

이 논문은 2004년도 인천대학교 멀티미디어연구센터에서 연구비를 지원받았음.

요 약

본 논문에서는 분산 환경에서 엔터프라이즈 애플리케이션을 빠르고 쉽게 개발하기 위해 썬사에서 제안한 Java 기반의 EJB를 사용하여 로컬과 웹에서 따로 운영되고 있는 도서 대출 관리 시스템을 통합하는 기법을 제시한다. EJB를 사용하여 따로 운영되는 도서 대출 관리 시스템을 통합하고 기능별로 분리하여 설계/구현함으로써 유지/보수의 비용을 절감 할 수 있으며 JSP를 사용하여 웹 기반 인터페이스를 제공함으로써 인터넷이 가능한 모든 플랫폼에서 서비스를 제공한다. 또한 EJB와 JSP를 사용함으로써 플랫폼에 독립적으로 구현이 가능하다.

ABSTRACT

In this paper, we propose the method of integrating the Book Lending Management System running either on local and web to develop Enterprise Application in distributed environment more easily and quickly using Sun's EJB based on the Java. We integrate the Book Lending Management System which is operated separately using EJB. And, by designing and implementing the Book Lending Management System separately according to its functions, it can reduce the cost of maintenance and repair. At all platforms where the Internet, Book Lending service is possible, because it provide Web based interface with JSP. Then, we can get platform independent implementation using EJB and JSP..

키워드

분산 환경, 엔터프라이즈 애플리케이션, EJB, JSP

Ⅰ. 서 론

엔터프라이즈 애플리케이션을 개발하기 위해서는 물리적으로 떨어져 있는 다른 시스템들을 네트워크에 연결하여 자원을 사용할 수 있는 분산 환경을 구축하는 것은 필수이다. 이러한 분산 환경에 산재해 있는 여러 자원을 관리하고 보안과 트랜잭션 등을 처리하는 것은 하나의 시스템에 운영되는 프로그램을 만드는 것에 비해 매우 복잡하며 어렵다[1].

엔터프라이즈 애플리케이션 개발에 최근 주목받는 기술은 Sun Microsystems에서 제시한 EJB가 있

*안동대학교 컴퓨터공학과 석사과정
**안동대학교 전자정보산업학부 부교수

접수 일자 2004. 12. 10

다. EJB는 성능 및 확장성에서 오랜 시간 성숙되고 입증되어 보안성과 신뢰성이 뛰어나며 Java 기반의 기술이므로 플랫폼에 중립적인 장점을 가지고 있다[2][3]. 그로 인해 현재 많은 회사들이 EJB를 사용하여 엔터프라이즈 애플리케이션을 개발/서비스하고 있다. 이러한 추세와는 달리 현재 운영되고 있는 도서 대출 관리 시스템은 로컬과 웹에서 따로 운영이 되어 각각의 개발과 유지/보수하는데 어려움이 따르며 그에 따른 비용이 증가하는 등의 문제점을 가지게 된다. 따라서 이와 같이 분리되어 운영되고 있는 시스템을 하나로 통합하여 개발하고 운영하는 방법이 요구된다.

본 논문에서 제시하는 웹기반 도서 대출 관리 시스템은 로컬과 웹에서 따로 운영되고 있는 현재 도서 대출 관리 시스템을 EJB를 사용하여 하나로 통합하고 JSP[4][5]를 사용하여 웹기반으로 모든 서비스를 제공하는 것을 보여 준다. Java 기반의 EJB와 JSP를 사용함으로써 플랫폼의 중립성을 제공하고 각 기능별로 모듈을 분리하여 설계/구현함으로써 유지, 보수의 비용을 최소화 할 수 있게 된다.

II. 시스템 요구 사항

도서 대출 관리 시스템을 구현하기 위해서는 도서의 대출과 관리를 담당하는 관리자를 위한 기능과 대출 정보와 대출 연장과 같은 이용자를 위한 기능이 나눠지며 각각의 요구사항 및 필요한 기능은 아래의 표와 같이 정의 한다(표 1-1, 1-2 참조).

표 1-1. 관리자 요구 사항
Table. 1-1 Administrator requirement

요구 기능	입 력
로그인	관리자 ID, 패스워드
계정 생성	학번, 이름, 패스워드, 학과
계정 삭제	학번
계정 정보 수정	이름, 패스워드, 학과
계정 검색	학번
책 추가	도서명, 저자, 출판사, 위치
책 삭제	도서 번호
책 정보 수정	도서명 ,저자, 출판사, 위치
책 검색	도서명, 저자, 출판사
책 대출	도서 번호
책 반납	도서 번호
연체자 확인	-

표 1-2. 이용자 요구 사항
Table. 1-2 User requirement

요구 기능	입 력
로그인	학번, 패스워드
책 검색	도서명, 저자, 출판사
개인 정보 변경	이름, 패스워드, 학과
대출 조회/연장	-

표 1-1은 도서 대출 관리를 위한 관리자의 요구사항을 정의한 것이다. 책을 빌려주기 위한 이용자의 계정 생성, 삭제, 수정과 검색, 대여를 위한 책을 추가, 삭제, 수정, 검색 그리고 대여와 반납 기능, 마지막으로 대출 기간을 초과한 연체자를 확인하는 기능을 포함한다. 표 1-2에 나타낸 이용자의 요구 사항은 책 대출을 위하여 책의 위치 및 대출 여부를 확인하기 위한 책 검색, 개인의 정보 변경과 대출한 책을 조회하고 연장하기 위한 대출 조회/연장 기능이 포함된다. 공통적으로 로그인 기능이 포함되며 ID를 확인하여 관리자와 이용자를 구별하여 접속을 하게 된다. 또한 도서 대출 관리에 필요한 이용자의 정보, 도서 정보, 대출 정보를 저장하기 위한 데이터베이스의 테이블 구성은 아래와 같이 정의한다(표 2-1, 2-2, 2-3 참조).

표 2-1. 이용자 정보 테이블(USER_TABLE)
Table. 2-1 STUDENT_TABLE

Column명	Type	비 고
SNUMBER	VARCHAR2	학번
SNAME	VARCHAR2	이름
SPASSWORD	VARCHAR2	패스워드
SMAJOR	VARCHAR2	전공

표 2-2. 책 정보 테이블(BOOK_TABLE)
Table. 2-2 BOOK_TABLE

Column명	Type	비 고
BNUMBER	NUMBER	도서 번호
BNAME	VARCHAR2	도서명
BAUTH	VARCHAR2	저자
BPUBLISHER	VARCHAR2	출판사
BLOCATION	VARCHAR2	위치
BRENT	CHAR	대출 여부

표 2-3. 대출 정보 테이블(RENT_TABLE)

Table. 2-3 RENT_TABLE

Column명	Type	비 고
BNUMBER	NUMBER	도서 번호
SNUMBER	VARCHAR2	학번
RENTDATE	DATE	대여일
RETURNDATE	DATE	반납일
EXTENSION	CHAR	연장 여부

위 표 2-1에 기술된 STUDENT_TABLE의 SNUMBER(학번)와 표2-2에 기술된 BOOK_TABLE 의 BNUMBER(도서 번호)는 주키(primary key)가 되며 BNUMBER는 도서의 정보를 저장할 때 자동으로 번호가 증가하는 시퀀스(sequence)를 적용하여 주키의 중복을 사전에 차단한다. 표 2-3에 기술된 RENT_TABLE은 대출 정보를 저장하는 테이블로써 STUDENT_TABLE의 SNUMBER와 BOOK_TABLE의 BNUMBER를 각각 외래키(foreign key)를 가지고 있다. 도서 반납일은 대여일로부터 15일로 제한하며 연장을 하였을 경우 다시 15일을 더하게 된다. 이 반납일이 현재 날짜를 초과한 경우 연체가 된다. 그림 1은 이와 같이 정의된 테이블 간의 관계를 ER(Entity Relation) 다이어그램으로 표기한 것이다. 대출 정보와 이용자 객체의 관계는 대출 정보 테이블에 여러 명의 이용자가 도서를 대출한 이용자 정보를 저장할 수 있으므로 일대다의 관계가 성립되며 대출 정보와 도서 정보 객체의 관계 역시 대출 정보 테이블에 여러 권의 도서를 저장하게 되므로 일대다로 정의 된다.

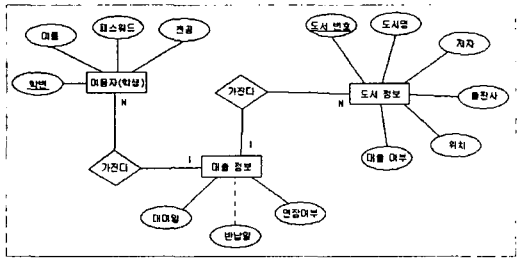

그림 1. ER 다이어그램
Fig. 1 ER(Entity Relationship) diagram

III. 시스템 설계 및 구현

분산 환경에서 데이터베이스에 저장되어 있는 영속성(persistence)을 유지하는 데이터를 표현하기 위해 일반적으로 EJB에서는 보안(security), 트랜잭션(transaction), 동시성(concurrency), 영속성(persistence)과 같은 서비스를 제공하는 Entity Bean[2]을 사용하며 클라이언트에서 Entity Bean을 직접 사용할 경우 과부하가 걸릴 수 있으므로 Session Bean[2]을 생성하여 클라이언트가 이용할 수 있도록 한다.

도서 대출 관리를 위해서는 데이터베이스에 도서의 정보, 도서를 대출하는 이용자의 정보, 그리고 대출 정보 등을 저장하는 테이블을 가지고 있으며 이 영속성을 지닌 데이터를 제어하고 관리하기 위해 애플리케이션 계층에 Entity Bean을 설계한다. Entity Bean은 각각의 데이터를 저장한 테이블마다 하나의 Entity Bean을 생성하며 이 Entity Bean이 각각의 테이블에 접근하여 데이터를 저장(insert), 삭제(delete), 변경(update) 하게 된다. 그리고 상위에 있는 Session Bean은 비즈니스 로직을 포함하며 Entity Bean과 연동하여 데이터를 저장한다. 최상위 프리젠테이션 계층의 JSP는 이용자의 입력을 받아 Session Bean에게 처리를 요청하고 결과를 받아 웹브라우저에 표현하는 역할을 담당한다.

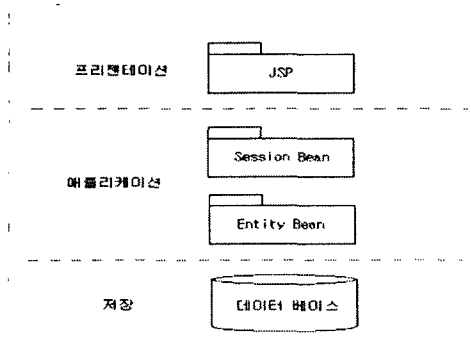

그림 2. 전체 시스템 구조
Fig. 2 System architecture

위 그림 2는 전체 시스템의 아키텍처 계층을 보여준다. 이와 같이 프리젠테이션 계층과 애플리케이션 계층이 완벽히 분리됨으로써 애플리케이션 계층의 Bean 개발자는 시스템의 비즈니스 문제를 해결하는 데에만 집중을 할 수 있다. 프리젠테이션 계층의 클라이언트 개발자는 비즈니스 로직의 개발이나 데이터베이스 액세스를 구현하는 작업이 필요 없게 되며 결과를 보여주는 작업에만 집중할 수 있게 된다[6].

이 구조는 MVC(Model-View-Controller) 디자인 패턴[7]에 기반을 두고 설계된 구조로써 EJB를 사

용하여 엔터프라이즈 애플리케이션을 개발할 때 커다란 장점이 된다.

1. 애플리케이션 계층

애플리케이션 계층에는 데이터베이스에 저장된 데이터를 관리하는 Entity Bean과 이 Entity Bean을 사용하여 이용자의 요구를 처리하는 Session Bean으로 구성된다.

1.1 Entity Bean 설계

애플리케이션 계층에 존재하는 Entity Bean은 대출에 필요한 데이터를 저장한 테이블과 1:1 관계를 가지게 된다. 즉

STUDENT_TABLE ↔ StudentEntityBean
RENT_TABLE ↔ RentEntityBean
BOOK_TABLE ↔ BookEntityBean

와 같이 각각의 테이블과 Entity Bean이 맵핑되며 Entity Bean은 또한 테이블에 정의된 컬럼(column)과 맵핑하기 위한 필드(field)와 이 필드를 엑세스하기 위한 setter/getter 메소드를 가지게 된다.

Session Bean을 통하여 사용되는 Entity Bean은 BMP(Bean-Managed Persistence) 방식을 사용한다. BMP로 작성된 Entity Bean은CMP(Container -Managed Persistence)로 개발하기 어려운 복잡한 SQL문을 처리하기 위해 사용하며 데이터베이스에 종속적인 SQL문을 사용한다.

Entity Bean은 모두 BMP로 설계하며 개발자가 직접 Bean에 선언된 모든 메소드를 작성하여야 한다. Entity Bean은 로컬에 작성되는 Session Bean과 통신을 하기 때문에 로컬 인터페이스를 가지게 된다. 이 로컬 인터페이스는 javax.ejb. EJBLocal Object를 상속하고 테이블에 각 컬럼의 값을 설정하고 가져오기 위한 setter/getter 메소드를 선언한다. 홈 인터페이스는 javax.ejb.EJBLocalHome를 상속받고 Entity Bean을 생성하고 검색, 삭제를 담당하는 메소드를 가진다.

1.1.1 StudentEntityBean

①StudentEntityBean의 로컬 인터페이스

```
public interface StudentEntity
                extends EJBLocalObject {
  public String getSnumber();
  public void setSname(String sname);
  public String getSname();
  public void setSpassword(String spassword);
  public String getSpassword();
  public void setSmajor(String smajor);
  public String getSmajor();
}
```

위에 보이는 StudentEntity 인터페이스는 StudentEntityBean의 로컬 인터페이스를 보여준다. 다른 메소드와는 달리 STUDENT_TABLE의 주키인 SNUMBER를 설정하는 setter 메소드는 선언하지 않는다. 주키는 한번 설정이 된 후 변경이 되면 중복의 우려가 있기 때문이다.

②StudentEntityBean의 홈 인터페이스

StudentEntityBean의 홈 인터페이스인 Student EntityHome인터페이스는 javax.ejb. EJBLocal Home을 상속 받고 Entity Bean을 생성, 검색, 삭제에 관한 메소드를 선언한다. 이 메소드는 EJB의 명세(specification)에 따라 작성된다.

create() 메소드는 계정을 생성할 때 사용하며 이용자의 정보를 받아 STUDENT_TABLE에 하나의 로우(row)을 저장하게 된다.

findByPrimaryKey() 메소드는 계정을 수정할 때 사용되는 메소드이다.

이 메소드는 StudentEntity를 반환하며 이것을 받아 StudentEntity에 정의된 setter/getter 메소드를 사용하여 계정 정보를 변경하게 된다.

findStudent() 메소드는 이름을 입력받아 테이블에 같은 이름들의 계정이 있으면 Collection에 저장하여 반환하는 역할을 한다.

```
public interface StudentEntityHome
                extends EJBLocalHome {
  public StudentEntity create(String snumber,
                String sname, String spassword,
                String smajor)
                throws CreateException;
  public StudentEntity findByPrimaryKey(
                String snumber)
                throws FinderException;
  public Collection findStudent(String sname)
                throws FinderException;
}
```

1.1.2 RentEntityBean

①RentEntityBean의 로컬 인터페이스

```
public interface RentEntity
                extends EJBLocalObject {
  public String getBnumber();
  public String getSnumber();
  public void setRentdate(Date rentdate);
  public Date getRentdate();
  public void setReturndate(Date returndate);
  public Date getReturndate();
  public void setExtension(String extension);
  public String getExtension();
}
```

RentEntity 인터페이스는 RENT_TABLE의 각 컬럼들을 엑세스하기 위한 메소드를 선언한다. RENT_TABLE은 주키를 가지고 있지 않고 BNUMBER와 SNUMBER 두 개의 외래키를 가

지고 있으며 이 외래키의 변경을 예방하기 위해setter 메소드는 선언하지 않는다. 그 외의 컬럼은 모두 설정과 읽기를 위한 sette /getter 메소드를 선언한다.

②RentEntityBean의 홈 인터페이스

RentEntityHome의 findDelayStudents() 메소드는 대출 기한을 초과한 이용자들을 Collection 객체에 저장하여 반환한다.

create()와 findByPrimaryKey() 메소드의 역할은 앞서 기술한 StudentEntityHome의 메소드와 동일한 기능을 수행한다.

```
public interface RentEntityHome
                extends EJBLocalHome {
                        .
                        .
  public Collection findDelayStudents()
                        throws FinderException;
}
```

1.1.3 BookEntityBean

①BookEntityBean의 로컬 인터페이스

```
public interface BookEntity
                extends EJBLocalObject {
  public String getBnumber();
  public void setBname(String bname);
  public String getBname();
  public void setBauth(String bauth);
  public String getBauth();
  public void setBpublisher(String bpublisher);
  public String getBpublisher();
  public void setBlocation(String blocation);
  public String getBlocation();
  public void setBrent(String brent);
  public String getBrent();
}
```

BookEntityBean의 로컬 인터페이스인 BookEntity 인터페이스는 BOOK_TABLE의 각 컬럼들을 엑세스하기 위한 setter/getter메소드를 가지게 되고 역시 주키인 BNUMBER의 변경을 막기 위해 setter 메소드는 선언하지 않는다.

②BookEntityBean의 홈 인터페이스

BookEntityHome의 create(), findByPrimaryKey() 메소드 역시 앞서 설명한 홈 인터페이스의 findByPrimaryKey() 메소드의 역할과 동일하며 findBook() 메소드는 도서명, 저자, 출판사를 각각 따로 입력 받거나 동시에 입력받아 테이블에서 검색하여 동일한 문자열을 가진 도서가 있으면 이 도서들의 정보를 Collection 객체에 저장하여 반환한다.

```
public interface BookEntityHome
                extends EJBLocalHome {
                        .
                        .
  public Collection findBook(String bname,
                String bauth, String bpublisher)
                  throws FinderException;
}
```

이와 같이 선언된 로컬 인터페이스와 홈 인터페이스는 StudentEntityBean, RentEntityBean, BookEntityBean에서 상속을 받아 구현한다. 그림 3은 UML(Unified Modeling Language)[8]을 사용하여 Entity Bean을 표시한 클래스 다이어그램이다.

다이어그램에 표현된 각 Entity Bean은 데이터베이스의 각 테이블에 맵핑하기 위한 필드들을 선언하고 로컬 인터페이스에 선언된 setter/getter 메소드의 실제 구현을 한다. 그리고 홈 인터페이스에 선언된 Entity Bean의 생성, 검색, 삭제에 관련된 메소드는 BMP Entity Bean의명세에 따라 ejbCreate() ejbFindByPrimaryKey() 등과 같이 접두사 ejb를 붙여 JDBC(Java DataBaseConnectivity)[4][9]를 사용하여 직접 데이터베이스에 접속하여 데이터를 저장하고 변경, 삭제하는 로직을 작성한다.

각 Entity Bean 사이에 연관 관계는 없으며 상위에 구현하는 Session Bean에서 관리자나 이용자의 기능 구현에 필요한 Entity Bean을 생성하여 사용하게 된다.

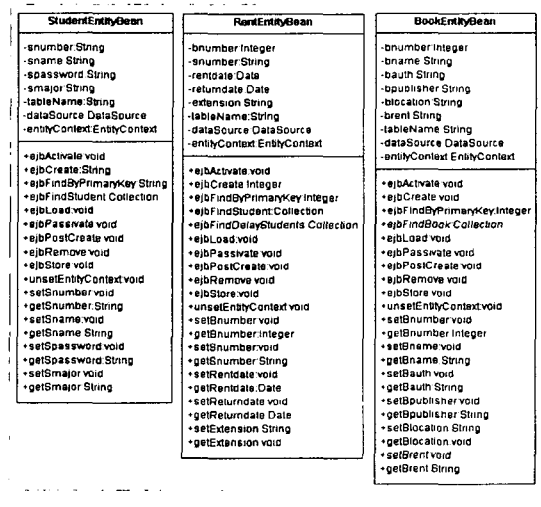

그림 3. Entity Bean의 클래스 다이어그램
Fig. 3 Class diagram of Entity Bean

1.2 Session Bean 설계

Session Bean은 앞서 설계한 Entity Bean을 조합하여 프리젠테이션 계층의 JSP에서 받은 요구를 처리하게 된다. JSP와의 통신을 위해서는 원격 인터페이스를 선언해야 하며 Entity Bean에서 모든 정보를 데이터베이스에 저장하고 관리하게 되므로 상태를 유지할 필요성이 없다. 이로 인해 Stateless Session Bean으로 구현하며 관리자와 이용자의 요구 사항을 처리하는 메소드들이 구현된다.

①BookLendingSessionBean의 원격 인터페이스

Session Bean은 프리젠테이션 계층의 JSP들이 사용하며 JSP와 EJB는 적재되는 컨테이너가 다르기 때문에 원격으로 통신을 하게 된다. 이 때문에 Session Bean은 원격 인터페이스를 사용하여야 하며 이 원격 인터페이스는 아래 와 같이 선언한다.

```
public interface BookLendingSession
                    extends EJBObject {
   public boolean login(String snumber,
            String spassword)
            throws RemoteException;
   public boolean createID(String snumber,
                String sname, String spassword,
            String smajor)
            throws RemoteException;
                    .
                    .
    public DelayInfo delayStudents()
            throws RemoteException;
}
```

원격 인터페이스 BookLendingSession은 javax.ejb.EJBObject를 상속 받고 관리자와 이용자의 요구 기능을 수행하는 메소드를 선언하며 이 메소드는 Session Bean에서 상속을 받아 구현한다.

②BookLendingSessionBean의 홈 인터페이스

BookLendingSessionHome 홈 인터페이스는Session Bean을 생성하는 create() 메소드 하나만 가지게 되고 나머지 Bean의 생명 주기에 관련된 메소드들은 EJB 컨테이너에서 처리하게 된다.

```
public interface BookLendingSessionHome
                    extends EJBHome {
    public BookLendingSession create()
            throws CreateException, RemoteException;
}
```

아래 그림 4에 클래스 다이어그램은 앞서 설계한 원격 인터페이스를 상속하고 Stateless Session Bean 명세에 맞추어 작성된다.

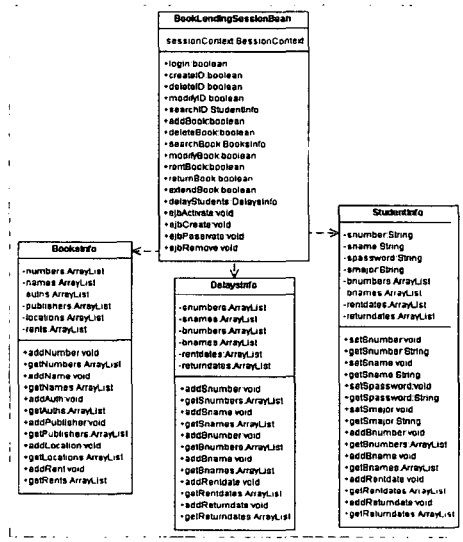

그림 4. Session Bean의 클래스 다이어그램
Fig. 4 Class diagram of Session Bean

BookLendingSession 원격 인터페이스에서 상속받은 메소드들은 각각의 기능에 필요한 Entity Bean을 생성하여 기능 수행에 필요한 정보를 데이터베이스에서 가져와 처리를 한 후 결과를 JSP에 반환한다. 위 다이어그램에 표기된 BooksInfo 클래스는 일반적인 Bean 클래스로써 책 검색을 하였을 경우 검색되는 책들의 정보를 저장하여 JSP에 반환할 때 사용된다. 같은 이유로 연체된 이용자들의 정보를 저장하여 반환할 때 사용하는 DelayInfo 클래스와 학생을 검색하였을 때 검색된 정보를 저장 및 반환에 사용하는 StudentInfo 클래스가 있다. 이들 3개의 일반 Bean 클래스는 원격 인터페이스를 통해 전송이 되므로 모두 java.io.Serializable을 구현(implements)한다.

2. 프리젠테이션 계층

JSP는 도서 대출 업무에 필요한 기능하나만을 수행 할 수 있도록 한다. 즉 BookLendingSession Bean에 존재하는 login() 메소드는 login.jsp에서만 호출이 되며 createID() 메소드는 create_id.jsp에서 실행되게 된다. 그 외 대출 관리 업무에 필요한 모든 메소드는 각각 하나의 JSP에서만 실행이 된다. 이렇게 함으로써 JSP의 보수가 필요할 때 모든 JSP를 수정할 필요가 없어지게 되며 해당 JSP만 수정하면 된다. 각 기능별 JSP는 그림 5와 같이 구성되며 JSP들 간의 연관성을 갖지 않는다.

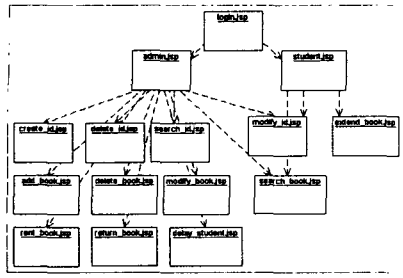

그림 5. 각 기능별 JSP
Fig. 5 JSP of each function

관리자나 이용자는 시스템에 접근하기 위해서 가장 먼저 login.jsp에서 로그인을 하게되며login.jsp는 ID를 식별하여 관리자의 admin.jsp나 이용자의 student.jsp로 이동을 하게 된다. admin.jsp는 계정 생성, 계정 삭제 및 도서 대출 등 대출 업무에 필요한 메뉴를 가지고 있으며 메뉴의 선택에 따라 계정 생성은 create_id.jsp, 도서 대출은 rent_book.jsp와 같이 각 기능에 필요한 JSP로 이동을 한 후 해당 JSP에서 Session Bean을 생성하여 요청을 처리하고 Session Bean에서 반환된 결과를 웹 브라우져에 표시한다. student.jsp는 이용자를 위한 메뉴를 가지고 있으며 로그인시 이용자의 개인 정보와 대출한 책들의 정보를 보여주게 된다. 계정 수정을 담당하는 modify_id.jsp와 책 검색을 하는 search_book.jsp는 이용자와 관리자가 같이 공유하여 사용한다. 관리자의 계정 수정과 책 정보 수정은 먼저 계정 검색과 책 검색을 수행한 후 결과를 받아 수정을 해야 하는 계정 정보 또는 책 정보를 수정을 하게 된다. 관리자와 이용자의 모든 기능을 수행하는 과정을 순서도나 순차 다이어그램(Sequence diagram)으로 한 번에 표기하는 것은 매우 어렵고 복잡하며 가독성이 떨어지므로 각 기능별로 분리하여 그림 6과 같이 하나의 기능은 하나의 협동 다이어그램(Collaboration diagram)으로 표현한다.

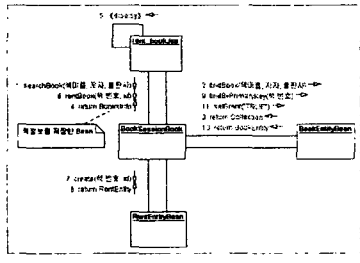

그림 6. 도서 대출 과정
Fig. 6 Sequence of book lending out

위 그림 6은 관리자의 도서 대출 과정을 협동 다이어그램(collaboration diagram)으로 표기한 것이다. 도서 대출 과정은 rent_book.jsp에서 먼저 대출을 할 책을 찾기 위해 BookLendingSessionBean을 생성하여 searchBook() 메소드를 호출한다.

searchBook()는 인자로 받은 검색할 책의 정보를 받아 BookEntityBean의 findBook() 메소드를 호출하여 데이터 베이스에 BOOK_TABLE를 검색하고 결과를 BooksInfo Bean에 넣어서 rent_book.jsp로 반환한다. rent_book.jsp는 받은 검색 결과를 웹 브라우져에 표시하고 대출자의 계정을 입력받아 BookLendingSessionBean의 rentBook() 메소드를 호출한다. rentBook()는 대출 정보를 저장하기 위해 RentEntityBean에 create() 메소드를 호출하여 RENT_TABLE에 대출 정보를 기록하게 된다. 마지막으로 BOOK_TABLE에 대출된 책에 대출 중이라는 표식을 표시하기 위해 BookEntityBean에setRent()메소드를 호출하여 "TRUE"값을 BOOK_TABLE에 기록하게 되고 도서 대출을 종료한다.

3. 시스템 구현

EJB와 JSP를 사용하여 설계된 도서 대출 관리 시스템은 로컬 PC에 클라이언트 프로그램을 설치하는 번거로움 없이 웹 브라우져를 통하여 전반적인 도서 대출 업무를 수행할 수 있도록 한 엔터프라이즈 애플리케이션이다. 이 시스템은 Windows 98/2000/XP를 운영체제로 사용하는 PC에서 구현하였으며 서버는 J2EE 플랫폼 중에 하나인 BEA사의 WebLogic 8.1을 설치하여 EJB와 JSP를 배포하였다. 그리고 대출 업무에 필요한 이용자 정보, 도서 정보, 대출 정보를 저장하기 위한 데이터베이스로 오라클을 사용하여 각 테이블을 정의하였다. 클라이언트의 테스트는 Explorer 6.0을 사용하였다.

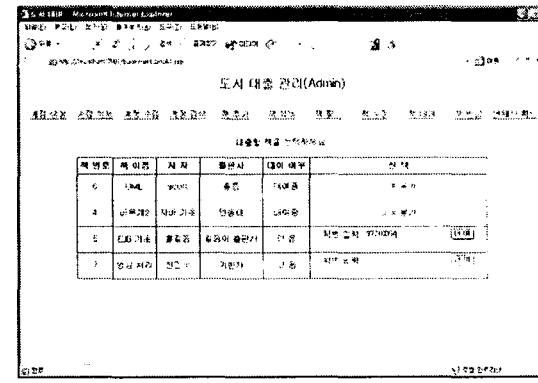

그림 7. 도서 대출 JSP 화면
Fig. 7 Picture of rent_book.jsp

그림 7은 관리자로 로그인 후 도서 대출 과정의 화면이다. 도서의 대출은 먼저 대출할 책을 검색하여 웹 브라우져에 그림 7과 같이 목록을 보여주게 되며 대출할 도서에 계정을 입력하여 대출을 하게 된다.

이와 같이 EJB와 JSP를 사용하여 개발된 도서 대출 관리 시스템은 Java가 실행되는 어떠한 플랫폼이라도 EJB와 JSP를 수정 없이 배포할 수 있게 구현되었다. 클라이언트는 프로그램을 설치할 필요 없어지며 인터넷이 가능한 모든 플랫폼에서 도서 대출 업무를 볼 수 있다.

IV. 결론 및 향후 과제

엔터프라이즈 애플리케이션 개발에 EJB를 사용하여 애플리케이션 계층과 프리젠테이션 계층을 분리함으로써 각각의 각 계층에 적합한 로직의 개발에만 집중할 수 있었으며 서로의 연관 관계가 최소화 되어 유지/보수의 편리함을 제공하게 되었다. 또한 분산 환경에서 자원의 관리나 보안등에 관여하지 않고 비즈니스 로직의 개발과 관련 서비스 구현에만 집중할 수 있었으며 웹 기반 인터페이스를 제공하여 장소와 H/W 및 OS에 구애받지 않고 도서 대출에 관한 전반적인 업무를 수행할 수 있었다. 제안된 도서 대출 관리 시스템은 BMP Entity Bean을 사용하여 데이터베이스의 종류에 따라 SQL을 재작성해야 한다. 그리고 웹을 기본 인터페이스로 채택하여 바코드 인식을 위한 장치를 사용할 수 없게 되었다. 이를 극복하기 위해서는 CMP Entity Bean을 사용하여 데이터베이스에 종속되는 문제점을 해결하고 웹에서 인증된 Applet을 배포하여 로컬 시스템의 디바이스를 인식하게 하는 연구가 필요하다.

참고문헌

[1] Sun Microsystems Inc., Java 2 Platform Enterprise Edition(J2EE), http://java.sun.com/j2ee/

[2] Sun Microsystems Inc., Enterprise Java BeansTM Specitication Version 2.0 Final Release, 2001.

[3] 안건태, 정명희 외, "iPlace:A Web - based Collaborative Work System Using Enterprise JavaBeans Technology," 정보처리학회논문지 D, Vol.8, No.6, 2001.12

[4] H.M.Deitel, P.J.Deitel, S.E.Santry, Advanced JavaTM2 Platform, Prentice Hall, 2001

[5] Sun Microsystems Inc.,JavaServer Pages Technology, http://java.sun.com/products/jsp/

[6] 김수동, "Enterprise JavaBeans(EJB) 기반의 컴포넌트 프로그래밍," 한국정보처리학회지Vol.7 No.4, 2000.

[7] Floyd Marinescu, EJB Design Pattern, Wiley, 2002.

[8] James Rumbaugh, Ivar Jacobson, Grady Booch, The Unified Modeling Language Reference Manual, Addison Wesley, 1999.

[9] Jon Ellis, JDBC API Tutorial and Reference, Addison Wesley 3rd Ed., 2003.

저자소개

서봉근(Bong-kun Seo)

2004.2안동대학교컴퓨터공학과 공학사
2004.3 -현재안동대학교 대학원 컴퓨터 석사 과정
※관심분야:Media Framework, 분산 컴퓨팅, 객체지향 분석/설계/프로그래밍

김윤호(Yun-Ho Kim)

1983.2.경북대학교전자공학과 공학사
1993.2.경북대학교대학원 컴퓨터공학과 공학석사
1997.2.경북대학교대학원 컴퓨터공학과 공학박사
1997. 8. - 현재안동대학교 전자정보산업 학부 부교수
※관심분야: 인터넷컴퓨팅 , 객체지향 분석/설계/프로그래밍, 분산객체시스템, 병렬처리