EJB기반 XML 상품 카탈로그의 구현 및 분석

김 경 래† · 하 상 호††

요 약

인터넷 중심의 정보화 사회가 되면서 B2C나 B2B간에 상품 정보의 교환이 활발해지고 있다. 우리는 상품정보를 통합적으로 표현할 수 있는 XML기반 통합 상품 표현 모델을 제시한 바 있다. 이 모델은 상품을 통합적으로 표현할 수 있고, 데이터의 중복성을 회피할 수 있는 특징을 가지고 있다. 본 논문에서는 Java의 컴포넌트 기술인 Java Bean과 EJB(Enterprise Java Beans)를 사용하여 제안된 상품 표현 모델을 구현하였고, 실제 웹 상의 상품 정보를 대상으로 하여 데이터 중복성 제거에 따른 효과를 분석한다.

Implementation and Analysis of a XML Product Catalog based on the EJB

Kyoungrea Kim† · Sangho Ha††

ABSTRACT

In the Internet based information technology community, the exchange of product information is more activated in e-commerce of B2C and B2B. We have suggested a XML based unified product description model with which all kinds of product informations can be represented. This model also has a feature to avoid duplication of the product information. In this paper, we will implement this model using Java Beans and EJB (Enterprise Java Beans) of the Java component technology, and then test this system over several products on the web. Finally, we will analyze the effect due to the removal of the information duplication.

키워드 : 전자상거래(e-commerce), XML, EJB, 상품카탈로그(product catalog), 자바(JavaBeans)

1. 서 론

인터넷의 확산과 더불어 B2C의 수요가 크게 증가하면서, 각각의 기업들에서는 B2B의 중요성을 깨닫게 되었고, 이를 위한 표준 거래 환경을 제정하기 위한 노력을 하고 있다[1]. 이와 같은 노력에 의해 제안된 프레임워크들은 기업간의 상품에 대한 정보교환을 위해 XML[2]에 기반하여 각각의 고유한 상품카탈로그를 제공하고 있다. 상품카탈로그는 각 기업에서 출시하는 상품에 대한 정보를 표현하기 위한 구조를 가지고 있고, 표현되는 각각의 모든 상품들은 공통적인 정보와 고유의 정보를 가지고 있다.

논문의 지난 연구에서 상품 정보를 통합적으로 표현할 수 있는 XML기반 통합 상품 표현 모델[3]을 제시한 바 있다. 이 상품 표현 모델은 여러 상품을 통합적으로 표현할 수 있고, 또한 공통 정보의 중복 표현을 문서 구조만으로 회피할 수 있다. 본 논문에서는 XML기반 통합 상품 표현 모델을 Java의 컴포넌트 기술인 Java Beans와 EJB(Enterprise Java Beans)[4]에 기반하여 구현하고, 실제 웹 상의 상품 정보들을 대상으로 이를 테스트한다. 또한 테스트 결과를 이용하여 상품 표현 모델의 중복성 제거의 효율을 분석한다.

본 논문의 구현에 기반이 되는 컴포넌트[5]란, 특정한 기능을 수행하기 위해 독립적으로 개발, 보급되고 잘 정의된 인터페이스를 가지며 다른 부품과 조립되어 응용시스템을 구축하기 위해 사용되는 소프트웨어의 단위를 의미한다. 컴포넌트의 가장 큰 장점은 재사용성과 다른 플랫폼과의 호환성, 이식성에 있다. 현재 컴포넌트 구조의 표준은 Sun의 EJB (Enterprise Java Bean)와 MS의 COM(Component Object Model)[6]이 양대산맥을 이루고 있다. COM은 모든 프로그램 언어를 이용하여 개발할 수 있지만, EJB는 Java만을 이용하여 개발해야 한다. 그러나 EJB는 Java의 이식성 및 개방성 등의 우수한 특성을 그대로 유지하며 COM보다는 많은 기존의 컴포넌트들을 제공한다는 장점이 있다. 또한, 현재 XML기반의 응용 프로그램의 제작은 Java 기술을 사용하여 많이 개발되고 있기 때문에, 우리는 컴포넌트 기술로 Java의 EJB를 선택하였다.

본 논문에서 제시한 저장 컴포넌트의 구현을 위하여 Java에서 제공하는 Java Beans와 EJB(Enterprise Java Be-

※ 본 연구는 한국소프트웨어진흥원의 ITRC 사업에 의해 수행된 것임.
† 준 회 원 : 순천향대학교 대학원 정보기술공학부
†† 종신회원 : 순천향대학교 정보기술공학부 교수
논문접수 : 2001년 12월 31일, 심사완료 : 2002년 3월 20일

ans)[4]를 사용한다. EJB는 Java2 Enterprise Edition의 일부분으로 분산환경하에서 응용 프로그램을 개발, 배포, 실행하는 것들에 대한 구조를 의미한다. 분산환경하의 응용 프로그램들은 트랜잭션관리, 보안, 데이터베이스 제어 등과 같은 시스템 수준의 서비스를 필요로 하며, J2EE(Java 2 Platform, Enterprise Edition) 플랫폼은 이러한 서비스 등을 제공하여 프로그래머가 비즈니스로직에만 전념할 수 있도록 다양한 기능을 제공하고 있다. 또한 J2EE의 권고 사양에 따라 제작된 Enterprise Bean코드는 다양한 J2EE 서버 상에서 재가공 없이 재 사용될 수 있다는 장점을 가지고 있다. Java Beans는 Java2 Standard Edition하의 컴포넌트 모델로써, 주로 비주얼한 소프트웨어 컴포넌트이다. Java Beans가 클라이언트환경에서 실행되는 단일 프로세스 위주의 컴포넌트 모델인 반면, EJB는 분산객체기술에 기반하여 설계되어서 Multi-tier환경에 적합한 서버 측의 모델로서 정의되어 있다. 본 논문에서는 클라이언트로부터 요구를 받아 그 요구사항을 분석하기 위해 Java Beans를 사용하였고, 분석된 요구사항을 처리하기 위해 EJB를 사용한다. 제작된 Java Bean은 특별한 재가공 없이 분산환경 하에서 재 사용할 수 있고, EJB는 애플리케이션 서버에 배포되어 규모나 환경에 구애받지 않고 사용할 수 있다.

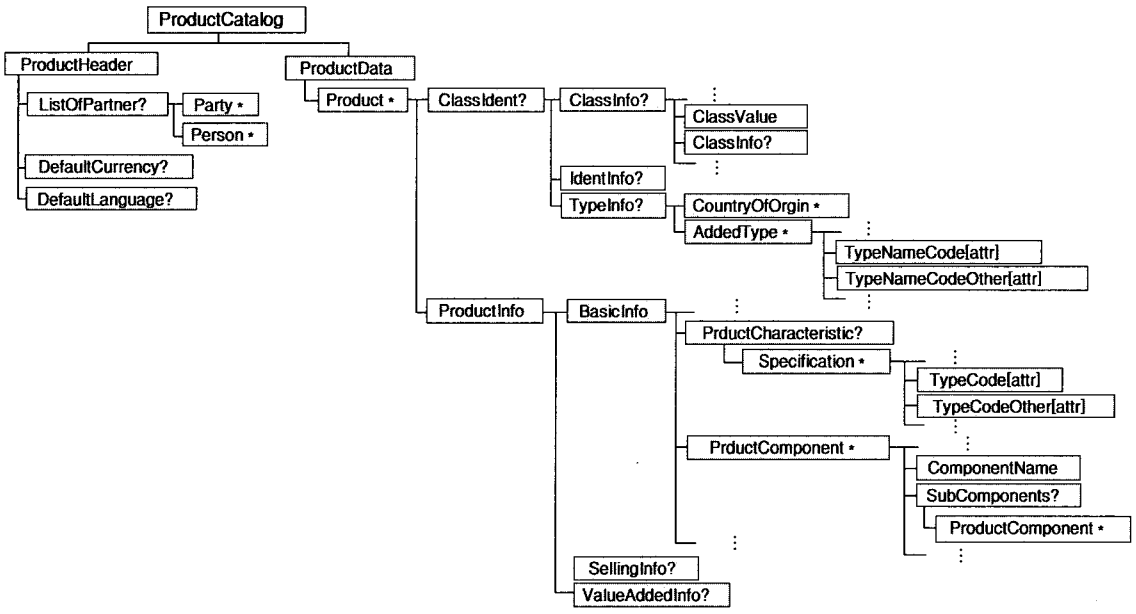
논문의 순서는 다음과 같다. 2장에서 XML기반 통합 상품 표현 모델에 대해 서술하고, 3장에서는 실제적으로 통합 상품 표현 모델을 저장하는 웹 기반의 저장모듈을 EJB로 구현 한다. 4장에서는 실제 웹 상의 쇼핑몰에서 자료를 추출한 후, 이를 3장에서 구현한 시스템에 적용하여 통합 상품 표현 모델의 중복성 제거의 효율을 테스트하고 분석해 본다. 마지막으로 5장에서 결론을 언급한다.

2. XML기반 통합 상품 표현 모델

본 논문에서 사용된 상품의 정보를 표현하기 위한 상품 카탈로그[3]는 상품 서술시 정보의 중복 표현과 데이터베이스에 저장 및 검색시 데이터 중복을 제거해주는 효과가 있다. (그림 1)은 통합 상품 표현 모델의 전체적인 구조를 보여준다.

이 모델은 상품의 정보를 ProductHeader와 ProductData의 두 부분으로 나누어서 표현한다. ProductHeader에서는 상품 카탈로그 전체에 쓰이는 기본 언어와 통화를 설정하고, 상품에 관련된 Partner들에 대한 정보가 리스트로 표현된다. Partner의 정보로는 상품에 관련된 업체(Party)와 사람(Person)으로 나뉘어 표현할 수 있다. Party와 Person은 각각은 특성에 맡는 정보를 표현할 수 있는 구조로 이루어져 있다. Party에는 제조사나 판매사, 중계상등의 정보가 표현되고, Person에는 상품의 제작자나 저자, 번역자 등의 인물정보가 표현된다.

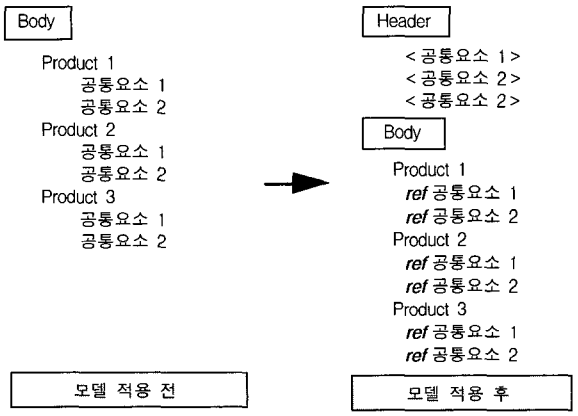
상품의 고유한 정보들은 ProductData에 표현된다. ProductData는 하부 엘리먼트로 0개 이상의 상품들을 표현 할 수 있다. 따라서 하나의 상품 카탈로그에 하나의 Header를 가지고 여러 개의 상품을 표현 할 수가 있게된다. 상품의 고유정보는 분류정보와 식별정보, 유형정보, 기본정보, 판매정보, 부가가치 정보로 나뉘는데, 분류정보(ClassInfo)와 식별정보(IdentInfo), 유형정보(TypeInfo)는 ClassIdent에 표현되고, 기본정보(BasicInfo), 판매정보(SellingInfo), 부가가치정보(ValueAddedInfo)는 ProductInfo에 표현된다. 분류정보는 여러 가지 상품을 품목별로 분류하는 정보를 나타내고, 식별정보는 여러 가지 상품 중에 해당 상품을 식별하기 위



(그림 1) XML기반 통합 표준 상품 표현 모델

한 정보를 나타낸다. 이와 같은 분류정보 및 식별정보는 코드로써 나타낼 수 있다. 또한, 유형정보는 하나의 대표제품(예 : 도서)내에서 각 개별제품들을 구분하고 유형화하는 기준을 제공하는 정보로, 주로 소비자의 탐색편의를 도모하기 위해서 제공된다. ProductInfo의 기본정보, 판매정보, 부가가치정보는 각각 상품의 기본적인 정보와, 판매에 관련된 가격이나 가격의 할인율 등의 정보, 상품의 부가적인 정보로써 상품 평가 및 관련 상품 등을 표현한다.

이 모델은 다음과 같은 특징들을 제공한다. ProductCharacteristic의 TypeCode에 따라 모든 상품의 세부적인 정보들을 통합적으로 표현할 수 있고, 상품 정보를 분류정보, 식별정보, 유형정보, 기본정보, 판매정보, 부가가치정보로 분류하여 표현할 수 있다. 또한 분류정보인 ClassInfo의 하부 계층에서 ClassInfo를 재귀적으로 사용함으로써 분류정보를 계층적으로 표현할 수 있다. 이러한 특징은 ProductComponent에서도 나타나는데, 이 특징에 의해 상품 정보를 계층적으로 표현할 수 있게 한다. 사용자의 편의를 위한 부가유형의 지원 역시 큰 특징이라 할 수 있다. 기본적인 부가유형으로 내용이나 형태, 용도 등의 상품 성질로 구분되는 유형정보와 신제품, 추천제품, 인기제품, 수상제품, 사용자구분, 사용자시기 등의 상품의 판촉에 관련된 유형정보가 지원된다. 유형정보는 TypeNameCode에 의해 구분되고 부가유형의 추가를 원할 경우 TypeNameCodeOther를 사용하여 추가할 수 있다. 이는 Specification에서도 나타나는 특징으로 모델에 확장성을 부여한다. 또한, 이 모델은 각 상품에 중복적으로 나타날 수 있는 공통 요소를 상품 모델의 Header 부분에 표현함으로써 효과적으로 중복성을 제거하였다.



(그림 2) 통합상품표현모델 적용전과 적용후

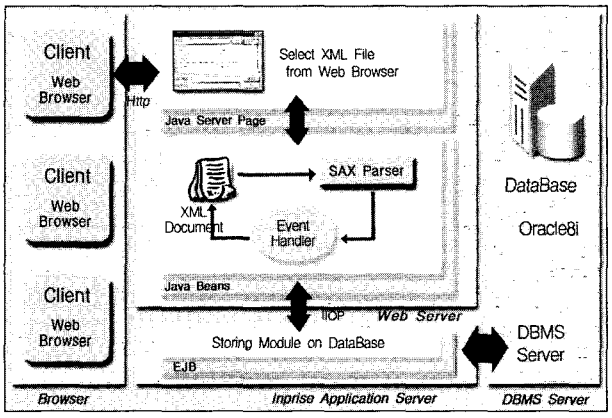
(그림 2)는 통합 상품 표현 모델에서 중복성 제거를 위해 제시한 방식이다. 예를 들어 하나의 공통요소가 1Kb의 용량의 정보를 가지고 있다고 한다면, 모델적용 전에는 6Kb의 정보를 표현해야 하지만 적용후에는 3Kb의 정보만으로 표현이 가능하다. 따라서 표현되는 상품의 양이 늘어나거나 공통요소가 많은 상품일수록 중복성 제거의 효율은 높아지게 된다.

3. XML기반 통합 상품 표현 모델의 구현

본 논문에서 구현한 시스템은 Window98 환경에서 JBuilder4 [5]를 사용하여 제작하였다. Sun OS 5.7의 운영체제에서 Oracle8i 데이터베이스를 사용하였고, WebServer를 위해 Tomcat3.2.1 Server와 EJB를 위한 Application Server로 Boland사에서 제공하는 IAS4.0 Server[6]를 사용하였다.

시스템은 크게 Client와 Middleware, 데이터베이스의 세 부분으로 구성되어 있다. Client의 인터페이스로는 XML문서를 웹상에서 입력받아 저장하기 위하여 웹브라우저를 이용하였다. Middleware는 JSP(Java Server Page)[7]와 JavaBeans, EJB, 그리고 JSP container를 포함하는 Tomcat Server와 EJB를 위한 Application Server로 이루어져 있다. 데이터베이스로는 Oracle8i[8, 9]를 사용하였다.

(그림 3)은 시스템의 전체 구성을 나타낸다. 사용자와 Middleware간의 출력을 담당해주는 JSP가 있고, 또한 웹브라우저와 EJB간의 인터페이스 역할을 수행하는 Java Beans가 있다. 그리고, 데이터베이스에 정보를 저장을 하는 저장 EJB가 있다. 시스템의 흐름은 다음과 같다. 사용자는 Web Browser를 통하여 검색 및 저장을 수행할 수 있으며, 사용자는 Web Browser를 통하여 Http(Hypertext transfer protocol)로 JSP에게 정보를 전달한다. JSP는 정보를 가지고 Java Beans를 호출하게 되며, 호출된 Java Beans는 IIOP(Internet Inter-ORB Protocol)로 EJB의 Remote 인터페이스와 교류하며 정보를 처리하고, 그 결과를 다시 JSP로 넘겨주게 된다. JSP는 결과를 받아 웹브라우저에 결과를 표현하게 된다. 저장 EJB는 EJB에서 제공하는 Session Beans로 만들었다. Session Beans의 내부 메소드를 Java Beans에서 사용하기 위하여, Remote 인터페이스를 통하여 사용한다.



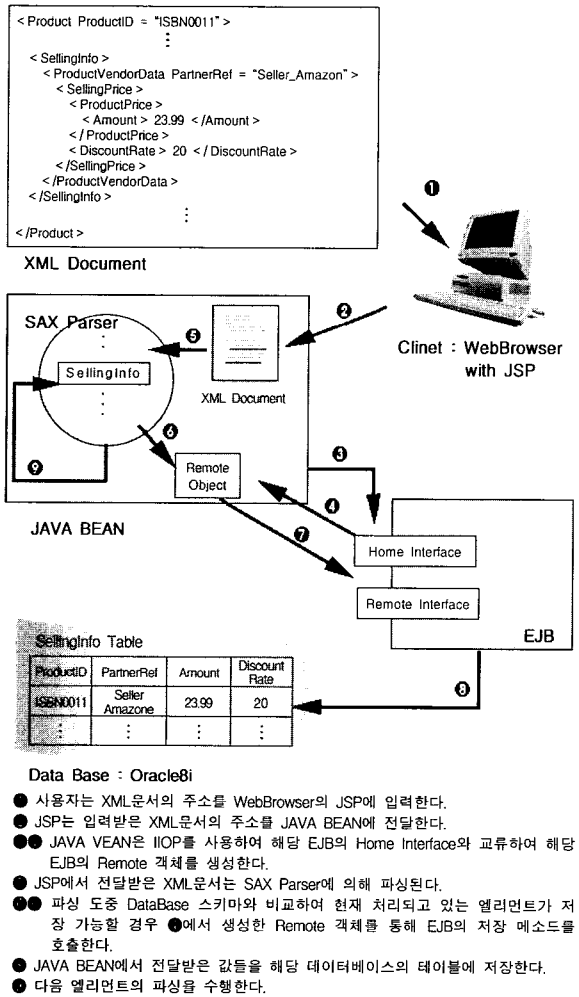
(그림 3) 시스템 구성도

(그림 4)는 시스템의 전체적인 흐름을 나타낸다. 사용자는 웹브라우저를 통하여 XML문서의 URL을 JSP모듈에 전달하고, JSP는 Java Bean을 호출한다. 이때 XML문서의 URL이 인수로 넘겨지고, 호출된 Java Bean은 먼저 EJB로의 연결

을 위한 리모트 객체의 생성을 위해 JNDI(Java Naming and Directory 인터페이스)[11] 서비스를 이용하여 Application Server상에 배포되어 있는 해당 EJB를 찾은 후, EJB의 Home 인터페이스를 통하여 리모트 객체를 생성한다. 객체를 생성한 후, XML문서는 SAX 파서[10]에 전달되어 파싱이 시작된다. XML 파서는 크게 DOM 파서와 SAX 파서로 분류된다. DOM 파서는 XML 문서의 요소들을 파싱 후 트리 형태로 반환한다. SAX 파서는 이벤트 파싱 방법을 사용하여 각 파싱 과정 중 해당 요소에 대해 사용자가 처리할 수 있도록 하여준다. 본 연구는 EJB를 이용하여 데이터베이스에 접근하기 때문에 최대한 처리 시간을 줄일 필요가 있다. 따라서 데이터베이스에 요청하는 쿼리의 용량을 줄이기 위해 이벤트 파싱 방법을 사용하는 SAX 파서를 사용하여 각각의 요소를 파싱한 후 바로 쿼리를 요청하는 방법을 선택하였다. SAX 파서의 특성상, 파싱 과정 중에 데이터베이스의 스키마와 비교가 이루어지며, 조건을 만족하는 엘리먼트가 처리되었을 경우, 리모트 객체를 통한 EJB의 저장 메소드 호출로 해당 엘리먼트에 대한 저장 작업이 이루어지게 된다. (그림 4)의 예에서 XML문서의 <SellingInfo> 엘리먼트가 Parser에 의해 처리되었을 경우, 이 엘리먼트는 데이터베이스의 [SellingInfo Table]로 사상된다. 따라서 [SellingInfo Table]로 저장될 정보들을 추출하여, 이를 인자로 EJB의 저장 메소드를 호출하게 된다.

(그림 4)의 예에서 인자는 ProductID(ISBN0011), PartnerRef(Seller_Amazone), Amount(23.99), DiscountRate(20)이다. <SellingInfo> 엘리먼트에 대한 처리가 끝난 후, Parser는 다시 다음 엘리먼트의 처리를 시작한다. 위의 처리 과정은 XML 문서의 마지막 엘리먼트를 처리할 때까지 반복된다.

(그림 4)의 시스템에서 주요 구성모듈인 Java Bean과 EJB에 대해 구체적으로 살펴보도록 하겠다.



● 사용자는 XML문서의 주소를 WebBrowser의 JSP에 입력한다.
● JSP는 입력받은 XML문서의 주소를 JAVA BEAN에 전달한다.
●● JAVA VEAN은 IIOP를 사용하여 해당 EJB의 Home Interface와 교류하여 해당 EJB의 Remote 객체를 생성한다.
● JSP에서 전달받은 XML문서는 SAX Parser에 의해 파싱된다.
●● 파싱 도중 DataBase 스키마와 비교하여 현재 처리되고 있는 엘리먼트가 저장 가능할 경우 ●에서 생성한 Remote 객체를 통해 EJB의 저장 메소드를 호출한다.
● JAVA BEAN에서 전달받은 값들을 해당 데이터베이스의 테이블에 저장한다.
● 다음 엘리먼트의 파싱을 수행한다.

(그림 4) 시스템의 전체적인 흐름

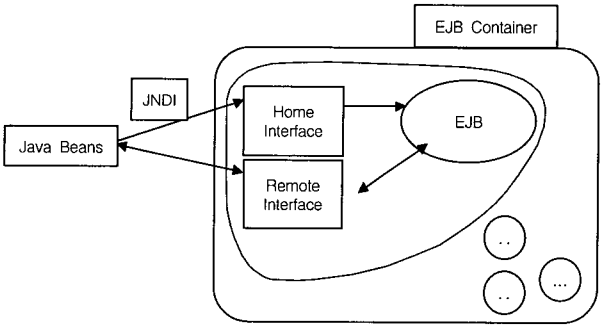
Java Beans

Java Bean은 배포되어있는 저장 EJB의 Home 인터페이스를 통하여 EJB를 생성하고, 해당 EJB에 대한 Remote 객체를 생성한다. 사용자가 웹브라우저에서 JSP를 통해 호출하는 이 모듈은 전달된 XML문서의 URL을 분석하여 문서를 읽어들인다. Java Bean은 JNDI를 통하여 EJB Server의 해당 EJB를 찾고, 해당 EJB의 Home 인터페이스를 통하여 Remote 객체를 생성한다. (그림 5)는 Remote 객체를 생성하는 코드의 예이다. SAX 파서[10]는 문서를 파싱한다. 파싱을 위해 IBM에서 제공하는 SAX Paser를 사용하였다. SAX Paser의 특성상 문서가 파싱 과정 중에 XML문서 내의 각각의 엘리먼트는 처리 가능한 상태가 된다. 이때 각각의 엘리먼트에 대해 데이터베이스 스키마와 비교가 이루어지고, 각 단계에서 처리된 엘리먼트가 데이터베이스의 스키마와 일치하여 저장 조건과 만족된다면, 생성된 Remote객체를 이용하여 EJB의 Remote 인터페이스를 통해 저장 메소드를 호출한다.

```
javax.naming.Context ctx = new javax.naming.InitialContext( ) ;
StoringHome home = (StoringHome)javax.rmi.PortableRemote
                   Object.narrow(ctx.lookup ("Storing"),
                   StoringHome.class) ;
Storing rmt = home.create( ) ;
```

(그림 5) Remote 객체 생성의 예

EJB



(그림 6) EJB 작동 원리

저장 EJB에서는 저장할 수 있는 메소드를 Overriding형태로 제공한다. Java Bean은 JNDI(Java Naming and Directory Interface)에 의해 IAS(Inprise Application Server)에 배포되어 있는 해당 저장 EJB의 Home 인터페이스와 교류하여 EJB를 생성하고, 해당 EJB에 대한 Remote객체를 생성한다. EJB는 생성과 동시에 데이터베이스로의 연결을 시도한다. Java Bean은 해당 EJB의 Remote 인터페이스를 통하여 EJB에서 제공하는 Overriding된 메소드들을 사용할 수 있다. (그림 7)은 EJB에서 제공하는 메소드들의 예이다. Java Bean에서 호출된 이 메소드들은 각각 parameter로 넘겨온 값들에 의해 해당 테이블의 해당 컬럼에 값들을 저장한다.

```
public void Storing(java.lang.String col1,String col2, String
  tabName) throws IOException, RemoteException ;
public void Storing(java.lang.String col1,String col2,String col3,
  String tabName) throws IOException, RemoteException ;
public void Storing(java.lang.String col1,String col2,String col3, String
  col4, String tabName) throws IOException, RemoteException ;
```

(그림 7) EJB에서 제공하는 메소드의 예

4. 테스트 및 분석

본 장에서는 구현된 저장 시스템(시스템A)에 실제적인 상품의 데이터를 저장하여 테스트한다. 테스트에 사용될 상품들의 정보는 amazon.com에서 Computer 카테고리의 Java키워드에 의해 검색된 도서상품 중 40개의 상품을 순차적으로 선택하였다. 선택된 상품 정보를 시스템에 적용해 본 결과 저장 시스템은 무리 없이 상품 정보를 저장하였다.

선택된 상품 정보를 본 시스템에 적용해 본 결과 각각의 상품에 대한 정보들이 데이터베이스상에 저장된 것을 확인하였다.

본 논문에서 적용한 상품표현모델의 중복성 제거의 효율을 분석하기 위하여, 상품 정보의 공통요소를 Header 부분에 표현하지 않는 모델을 별도로 작성하여, 저장 시스템(시스템B)을 구현한 후, 위의 시스템A의 테스트에 사용된 상품들과 동일한 상품들의 정보를 저장하였다. 그리고 시스템A의 저장 결과와 시스템 B의 저장 결과를 분석하여 중복성 제거의 비율을 조사하였다. 시스템B의 구현에 사용된 상품 표현 모델은 시스템A에 사용된 상품 표 현 모델과 동일한 크기의 동일한 자료를 표현함을 원칙으로 하였다. 시스템A와 시스템B의 구성 및 흐름은 3장에서 구현한 (그림 4)의 시스템과 동일하다.

공통요소를 Header에 포함한 모델의 XML DTD를 A.dtd라 하고, 공통요소를 Header에 포함하지 않은 모델의 DTD를 B.dtd라 한다. 위에서 선택된 40개의 상품을 각각 A.dtd와 B.dtd를 적용하여 XML문서로 제작하고, 시스템A와 시스템B에 의해 저장한 후, 저장에 걸리는 시간과 데이터베이스의 용량, XML 파일의 용량을 비교한 결과를 <표 1>이 보여준다.

<표 1> A case와 B case에 의한 효율비교

	테이블 수	파일용량	DB용량	DB저장시간
A	23개	331Kb	1064Kb	31.3sec
B	23개	468Kb	1702Kb	35.4sec
A/B	0%	30%	37%	12%

- A : A.dtd에 의한 XML문서
- B : B.dtd에 의한 XML문서

결과에 의해, A의 상품카탈로그가 B의 카탈로그 보다 약 30%의 파일용량을 절약한 것으로 나타났다. 한 출판사는 여러 권의 책을 출판 할 수 있고, 또한 한 저자는 여러 권의 책을 저술 할 수 있다. 따라서 출판사와 저자에 관련된 정보가 여러 권의 책에 중복하여 표현될 수 있다. 본 논문에서 사용한 상품 표현 모델에 의해 이러한 중복표현이 제거되어 파일용량의 절약을 가져왔다. 파일용량의 절약은 카탈로그 제작시 편의성 및 웹 기반의 전자상거래에서 파일 전송시간의 단축을 가져올 것이다. 파일용량의 절약으로 데이터베이스의 저장공간을 약 37% 절약 할 수 있었다. 이는 상품 카탈로그의 제거할 수 있는 중복 저장의 정도를 보여준다. 또한 저장되는 데이터의 감소로 인하여, 저장에 걸리는 시간도 약 12% 단축할 수 있었다.

비교에 사용된 총 40개의 도서 상품을 17곳의 출판사에서 제공하였다. 이는 한 출판사가 약 57%정도 중복하여 표현됨으로써, 이 중복 정보들을 제거함으로써 위의 결과를 얻을 수 있었다. 또한 별도로 CD상품에 대해 조사해 본 결과 28곳의 기획사가 총 50개의 상품을 기획하여 약 44%의 중복성을 가지고 있었다(CD상품은 interpark.com에서 최신 가요에 의해 검색된 상품을 순차적으로 50개를 선택). 따라서 표현되는 상품의 양이 많아질수록 파일용량의 절약율은 더 커질 것이라 예상된다.

5. 결 론

본 논문에서는 XML 기반 통합상품표현모델을 기반으로, Java의 컴포넌트 기술인 Java Beans와 EJB를 사용하여 저장 시스템을 구축하였다. 또한 쇼핑몰에서 얻은 상품들의 정보를 저장시스템에 의해 저장하여 보았다. 그리고, 상품의 공통 정보를 Header에 포함하지 않는 별도의 모델에 대하여 저장 시스템을 구현하고, 동일한 상품들을 적용하여 저장한 후, 이들 두 모델간의 중복성 제거율에 대한 분석을 해보았다. 분석의 결과로 37%의 저장공간 절약과 30%의 파일용량 절약, 그리고 12%의 저장시간 절약을 가져옴으로써, 네트워크를 이용한 파일 전송 및 저장시 많은 시간과 공간의 절감 효과를 가져올 것으로 예상된다. 또한 컴포넌트로 구현하였기 때문에 별도의 수정 없이 통합 상품 표현 모델을 기반으로 하는 시스템에 손쉽게 사용할 수 있을 것이다.

향후 연구과제로는 검색 컴포넌트와 갱신 컴포넌트를 개발하여, 쇼핑몰등의 구축시 손쉽게 사용할 수 있는 통합 컴포넌트를 제작할 계획이다.

참 고 문 헌

[1] 조현성, 박찬규, 송병열, 오수영, 김록원, 김경일, 조현규, 함호상, "XML 기반 전자상거 래 프레임워크 기술", 정보과학회지, 제19권 제1호, p.38, 2001.

[2] "Extensible Markup Language (XML)," http://www.w3.org/XML/, 2000.

[3] 김경래, 하상호, 서건수, "XML 기반 통합 상품 표현 모델", 정보과학회학술대회, 2001.

[4] Paul tremblett, "Instant Enterprise JavaBeans," _, McGraw-Hill, 2001.

[5] George T. Heineman, William T. Councill, "Component-Based Software Engineering," Addison Wesley Pub, 2001.

[6] Dale Rogersone, "Inside COM," _, MicroSoft press. 1997.

[7] Duane K. Fields, Mark A. Kolb, "Web Development with Java Server Page," _, MANNING, 2000.

[8] Oracle Corporation, "Oracle8i, The XML Enabled Data Management System," 1999.

[9] Kevin Loney, George Koch, "Oracle8i : The Complete Reference," _, McGraw-Hill, 2000.

[10] Hiroshi Maruyama, Kent Tamura, Naohiko Uramoto, "XML and Java : Developing Web Applications," _, Addison-Wesley, 1999.

[11] Christopher Taylor, "Essential JNDI," 1 edition, Prentice Hall, _.

김 경 래

e-mail : krkim@java.sch.ac.kr

2000년 순천향대학교 전산학과(학사)

2000년~현재 순천향대학교 정보처리학부 (석사과정)

관심분야 : XML응용, 컴포넌트 모델링, 전자상거래, 데이터베이스

하 상 호

e-mail : hsh@sch.ac.kr

1988년 서울대학교 계산통계학과(학사)

1991년 서울대학교 계산통계학과(석사)

1995년 서울대학교 전산학과(박사)

1995년~1996년 한국전자통신연구원 박사후 연구원

1996년~1997년 MIT LCS 박사후 연구원

1997년~현재 순천향대학교 정보기술공학부 조교수

관심분야 : 프로그래밍 언어, 병렬 및 분산 처리, XML