

# 전자상거래 플랫폼에서의 객체지향 상품 클래스 설계기법

최 창호, 김 수동  
승실 대학교 전자계산학과

## An Object-Oriented Design Model for Electronic Commerce Product Database

Chang Ho Choi, Soo Dong Kim  
Dept. of Computer Science, Soong Sil University

### Abstract

현재 운영중인 대부분의 전자상거래 시스템들은 서적, 의류, 컴퓨터등 특정 상거래 분야에 종속적으로 데이터베이스 및 운영 시스템이 개발되어 있다. 그러나, 전자상거래가 본격적으로 활성화 되면 거의 모든 상품 및 서비스가 전자상거래의 대상이 될 것이므로, 전자상거래 제품 데이터베이스의 설계를 여러 도메인에서 다양한 상품들의 정보를 저장할 수 있도록 범용적으로 설계하여야 한다.

본 논문에서는 전자상거래 공통 플랫폼 개발의 중요한 구성요소인 상품 데이터베이스의 클래스를 객체지향 방식으로 설계하여 재사용성, 적응성, 범용성을 극대화하는 기법을 제안한다. 또한, 제안한 객체지향 상품 클래스 설계기법을 이용한 사례연구를 통하여 그 적용방법을 구체적으로 설명한다.

## 1. 서론

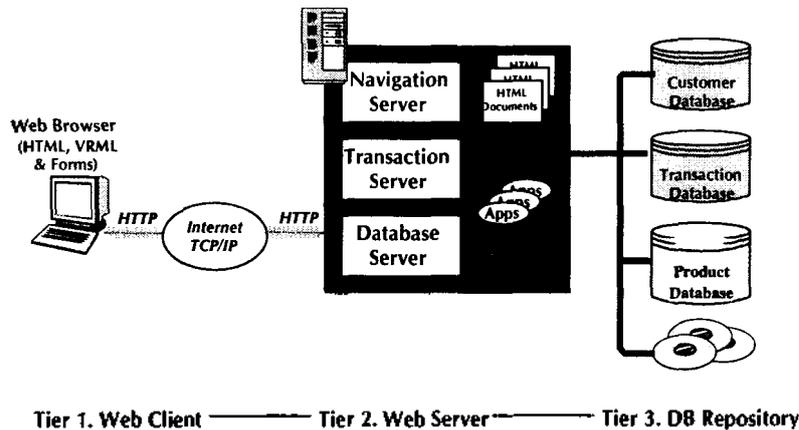
산업계에서 EC 시스템과 그 응용에 관심이 태동하고 있으나, 특정 도메인에 대한 개별적인 EC 시스템 구축이 주를 이루고 있다. 전산 선진국에서 EC 시스템을 다수 운영 중에 있으나, 이들 시스템들의 사용자 인터페이스가 그래픽과 오디오등 단순 타입에 그치고 있고, 사용되는 데이터베이스도 대부분 관계형 데이터베이스 이다. 즉, 현재 EC 시스템들은 응용 도메인에 종속적인 구조를 가지고 있어 플랫폼 연구가 미흡한 실정이므로, 이에 대한 연구가 절실히 필요한 실정이다.

## 2. 전자상거래 플랫폼 S/W 의 구성

그림 1에서와 같이 EC 플랫폼은 3Tier 아키텍처로 이루어져 있다. 클라이언트에서는 고객이 브라우저를 통해 상품을 검색하고 주문을 하도록 구성되며, 자바 애플릿을 사용하여 동적인 사용자 인터페이스를 구축한다.

서버는 역할에 따라 Navigation, Transaction, Database 서버로 구분되며, 사용자 인터페이스를 위한 HTML 문서와 애플릿, 그리고 이를 지원하기 위한 클래스 파일들이 위치한다. 서버에는 어플리케이션 개발에 필요한 클래스와 인터페이스들이 프레임워크(Frameworks) [1]을 이루고 있으며, 어플리케이션은 프레임워크의 클래스와 인터페이스를 상속하고 집산화하여 구성한다.

데이터베이스에는 고객의 정보, 트랜잭션 정보 및 상품 정보가 구축되며, 객체지향 어플리케이션에 적합한 객체지향 데이터베이스가 위치한다.



Tier 1. Web Client ——— Tier 2. Web Server ——— Tier 3. DB Repository

그림 1. 전자상거래 플랫폼 아키텍처

시스템의 적용성(Applicability), 확장성(Extendibility), 유지보수 용이성(Maintainability)을 고려하여, EC 플랫폼은 객체지향 어플리케이션 프레임워크(Object-Oriented Application Frameworks) [2]의 구조를 갖는다. EC 프레임워크는 Object Management Group's Object Management Architecture (OMG OMA) [3]을 기반으로 한다. OMG's OMA 는 분산 컴퓨팅을 위한 Infrastructure 를 제공한다. 그림 2에서 보듯이 CORBA 의 ORB 를 통하여 고객은 다양한 서비스를 받을 수 있으며, 가상 상점 어플리케이션을 개발하려는 개발자는 ORB 에 등록된 객체들을 이용하여 다양한 어플리케이션을 쉽게 구축할 수 있다.

EC 플랫폼은 아키텍처, 데이터베이스 설계, 트랜잭션 정의를 바탕으로 핵심 모듈을 개발하며, 이렇게 정의된 핵심 모듈로 EC 응용 도메인에 맞게 데이터베이스를 구축하고, EC 프레임워크에서 제공되는 클래스를 상속과 집산화 하여 확장 또는 도메인에 맞는 클래스를

재정의 하며, 사용자 인터페이스를 추가하여 가상 상점 어플리케이션을 개발한다.

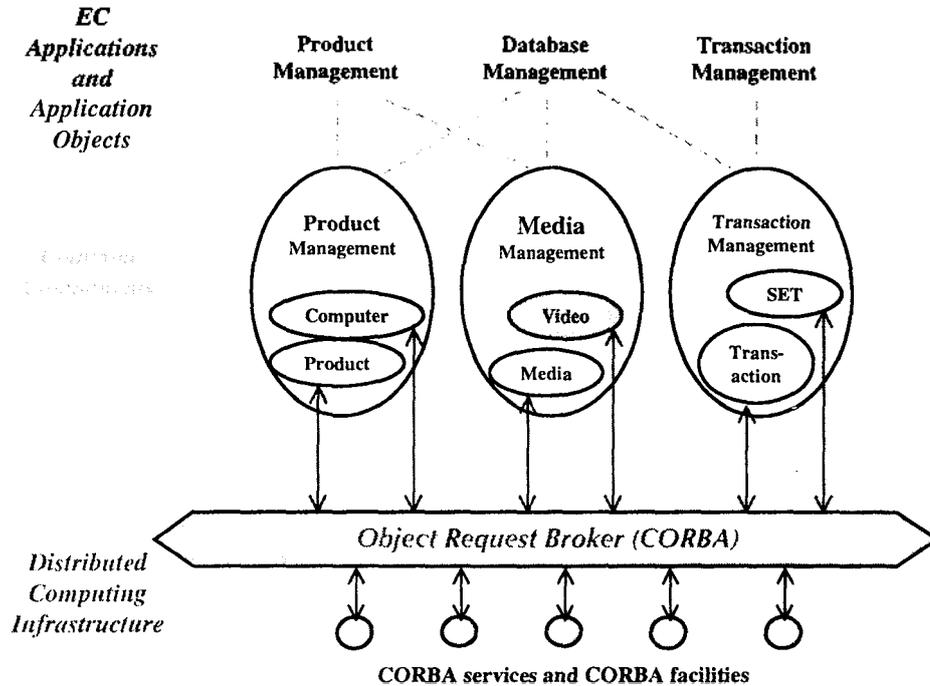


그림 2. EC 프레임워크 아키텍처

EC 플랫폼은 다양한 응용 도메인에 폭 넓게 사용되며 확장성을 높이기 위하여 객체지향 데이터베이스를 전제로 하며, 지원할 데이터베이스의 데이터 유형은 텍스트, 그래픽 등의 기본 타입은 물론 오디오, 비디오, JPEG, MPEG, 다양한 멀티미디어 타입, 3-D 및 가상현실을 지원해야 한다.

또한, 사용자로부터의 상품, 정보 및 서비스의 주문에서부터 결제과정을 거친 후 원하는 상품 및 서비스들이 전달된 후 사용자의 확인까지 이르는 트랜잭션 처리 기술과 현금결제, 신용카드결제 방식을 포함하여 디지털 화폐, E-Cash 등 다양한 유통 수단 및 결제를 위한 암호화와 보안기술이 필요하다.

위에서 검토한 바와 같이 EC 플랫폼 구축은 EC 어플리케이션을 개발하는 것보다 난이도가 높으며, 고도의 기술을 필요로 한다. 그 중에서도 상품 정보의 구축과 트랜잭션 처리를 위한 암호화와 보안기술이 핵심 기술이다. 본 논문에서는 앞에서 밝힌 바와 같이 상품 정보의 구축에 대한 가이드라인 및 이의 활용방법에 대한 가이드라인을 소개하고, 구축 기법을 사례를 통하여 고찰해 볼 것이다.

### 3. 객체지향 상품 클래스 설계 기법

전자 상거래 플랫폼의 상품 클래스는 적용성과 확장성이 좋아야 한다. 플랫폼을 사용해 개발자들이 쉽게 가상상점을 구축할 수 있어야 하며, 어떠한 도메인에 대해서도 잘 적용되어야 한다. 그리고, 새로운 기능을 추가할 경우에도 쉽게 확장할 수 있는 구조를 가져야 한다.

객체지향 상품 클래스는 위와 같은 요건을 충분히 만족해야 하며, 다음을 만족하기 위해 상품 클래스들은 그림 3에서와 같이 거대한 클래스와 인터페이스의 상속과 집단화 구조로 이루어져 있다.

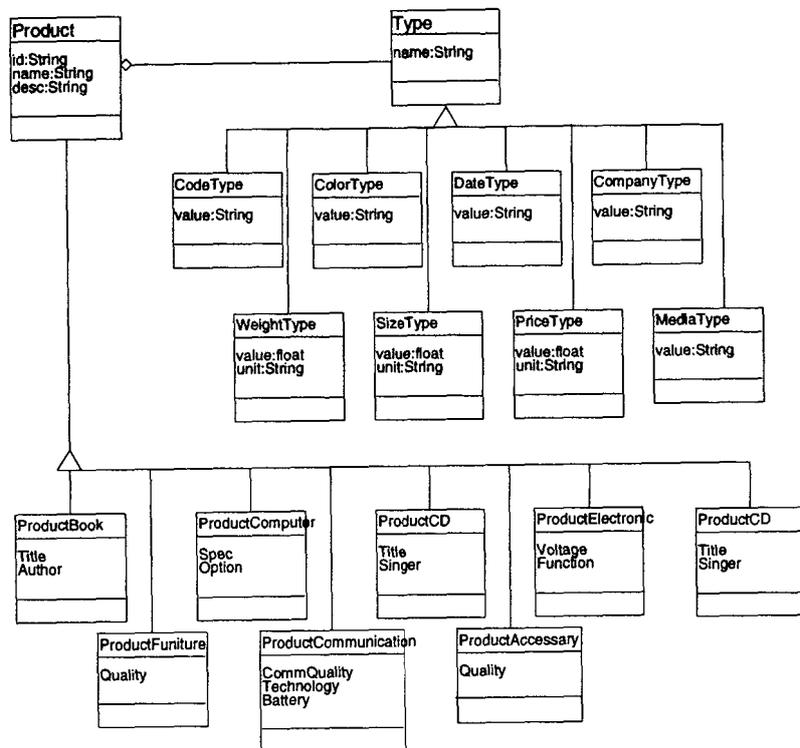


그림 3. 상품 클래스 다이어그램

상품 클래스는 구성 방법에는 몇 가지 방법이 있을 수 있다. 처음으로는 상품의 Attribute 별로 Type 을 정의하는 방법인데, 기본 Type 을 정의하고 다시 이를 상속 받은 Subtype 을 구성한 후 상품 클래스가 이들을 집단화(Aggregation) 하는 방법이다. 다음으로는 상품의 Attribute 를 Type 에 따라서 Subtype 으로 정의하고 이들을 집단화 하는 방법인데, polymorphism 을 이용하여 Subtype 의 추상 클래스를 배열로 선언하여 Subtype 을 갖으므로 Type casting 이 필요하다. 또 다른 방법은 상품 클래스에 필요한 Attribute 를 모두다 선언하여 사용하는 방법인데, 이는 도메인별로 필요한 Attribute 와 불필요한 Attribute 가 차이가 많이 생길 경우, 확장성과 적용성 표현성 면에서 뒤진다. 본 논문에서는 처음의 방법을 사용할 것을 추천하며, 구성은 그림 3과 같다.

이하에서는 위에서 추천한 방법에 대해 각 특성 별로 상품 클래스 구축에 대한 가이

드라인을 제시하면서 어플리케이션 구축의 용이함을 살펴보고자 한다.

### 3.1. 적용성 (Adaptability)

적용성을 향상시키기 위해 EC 플랫폼은 상품 도메인별로 가장 적절한 클래스들을 제공해야 한다. 즉 상품을 도메인별로 구분하여 각 도메인의 특성을 최대한 반영할 수 있는 클래스들을 제공해야 한다. 이렇게 도메인별로 분류된 상품 클래스들은 개발자가 가상상점을 개발할 때 필요한 상품 클래스들을 선별하여 가상상점을 구축할 수 있도록 한다.

### 3.2. 확장성 (Extendibility)

확장성 측면에서 EC 플랫폼은 모든 도메인에 대해 상품 클래스를 제공할 수 없으므로, 이들 상품 클래스들의 공통 Attribute 와 method 를 추출하여 일반적인 상품 클래스로 제공해야 한다. 이렇게 제공된 일반적인 상품 클래스는 개발자가 실제로 구축할 상품과 특수한 상품 클래스를 추가하기 쉽도록 도와준다.

그림 3에 제시된 Class Diagram 은 적용성과 확장성을 고려하여 상품 클래스를 구성한 모델이다. 상품 클래스로부터 상속 받아 도메인별로 분류된 각각의 상품 클래스들은 개발자가 가상상점을 구축하는데 편리하게 이용될 수 있다.

### 3.3. 표현성 (Expressiveness)

또한, 상품을 주문하려는 고객의 구매욕구를 높이려면, 개별 상품의 특성을 최대한 살려서 보여주어야 한다. 같은 종류의 상품이라도 상품마다 고유의 Attribute 를 갖는 경우가 적지 않으므로, 상품 클래스는 Attribute 의 추가와 삭제가 용이하도록 구성해야 한다. 이렇게 구성된 상품 클래스는 각 Attribute 마다 클래스 구조를 갖으므로 복잡한 구조를 갖지만, 상품의 표현성을 높여준다는 측면에서 그 가치가 있다.

## 4. 상품 클래스 활용 기법

상속과 집단화는 객체지향[4] 기술의 대표적인 기술로, 클래스의 적용성, 확장성과 표현성을 좋게 할 수 있다. EC 어플리케이션인 가상상점 어플리케이션 구축 시에는 이미 정의된 EC 플랫폼을 활용할 경우, 사용자 인터페이스 기술과 실제 데이터베이스 구축기술, 그리고, EC 프레임워크의 활용 기술들이 필요하다.

### 4.1. 상속 (Inheritance)

EC 프레임워크의 상품 클래스들은 도메인 별로 정의되어 있으므로, 이를 바로 가상상점 어플리케이션에 사용할 수 있다. 또한, 해당 도메인이 특수한 경우이거나 정의된 상품 클

래스를 확장하고자 할 경우애, 상속을 사용해 새로운 상품 클래스를 정의하고 필요한 Attribute, method 들을 추가하거나 변경이 필요한 method 들은 override 를 통하여 재정의 한다. 이렇게 함으로써, 구축할 도메인에 적합한 상품 클래스를 쉽게 정의할 수 있다.

#### 4.2. 집단화 (Aggregation)

그림 4의 왼쪽에서 보듯이 상품 클래스는 기본 Attribute 외에 여러 가지 Type 을 집단화 하여 갖는다. 어플리케이션에서는 해당 도메인에 맞는 Type 을 새로 정의하여 상품 클래스에서 사용할 수 있으며, 상품에 관련된 멀티미디어 데이터는 미디어 클래스를 집단화 하여 이미지, 오디오, 동영상 등을 표현할 수 있다.

상품 클래스가 적용성, 확장성과 표현성이 좋게 정의되어 있으므로, 어플리케이션 개발자는 상품 클래스가 갖는 Subtype 및 미디어 정보가 변경될 때마다 상품 클래스를 변경할 필요 없이 필요한 정보에 맞는 Type 과 미디어 class 를 집단화 하여 상품 클래스를 구성할 수 있다.

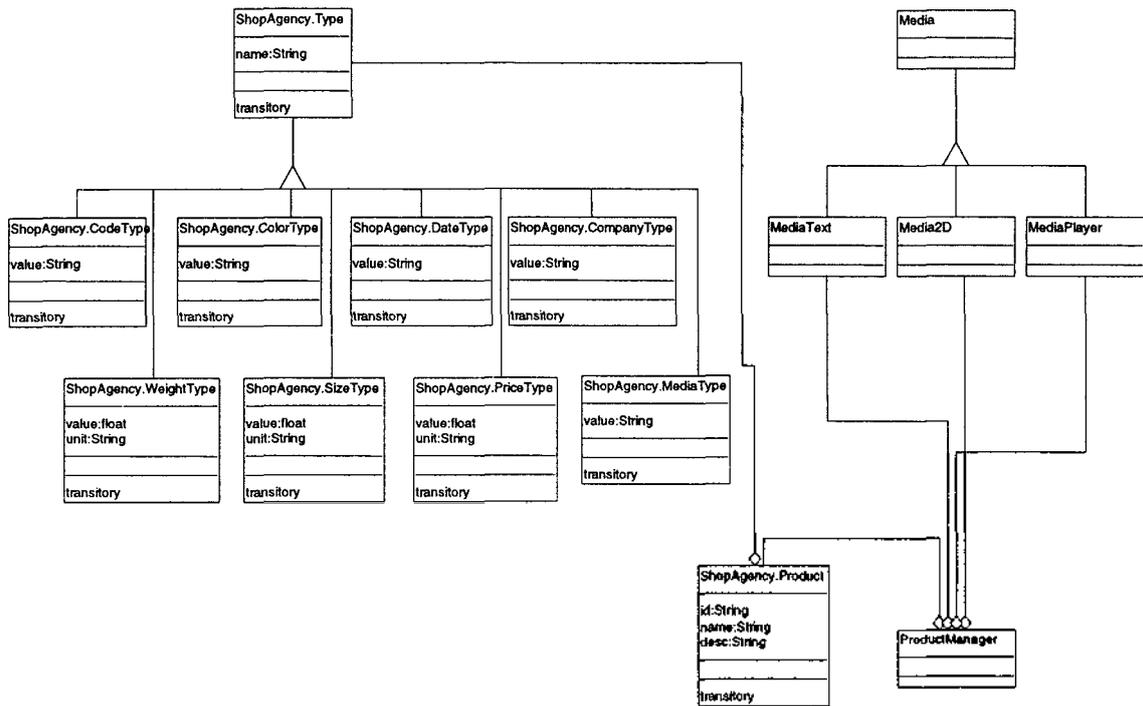


그림 4. 상품 클래스 활용 방안 : 집단화

### 5. 상품 클래스 구축 사례연구

위의 객체지향 상품 클래스 활용방안에서 소개한 가이드라인에 따라 다음과 같은 구축 사례를 소개하고자 한다. 플랫폼의 아키텍처는 다음과 같다.



등과 같은 Subtype 의 배열을 Attribute 로 갖는다. 각 Subtype 은 Type 이라는 추상 클래스로부터 상속을 받아 구성함으로써, 필요한 Subtype 을 상품 클래스에 추가하여 쉽게 확장할 수 있다.

그림 7에서 ShopAgency Module 에는 상품 클래스와 Subtype 들에 대한 클래스가 정의되어 있고, 이를 관리하는 ShopAgencyManager 클래스가 정의되어 있다. 가상 상점 서버에서 상품을 데이터베이스에 추가, 변경, 삭제하는 상품 데이터베이스 관리자 클래스에 대한 IDL 을 나타낸다.

<pre> <b>module ShopAgency {</b> <b>interface Type {</b> <b>attribute wstring name;};</b> <b>interface CodeType:Type {</b> <b>attribute wstring value;};</b> <b>interface ColorType:Type {</b> <b>attribute wstring value;};</b> <b>interface DateType:Type {</b> <b>attribute wstring value;};</b> <b>interface CompanyType:Type {</b> <b>attribute wstring value;};</b> <b>interface MediaType:Type {</b> <b>attribute wstring value;};</b> <b>interface SizeType:Type {</b> <b>attribute float value;</b> <b>attribute wstring unit;};</b> <b>interface WeightType:Type {</b> <b>attribute float value;</b> <b>attribute wstring unit;};</b> <b>interface PriceType:Type {</b> <b>attribute float value;</b> <b>attribute wstring unit;};</b> </pre>	<pre> <b>typedef sequence&lt;CodeType&gt;CodeTypeArray;</b> <b>typedef sequence&lt;ColorType&gt;ColorTypeArray;</b> <b>typedef sequence&lt;DateType&gt;DateTypeArray;</b> <b>typedef sequence&lt;CompanyType&gt;CompanyTypeArray;</b> <b>typedef sequence&lt;MediaType&gt;MediaTypeArray;</b> <b>typedef sequence&lt;SizeType&gt;SizeTypeArray;</b> <b>typedef sequence&lt;WeightType&gt;WeightTypeArray;</b> <b>typedef sequence&lt;PriceType&gt;PriceTypeArray;</b>  <b>interface Product {</b> <b>attribute wstring id;</b> <b>attribute wstring name;</b> <b>attribute wstring desc;</b> <b>attribute CodeTypeArraycodes;</b> <b>attribute ColorTypeArraycolors;</b> <b>attribute DateTypeArraydates;</b> <b>attribute CompanyTypeArraycompanys;</b> <b>attribute MediaTypeArraymedias;</b> <b>attribute SizeTypeArraysizes;</b> <b>attribute WeightTypeArrayweights;</b> <b>attribute PriceTypeArrayprices;</b> <b>};</b> </pre>
---	---

그림 6. 가상 상점 상품 클래스 IDL : 상품 및 Type 클래스

```

interface ShopAgencyManager {
Product open( in wstring id);
boolean delete( in wstring id);
boolean create( in Product p);
boolean update( in Product p);

Product newProduct();
CodeType newCodeType( in wstring name, in wstring value);
ColorType newColorType( in wstring name, in wstring value);
DateType newDateType( in wstring name, in wstring value);
CompanyType newCompanyType( in wstring name, in wstring value);
MediaType newMediaType( in wstring name, in wstring value);
SizeType newSizeType( in wstring name, in float value, in wstring unit);
WeightType newWeightType( in wstring name, in float value, in wstring unit);
PriceType newPriceType( in wstring name, in float value, in wstring unit);
};

```

그림 7. 가상 상점 상품 클래스 IDL : 상품 데이터베이스 관리자 클래스

그림 8은 가상 상점 서버를 나타내는데, 본 사례에서는 상품 정보를 관리하는 부분만을 포함했다. 그림 8은 서버가 ORB 와 BOA, 데이터베이스를 초기화하고 상품 데이터베

이스 관리자 객체를 바인딩 한 후, 가상 상점 서버 객체를 ORB 상에 등록[3,5]하는 과정을 나타낸다. 클라이언트 상품 검색 요청에 대해, "PC166S"이라는 상품을 open 함을 보여 준다.

```

C:\WCafe\Java\Bin\java.exe
Initializing ORB.....
Initializing BOA.....
Binding ShopAgency Manager .....
Initializing ObjectStore.....
Opening Database.....
ShopAgencyManager created!!!!
Binded!!!
ShopAgencyManager[Server, oid=PersistentId[repId=IDL:ShopAgency/ShopAgencyManager
:1.0.objectName=ShopAgency Manager]] is ready.
Open PC166S
  
```

그림 8. 상품 클래스 관리 서버 어플리케이션

Applet

아이디 | PC166S

이름 | 삼성 매직스테이션

설명 | 제품구성: Pentium166 CPU/ 32M RAM/ 2M Video/ 3.2G HDD/ MPEG

(코드) | (크기)

상품코드	P101-201-301	본체높이	400.0	mm
제조일련번호	P970901-00001	본체폭	100.0	mm
		본체너비	400.0	mm

(색상) | (무게)

본체색상	흰색/아이보리색/	본체중량	5.0	Kg
마우스색상	파랑/녹색/아이보	전체중량	7.0	Kg

(날짜) | (가격)

제조일	97.09.01	소매자가	127.0	만원
A/S기간	구입일로부터2년	판매가	115.0	만원
		공정도가	104.0	만원

(회사) | (미디어)

제조회사	삼성전자	사진	PC166S.gif
판매회사	삼성유통	오디오	PC166S.wav

[검색] [삭제] [수정] [추가] [대장] [취소]

검색되었습니다.

그림 9. 상품 클래스 관리 클라이언트 애플릿

그림 9는 상품 클래스를 관리하는 애플릿을 사용하여, 상품공급자가 컴퓨터 상품을 검색하는 화면이다. 상품 클래스 관리 클라이언트는 애플릿으로 작성되어 있으므로, 여러 상품 공급업자가 원격지에서 브라우저를 통해 쉽게 상품을 관리하는 것을 가능하게 한다.

상품의 코드는 판매시의 상품코드와 제조사의 일련 번호 등이 있을 수 있으며, 가격에

도 소비자가격, 공장도가격, 실제 판매가격 등이 존재할 수 있다. 즉, 각 Attribute 에 대해서 여러 가지의 Subtype 을 갖을 수 있다. 구축된 상품 정보는 상품마다 다른 고유의 Subtype 에 대해 설명(description)과 값(value)을 두어 이를 허용한다.

고객이 상품을 검색할 경우에 상품에는 미디어 정보를 추가하고 해당 미디어 클래스를 바인딩하여 고객에게 보여줄 수 있으며, 이 애플릿이 실제로 가상상점 구축 시에 고객에게 보여질 HTML 문서에 포함된다. 그림 9에 소개한 클라이언트 애플릿은 상품 클래스 구축에 대한 한가지 예이며, 상품 클래스가 서버에 구축되면, 가상 상점 구축 개발자는 다양한 사용자 인터페이스를 사용하여 고객에게 보여질 화면이나, 상품 정보 관리 Tool 을 개발할 수 있다.

## 6. 결론

이상에서 EC 플랫폼과 EC 프레임웍에서 상품 클래스 구축이 차지하는 비중과 역할, 구축에 관한 가이드라인, 그리고, 상품 클래스 활용방안 및 구축사례를 살펴보았다. EC 어플리케이션 개발이 표준이나 Infrastructure 없이 개발되고 있는 시점에서 EC 플랫폼 개발의 필요성이 대두되고 있고, 이에 필요한 상품 클래스의 효율적인 구축 방법에 대한 지속적인 연구가 필요해 지고 있다.

본 논문은 이러한 EC 어플리케이션 개발을 촉진시키고, 품질을 높일 수 있는 방안으로 EC 플랫폼과 EC 프레임웍의 도입을 제안하며, 이의 개발에 필요한 상품 클래스의 효율적인 구축 방안에 대해 제시하였다.

## 참고문헌

- [1] Johnson, R. "Frameworks = (Component + Patterns)", CACM, Oct. 1997.
- [2] Fayad, M. and Schmidt, D., "Object-Oriented Application Frameworks", CACM, Oct. 1997.
- [3] Object Management Group. The Common Object Request Broker: Architecture and Specification. Revision 2.0, July 1995. Object Management Group, Framingham, Mass., 1995.
- [4] James R, Michael B, William P, Frederick E, William L, "Object-Oriented Modeling And Design", Prentice Hall, 1991.
- [5] Visigenic, "VisiBroker for Java", Reference Manual Version 3.0, Visigenic Software, Inc. 1997.

## 저자 소개

### 김 수 동

1984, 전산학 학사, Northeast Missouri State University

1988, 전산학 석사, The University of Iowa

1991, 전산학 박사, The University of Iowa

1991 - 1993 한국통신 연구개발단 연구실장/선임연구원

1993 - 1994 The University of Iowa, 교환교수

1994 - 1995 현대전자 S/W 연구소 책임연구원

1995.9 - 현재 송실대학교 컴퓨터학부 조교수

#### 주요관심분야

객체지향 개발방법론 (UML, Java)

웹 기반 분산 객체 컴퓨팅 (CORBA, Intranet, Client-Server Web, Web Agents)

인터넷 상거래 시스템 기술

### 최 창 호

1995. 전산학 학사, 송실 대학교

1996. ㈜데이콤 SI 사업단

현재, 송실대학교 컴퓨터학부 대학원 (석사과정)

#### 주요관심분야

객체지향 데이터베이스 (OODB)

객체지향 개발방법론 (UML, Java)

웹 기반 분산 객체 컴퓨팅 (CORBA, Client-Server Web)