한국지능정보시스템학회
02 춘계학술대회 논문집, pp. 115~122

전자상거래 시스템을 위한 범용적 O-R Mapping Tool: Object Organizer ™

한상목 · 곽우섭 · 조규찬

A General-purpose Object-Relational Mapping Tool for E-Commerce Applications: Object Organizer ™

Sang Mok Han · Woo Sub Kwak · Kyu Chan Cho

요약

O-R 매핑툴을 이용하면 업체별로 상이하며 가변적인 요소를 포함하는 전자 상거래 시스템을 효과적으로 구현할 수 있다. 본 논문에서는 전자상거래 시스템을 위한 범용적 O-R Mapping Tool인 Object Organizer ™를 소개하고, 이를 설계하고 구현하기 위해 이용된 접근법에 대해 설명한다. 기존의 O-R 매핑툴에 대한 분석을 통해 사용 편의성과 수행 성능을 개선할 수 있도록 설계하였으며 전자 상거래 시스템의 구현 방법에서 일반화될 수 있는 부분을 추출하기 위하여 개발 프로세스와 구현 방식을 분석하였다. 또한 전자 상거래 시스템의 요구 사항을 충분히 반영할 수 있는 애플리케이션 도메인을 선정하여 현업의 구체적인 요구 사항을 효과적으로 처리 가능하도록 설계하였다.

본 논문은 전자 상거래 시스템에 범용적으로 사용될 수 있는 O-R 매핑툴의 구현 방향을 제시했으며 Object Organizer ™를 통해 이를 구현하였다. O-R 매핑툴의 범용성을 확보하기 위해 기존의 제품을 분석하고 전자상거래 시스템의 구현 방법을 분석하고 일반화하였으며 실제 전자 상거래 애플리케이션을 구현해 봄으로써 Object Organizer ™의 개발 효율성을 평가하였다.

Key words : Electronic commerce, O-R Mapping, Database, Java, Development

1. 서론

전자 상거래 시스템을 개발하는 과정은 목표 시스템을 위한 다양한 비즈니스 컴포넌트를 개발하여 통합하는 과정으로 볼 수 있다. 비즈니스 컴포넌트는 주문, 검색과 같은 다양한 트랜잭션을 수행하며 동시에 비즈니스 데이터를 영속성 있게 저장하는 기능을 한다. 데이터베이스 시스템은 다양한 비즈니스 트랜잭션을 수행하는 전자 상거래 시스템을 구현하기 위해 필수적으로 채택되는 시스템 구성 요소이다.

일반적으로 트랜잭션 처리를 위해선 오브젝트와 관계형 데이터베이스 사이의 대응 관계를 이용하여 오브젝트의 변화를 적절한 트랜잭션 요청으로 변환해 주는 컴포넌트를 이용한다. 개발 스케줄링에 있어서 이러한 컴포넌트를 구현하기 위한 기간은 전체 개발 기간의 50%~70%를 소요한다. 이는 개발에 사용되는 객체지향 언어의 오브젝트를 관계형 데이터베이스에 직접 대응시켜 트랜잭션을 처리할 수 없기 때문이다.

O-R 매핑툴이란 오브젝트와 관계형 데이터베이스 간의 매핑을 설정함으로써 손수 구현해야 하는 데이터베이스 트랜잭션 코드 작성을 생략하고 구현 과정을 자동화 해주는 역할을 하는 개발 도구이다. 자바 애플리케이션에 일반적인 적용될 수 있는 O-R 매핑툴은 기존에 시도된 바 있으나 적용 편의성, 확장성, 유지 보수성 등에서 한계점을 가지고 있었다.

본 논문에서는 전자 상거래 시스템을 개발 과정을 효율화하고 시스템의 확장성 및 유지 보수성 향상시키는 범용적인 O-R 매핑툴의 설계를 위해 다음과 같이 접근하였다.

(1) O-R 매핑툴의 범용성을 시험할 수 있는 전자상거래 애플리케이션 도메인을 선정.

(2) 해당 애플리케이션에 전형적인 개발 방식을 적용하고 개발 단계 별로 일반화.

(3) 기존 O-R 매핑툴의 보완점을 점검하고 개선하여 설계에 반영.

위와 과정을 통해 설계된 O-R 매핑툴인 Object Organizer ™는 사용 편의성, 개발 효율성, 유지 보수 및 확장 편의성을 제공한다. 본 논문에서는 2.제안 배경을 통해 기존의 O-R 매핑툴 제품이 지닌 한계점을 분석하였으며 3. 타겟 애플리케이션 도메인 선정, 4. 개발 프로세스 분석, 5. 일반적인 웹 애플리케이션 구현 방법 6. 구현 과정의 일반화를 통해 O-R 매핑툴을 설계하였다. 7. O-R 매핑툴의 구현에서는 전자상거래에 범용적으로 사용될 수 있는 O-R 매핑툴인 Object Organizer ™를 구현하였다. 8.개발 효율성 비교에서는 O-R 매핑툴을 적용하여 전자 카탈로그 애플리케이션을 구현함으로써 그 유용성을 평가하였다.

2. 제안 배경 및 개선 방향

O-R 매핑툴 제품은 단순한 기능을 제공하는 공개 소스의 제품부터 다양한 기능을 가진 고가의 제품까지 상당히 다양하다. 관련 표준으로는 Sun의 J2EE 의 CMP와 JDO, ODMG 3.0이 있으며 이를 통해서 데이터베이스에 대한 영속성 구현 방식을 표준화하려는 노력을 계속해오고 있다. 이러한 시도는 이용 편의성, 수행 성능 면에서 한계를 지니고 있기 때문에 아직 널리 환영 받지 못하는 상황이다. [3]

2.1 이용 편의성의 문제점

개발자의 입장에서 O-R 매핑툴은 간편하게 영속성을 구현하기 위해서 사용된다. J2EE의 CMP의 경우 클래스를 영속성 있게 저장하기 위해선 XML 파일을 수정하고 관련 클래스와 인터페이스를 구현해야 한다. J2EE 표준을 지원하는 웹 애플리케이션 서버를 사용해야 하는 것도 부담으로 작용하며 대량의 데이터를 취급하는 경우 JDBC를 직접 이용하는 것에 비해 상당히 낮은 성능을 보인다. 이러한 점은 J2EE의 CMP가 구현하고 있는 다양한 기능에도 불구하고 널리 환영 받지 못한 원인이 되었다. [4]

일부 O-R 매핑툴 제품은 자체적으로 트랜잭션을 관리하고 오브젝트의 상태의 변화를 모니터링하기 위해 추가적인 코드를 작성해야 한다. 기존 방식으로 처리 가능했던 트랜잭션 관리를 O-R 매핑툴이 제공하는 방식으로 변경하는 작업은 번거로우며 성능면에서 차이가 없다.

2.2 수행 성능의 문제점

O-R 매핑툴이 제공하는 API를 이용하여 코드를 작성하게 되면 Update를 위해서 우선 Select로 커서를 위치시켜야 하는 등 불필요한 SQL 쿼리를 실행하게 된다. 또한 오브젝트의 상태를 변화를 모니터링하기 위해 트랜잭션의 시작과 끝을 알리는 추가적인 코드를 작성해야 하며 오브젝트의 사본을 저장하여 변경된 부분을 찾아내기 위한 계산도 수행하게 된다. J2EE의 CMP를 이용하면 레코드의 수가 커지는 경우 오브젝트 풀링(object pooling)의 영향으로 성능이 저하되는 경향을 보인다.

2.3 개선 방향

새로운 O-R 매핑툴인 Object Organizer ™는 기존 제품들에 비해 이용 편의성을 개선하기 위해서 영속성을 구현하는 절차를 단순화 하여야 한다. J2EE의 CMP 경우와 같이 영속성을 구현하기 위해서 관련된 클래스를 추가 하는 것은 적절하지 않은 접근 방식이며 특정 클래스를 상속 받아야 하는 제약을 두는 것도 바람직하지 않다.

기존의 다른 API를 통해 해결 가능한 부분은 기존의 API와 통합될 수 있도록 구현한다. 이미 존재하는 트랜잭션 처리 기능을 다시 개발(re-invent)하기 보다는 기존의 트랜잭션 처리 방식과 쉽게 통합될 수 있도록 설계한다. 성능 저하를 방지하기 위해서 단지 오브젝트 지향적인 프로그래밍을 위해 불필요한 모니터링 기법을 사용하지 않는다. 오브젝트 풀링(object pooling) 기법은 너무 많지 않는 레코드가 존재하는 경우 유용하게 사용될 수 있으므로 선택적으로 적용한다.

설정 정보를 편리한 사용자 인터페이스를 통해서 수정할 수 있게 함으로써 설정 파일 내의 텍스트를 손수 편집할 필요가 없도록 하고 설정 파일 내용의 오류를 방지한다. 설정 정보를 자동 입력하는 기능을 이용하여 빠르고 편리하게 매핑을 설정할 수 있도록 한다.

설정 정보를 하나의 파일로 통합 관리한다. 설정 정보를 통합 관리함으로써 설정 정보 파일을 복사하는 것만으로 모든 매핑 정보를 백업하고 다른 서버에 그대로 적용할 수 있다. 설정 정보가 XML로 저장되어 있는 경우 설정 파일을 읽기 위해서 Runtime의 클래스패스에 XML 처리 API가 포함되어 있어야 하는 단점이 있다. 자바 기본 API로 처리 가능한 Properties 파일을 이용하여 설정 정보를 저장한다.

3. 타겟 애플리케이션 도메인 선정

3.1 전자 상거래 애플리케이션

전자상거래 애플리케이션은 주문 관리, 고객 프로파일, 전자 카탈로그, 재고 관리 컴포넌트 등으로 구성된다. 고객 프로파일은 고객의 상세 정보를 조회, 수정, 삽입, 삭제할 수 있는 컴포넌트이며 데이터베이스에 여러 개의 테이블로 분산 저장되는 경우도 지원한다. 전자 카탈로그는 제품 분류나 검색 옵션을 적용하여 고객들에게 제품 정보를 제공하며 제품에 대한 정보를 삽입, 수정, 삭제할 수 있는 기능을 제공한다. 고객 프로파일과 전자 카탈로그는 전자상거래 애플리케이션의 가장 핵심적인 데이터를 관리하는 컴포넌트로써 뛰어난 확장성을 요구하고 유지 보수가 용이해야 한다는 점에서 O-R 매핑툴의 범용성을 시험하기에 적합한 컴포넌트라고 판단된다.

3.2 전자 카탈로그의 요구 사항

전자 카탈로그는 브라우징 기능, 검색 기능, 제품 전시 기능을 가진다. 브라우징 기능은 카테고리에 따라서 제품을 분류하여 보여주는 기능이며 카테고리는 트리 형태로 조직화될 수 있다. 제품이 가지는 속성은 제품의 종류에 따라 상이하기 때문에 전자 카탈로그를 저장하는 테이블은 취급하는 제품별로 상이한 속성을 유연하게 수용할 수 있어야 한다.

검색 기능은 고객이 원하는 제품을 카탈로그에서 찾을 수 있도록 지원하는 기능이다. 기업별, 제품별로 상이한 제품의 다양한 속성을 이용해서 검색을 수행할 수 있도록 지원해야 한다.

제품 디스플레이 기능은 제품의 상세 정보를 화면에 표시해주는 기능이다. 제품의 상세 정보에는 재고 관련 정보, 연관 상품 정보 등이 포함될 수 있으므로 여러 테이블에 분산되어 있는 제품 관련 정보를 수집하여 표시할 수 있도록 지원해야 한다.

3.3 범용성의 판단 기준

범용적으로 사용될 수 있는 전자 카탈로그는 업체나 제품에 관계없이 전자 상거래 시스템을 구축하는데 이용될 수 있는 컴포넌트이다. 기본적으로 범용성은 전자 카탈로그의 정의를 코드에 하드 코딩된 형태로 포함시키지 않고 별도 데이터로 분리함으로써 가능하며 O-R 매핑툴은 이러한 기술을 가능하게 해준다. 전자상거래에 범용적으로 적용될 수 있는 O-R 매핑툴이 갖춰야 할 요건을 위의 요구사항을 기반으로 정리하면 다음과 같다.

(1) 여러 테이블에 분산되어 저장되어 있는 제품 정보를 하나의 오브젝트로 통합하여 수집

(2) 제품별, 업체별로 상이한 제품의 속성을 유연하게 수용

(3) 제품이 지닌 다양한 속성을 이용하여 검색

4. 개발 프로세스의 분석

애플리케이션을 개발하는 프로세스는 방법론에 따라서 다양한 방식이 나타난다. 본 논문에서는 각 방법론에 대한 따라 나타날 수 있는 적용에서의 차이점을 최소화하기 위해서 간단히 설계 단계, 구현 단계, 시스템 이양 및 유지 보수 단계의 세 단계로 나누어 접근한다.

4.1 설계 단계

이 단계는 구현하고자 하는 전자 상거래 시스템의 클래스와 데이터베이스 테이블, 하드웨어 구성을 설계하는 단계이다. 일반적으로 고객으로부터 요구 사항을 수집하여 목표 시스템에 구현될 클래스와 데이터베이스 테이블에 대한 설계 산출물을 작성하는 단계이다. 이 단계의 산출물 중에서 구현 단계에서 직접적으로 사용되는 산출물로는 클래스 명세서와 데이터베이스 테이블 명세서를 들 수 있다. 개발 방법론에 따라선 설계 단계에 반복 과정(Iteration)이 존재할 수도 있다.

4.2 구현 단계

이 단계는 설계 단계에서 정의한 클래스와 데이터베이스 테이블을 구현한다. 일반적으로 설계된 클래스는 그 기능에 따라 Model에 해당하는 클래스, Controller에 해당하는 클래스, View에 해당하는 클래스로 구분된다. 각 클래스는 다음과 같은 기능을 가진다.

Model 클래스는 데이터 저장 및 조회를 위한 기능을 구현한다. Model 클래스를 구현하는 방식은 다양한 방식이 존재할 수 있는데 본 논문에서는 데이터 자체를 표현하는 클래스와 데이터베이스 트랜잭션을 담당하는 클래스로 분리하여 처리하는 방식을 택한다. Controller 클래스는 비즈니스 로직을 구현한다. 비즈니스 로직을 담당하는 클래스는 내부적으로 Model 클래스를 이용하여 비즈니스 트랜잭션을 실행하는 형태로 구현된다. View 클래스는 사용자 가 이용하는 화면 및 인터페이스를 구현한다. View 클래스는 전자 상거래

시스템의 용도에 따라서 매우 다양한 인터페이스를 가질 수 있는데 본 논문에서는 웹 기반의 시스템의 경우를 다루며 서버 사이드 스크립트(Server-side scripting) 기술을 이용한다.

4.3 시스템 이양 및 유지 보수 단계

이 단계는 구현 단계에서 구현된 전자 상거래 시스템을 통합 테스트하는 작업, 주문자의 하드웨어 환경으로 배치하는 작업, 추후 신규 요구 사항 발생에 따른 유지 보수 작업이 포함된다. 본 논문에서는 유지 보수 작업 중 전자 상거래 시스템이 취급하는 제품의 형태가 바뀌어 전자 카탈로그에 수정이 발생하는 경우나 고객에 대해 관리하는 데이터가 추가되는 경우 등에 대한 처리 방식 및 유지 보수 용이성에 관심을 갖는다.

5. 일반적 웹 애플리케이션 구현 방식

설계, 구현, 시스템 이양 및 유지 보수의 세 단계의 구현 프로세스를 통해 전자 카탈로그를 웹 애플리케이션으로 구현한다. 시스템에 이용되는 각 클래스를 구현하는 일반적인 접근 방식을 분석한다.

5.1 전자 카탈로그 설계

전자 카탈로그를 구현하기 위해서 필요한 클래스 명세서와 데이터베이스 테이블 명세서를 작성한다. Model 클래스는 Product라는 제품 데이터를 담는 클래스와 데이터베이스 트랜잭션을 처리하는 ProductMgr라는 클래스로 설계되었다. 전자 카탈로그에 대한 다양한 비즈니스 로직을 구현하는 Controller 클래스는 Catalog 라는 클래스로 설계되었으며 카탈로그 브라우징, 제품 검색, 제품 상세 보기, 제품 정보 수정을 지원한다. 사용자의 인터페이스를 구현하는 View 클래스는 관련된 여러 Jsp 페이지로 설계되었다.

전자 카탈로그의 트랜잭션을 처리하고 영속성 있게 데이터를 저장하는 데이터베이스 테이블은 EC_PRODUCT라고 이름을 가지며 일련번호, 제품명, 가격, 적립금, 관련 제품의 필드를 가지고 있다. 이 테이블은 ProductMgr 클래스를 이용하여 트랜잭션을 처리할 수 있다.

5.2 데이터 모델 클래스 구현

Product은 Model 클래스 중 제품 데이터를 담는 클래스로서 제품의 각 속성을 설정하고 반환하는 메쏘드를 가진다.

5.3 데이터베이스 트랜잭션 클래스 구현

ProductMgr은 카탈로그의 제품 데이터가 들어있는 OD_PRODUCT 테이블에 대한 트랜잭션을 수행한다. 브라우징, 검색, 상세 보기를 위한 검색 기능과 정보 변경을 위한 수정 기능이 구현된다.

5.4 비즈니스 로직 핸들링 클래스 구현

Catalog는 전자 카탈로그의 비즈니스 로직을 구현하는 클래스로서 구체적인 비즈니스 트랜잭션에 관련된 메쏘드를 가지고 있다. 이 클래스는 데이터베이스 트랜잭션 클래스인 ProductMgr을 호출하여 데이터베이스 트랜잭션을 수행하며 데이터 모델 클래스인 Product의 목록을 반환 받는다. 카탈로그 수정과 같은 비즈니스 트랜잭션의 경우 트랜잭션의 커밋(commit) 여부를 결정하는 구현도 이 부분에 포함된다.

5.5 사용자 인터페이스 구현

Catalog의 인터페이스는 웹 인터페이스로 구현되며 카탈로그 정보를 보여주는 여러 페이지로 구성된다. 브라우징 페이지는 카탈로그를 탐색할 때 카테고리 정보와 상품 목록을 표시하기 위해 사용되며 browse.jsp 라는 이름을 갖는다. 상세 검색 페이지는 검색 조건 입력 페이지와 검색 결과 페이지로 구성되며 search.jsp 라는 이름을 갖는다. 상세 보기 페이지는 제품에 대한 상세 정보와 연관 제품에 대한 목록을 보여 주기위한 것으로서 detail.jsp라는 이름을 갖는다. 제품 정보 수정 페이지는 제품에 대한 속성 정보를 수정하기 위한 것으로서 edit() 라는 이름을 갖는다.

6. 구현 과정의 일반화

전자 카탈로그에 사용될 각 클래스의 구현 방식과 설계 내용을 기반으로 일반적으로 적용될 수 있는 설계 방식을 추출하고 O-R 매핑툴을 이용하여 이를 자동화할 수 있도록 분석한다.

6.1 테이블과 데이터 모델 클래스의 관계 일반화

일반적으로 데이터 모델 클래스는 테이블 한 행의 데이터에 대응되도록 설계된다. 테이블의 컬럼 이름과 데이터 모델 클래스의 변수 이름을 매핑 시키는 정보를 가지고 있는 경우 컬럼의 필드 타입을 이용하여 자바 변수의 타입을 설정할 수 있으므로 데이터 모델 클래스를 자동 생성하는 것이

가능하다. 단, 데이터베이스의 필드 타입을 다른 자바 타입으로 변환시켜 대응 시킬 필요가 있는 경우가 존재하므로 자바 변수의 타입을 직접 설정하는 것도 지원해야 한다. 데이터 모델 클래스가 다른 테이블의 컬럼에 대한 데이터를 가질 필요가 있는 경우 해당 테이블에 대한 데이터 모델 클래스에 대한 참조를 가지도록 구현함으로써 일반적인 데이터 모델 클래스 구현이 가능하다.

6.2 데이터베이스 트랜잭션 메쏘드 구현의 일반화

데이터베이스 트랜잭션 클래스가 가진 메쏘드는 그 기능에 따라서 조회, 추가, 제거, 변경으로 구분된다. 조회, 추가, 제거, 변경 메쏘드는 각각 SELECT, INSERT, DELETE, UPDATE SQL 쿼리와 대응되므로 이를 이용하여 각 메쏘드가 어떤 기능을 하는 메쏘드인지 정해지면 관련 SQL 쿼리를 생성할 수 있도록 일반화가 가능하다.

SQL 쿼리 생성시에는 컬럼에 대한 정보가 이용된다. 한 테이블에 주어진 컬럼의 이름은 SELECT, INSERT, DELETE, UPDATE 쿼리에 대하여 공유되며, 데이터 모델 클래스의 변수는 테이블의 각 컬럼과 대응 관계에 있도록 설계되므로 클래스의 변수를 각 쿼리를 생성하는데 대응시켜서 사용할 수 있다. 클래스와 테이블 간의 매핑 정보를 이용하면 클래스에 들어있는 값을 SQL 쿼리의 관련 컬럼에 설정하는데 이용 가능하다.

SELECT, UPDATE, DELETE SQL 쿼리는 공통적으로 WHERE 절을 가진다. 일반적인 데이터베이스 트랜잭션 클래스의 WHERE 절은 동적으로 생성되는 형태로 구성된다. 트랜잭션 수행 시 인자를 받아서 그것을 이용하여 트랜잭션 쿼리를 수행할 대상을 한정하는 방식으로 작동하기 때문이다. 이 부분을 일반화된 형태로 표시하기 위해선 WHERE 절의 내용을 인자를 받아서 동적으로 설정해줄 수 있도록 설계되어야 한다. ODMG 3.0의 OQL이나 EJB 2의 EJB QL의 경우 이러한 문제를 해결하기 위해 별도의 Escape 문자를 쿼리에 삽입하여 동적으로 값이 설정될 부분을 표시하는 방식을 이용하였다. [1]

6.3 비즈니스 로직 핸들링 메쏘드와 사용자 인터페이스 구현의 일반화

비즈니스 로직 핸들링 메쏘드와 사용자 인터페이스는 구현하는 전자 상거래 시스템의 형태에 따라서 매우 다양하게 나타날 수 있다. 따라서 이 부분을 완전히 일반화하려는 것은 의미 없는 작업이 될 것이다. 다만 개발의 효율성을 고려할 때 이들을 구현할 때 빈번히 출현하는 코드를 자동 생성하여 이용할 수 있도록 해주는 것은 유용한 기능이 될 수 있다.

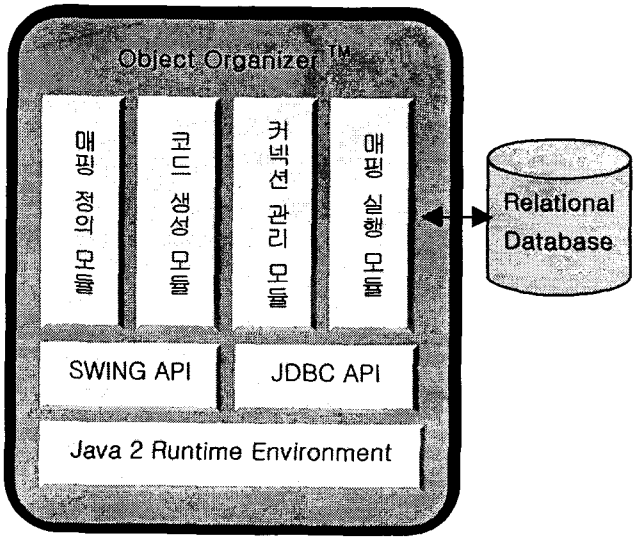
7. O-R 매핑툴 구현

새로운 O-R 매핑툴 Object Organizer ™는 설정 클래스, 실행 클래스, GUI 클래스로 구성된다. 설정 클래스는 오브젝트와 데이터베이스 테이블 사이의 매핑 정보를 담고 있는 클래스로서 설정 정보를 저장하고 읽어 오는 기능을 가진다. 실행 클래스는 설정 클래스의 정보로 초기화 되며 매핑 정보를 적용하여 데이터베이스 트랜잭션을 수행하는 기능을 가진다. GUI 클래스는 설정 클래스의 내용을 편리하게 수정할 수 있는 인터페이스를 제공한다.

7.1 제품 아키텍처 구성도

Object Organizer ™는 Java 2 Runtime *Environment*에서 구동 되며 JDBC API와 SWING API를 이용하여 데이터베이스 트랜잭션과 사용자 인터페이스를 구현하였다. 매핑을 정의하고 코드를 자동 생성하는 모듈이 사용자 인터페이스를 구현하는데 적용되었으며, 커넥션을 관리하고 매핑을 실행하는 모듈을 이용하여 데이터베이스 트랜잭션을 수행하도록 설계되었다.

[그림 1]



7.2 컴포넌트 소개

매니저(Manager) 컴포넌트는 테이블과 오브젝트 간의 매핑 정보를 저장하는 역할을 한다, 테이블의 각 컬럼이 자바 클래스의 어떤 속성과 대응되는지 정의하며 각 속성의 자바 타입도 정의한다. 이러한 매핑 정보를 이용하여 데이터 모델 클래스를 자동 생성할 수 있으며 하나의 매니저(Manager) 컴포넌트는 여러 개의 액션(Action) 컴포넌트를 가지고 있다.

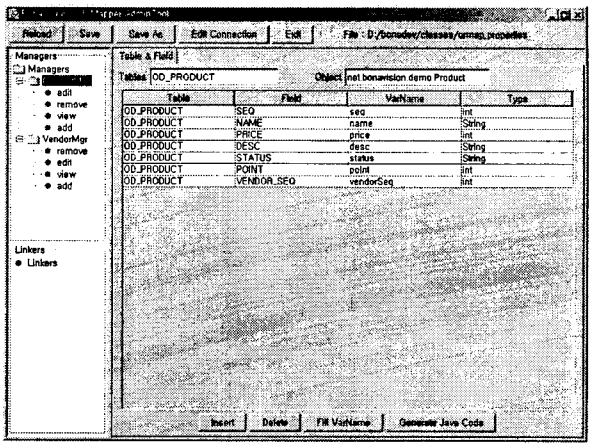
액션(Action) 컴포넌트는 자신이 속한 매니저(Manager) 컴포넌트에 대해 수행할 쿼리를 나타낸다. 액션(Action)에는 SELECT, INSERT, UPDATE, DELETE의 네 가지 타입이 존재한다. SELECT와 DELETE 타입의 액션(Action)은 레코드의 선택 조건을 입력 받을 수 있으며 이는 동적으로 조건 변수를 할당할 수 있도록 구현되었다. INSERT 타입의 액션(Action)은 자동 증가 컬럼을 설정할 수 있으며 자바 오브젝트를 받아서 레코드에 입력하는 쿼리를 수행한다. UPDATE 타입의 액션(Action)은 레코드의 선택 조건을 입력 받을 수 있으며 동적으로 조건 변수를 할당할 수 있도록 구현되었다. 또한 UPDATE 타입의 액션(Action)은 변경할 자바 오브젝트를 받으며 이 내용을 이용하여 선택된 레코드의 내용을 변경하게 되며 변경 시 값으로 사용될 자바 오브젝트의 속성 이름을 입력 받는다.

링커(Linker) 컴포넌트는 매니저(Manager)가 자바 오브젝트(A) 생성시 연결(Link) 관계가 정의되어 있는 다른 매니저(Manager)의 자바 오브젝트(B)를 자바 오브젝트(A)의 참조 변수로 설정해주는 기능을 한다. 링커(Linker)는 SELECT 타입의 액션(Action)을 수행 하는 경우에 작동하며 두 매니저(Manager)를 연결 시 이용될 대응 관계, 액션(Action), 동적 변수, 자바 변수를 설정함으로써 생성된다. 링커(Linker)는 불필요한 오브젝트 정보를 가져오는 단점이 있을 수 있기 때문에 액션(Action)에서 사용 여부를 선택할 수 있도록 구현하였다.

7.3 매핑 설정 인터페이스

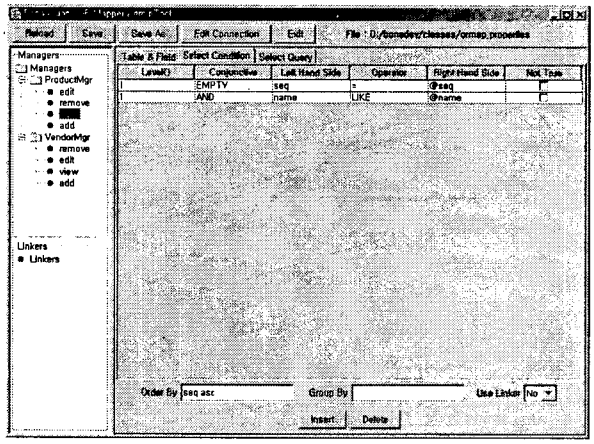
매니저(Manager)는 아래 그림 2와 같은 사용자 인터페이스를 이용해 정의할 수 있다. 입력하는 과정에서 주어진 테이블에 존재하는 컬럼의 이름이 자동 선택되고 자바 변수 이름을 컬럼 이름을 이용하여 자동 생성해주므로 손쉽게 입력이 가능하다. 컬럼에 대응되는 자바 변수의 이름과 타입을 선택한 후 데이터 모델 클래스 코드를 자동 생성할 수 있다.

[그림 2]



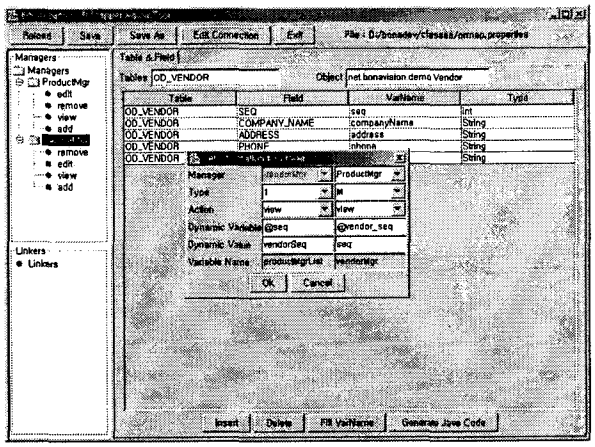
액션(Action)은 아래 그림 3과 같은 사용자 인터페이스를 이용해 정의할 수 있다. 액션(Action)의 타입을 선택하면 해당 액션에서 선택할 수 있는 선택 사항이 표시된 패널이 나타나며 이것을 통해 선택 조건이나 자동 증가 옵션을 설정할 수 있다.

[그림 3]



링커(Linker)는 아래 그림 4와 같은 사용자 인터페이스를 이용해 정의할 수 있다. 연결 시킬 두 매니저(Manager)를 선택하고 타입, 액션(Action), 동적 변수, 자바 변수를 선택하면 새로운 링커(Linker)가 생성된다.

[그림 4]



7.4 실행 환경

Object Organizer ™는 JDK 1.2 이상의 Java Runtime Environment 하에서 실행될 수 있으며 64MB 이상의 메모리와 20M 이상의 하드디스크 공간을 필요로 한다.

8. 개발 효율성 비교

O-R 매핑툴의 도입 효과를 측정하는 기준은 구현 시간 또는 구현 코드의 양이 고려된다. 구현 시간의 경우 프로그래머의 상황에 따라서 객관적인 측정이 어려운 부분이므로 본 논문에서는 손수 구현해야 하는 코드의 양을 이용하여 도입 효과를 측정하고자 한다.

8.1 산출물 요약

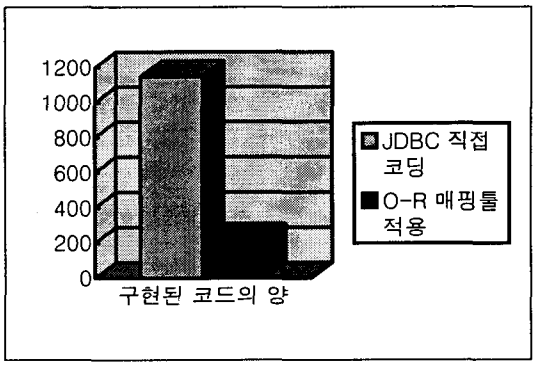
전자 카탈로그 컴포넌트를 구현하기 위해서 일반적인 웹 애플리케이션 설계 방식을 적용 하였으며 데이터베이스 트랜잭션 클래스는 Object Organizer ™를 이용하여 매핑 정보를 설정하는 것만으로 구현하였다. 이 구현 결과 산출물은 다음과 같다.

(1) 데이터 모델 클래스 : Product, Category
(2) 비즈니스 로직 핸들링 클래스 : Catalog
(3) 사용자 인터페이스 : JSP 페이지

8.2 코드의 구현 양의 비교

실제 구현에 필요한 코드의 양을 줄 수를 통해 비교하였다. O-R 매핑툴을 이용하는 경우 자동 생성되는 코드를 이용할 수 있으며 데이터베이스 트랜잭션 클래스를 작성하지 않아도 되기 때문에 20%의 코드를 이용하여 동일한 기능을 하는 애플리케이션을 작성할 수 있다. <표 1>

<표 1>

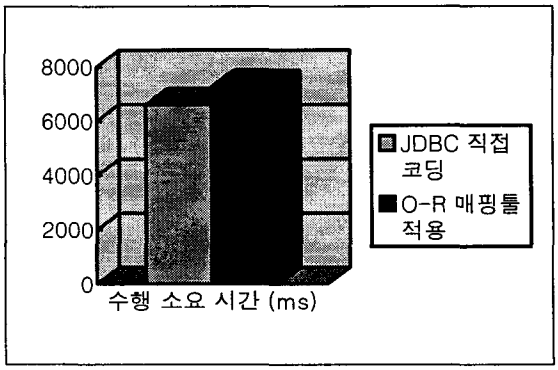


클래스 이름	JDBC 직접 코딩 시	O-R 매핑툴 적용 시
Product	207 줄	(자동 생성)
Category	63 줄	(자동 생성)
ProductMgr	413 줄	불필요
CategoryMgr	254 줄	불필요
Catalog	211 줄	223 줄
합계	1148 줄	223 줄

8.3 수행 성능 비교

실행 성능을 측정하는 것은 수행하는 쿼리와 데이터베이스에 저장되어 있는 레코드의 수에 따라서 매우 상이하게 나타난다. 본 논문에서는 O-R 매핑툴을 적용함으로써 매우 현격한 성능 저하 현상이 나타나는지 여부에 관심을 갖는다. 이를 평가하기 위해서 12000 개의 레코드가 저장되어 있는 상품에서 브라우징을 수행하는 시간을 순수 JDBC를 이용하여 구현한 경우와 O-R 매핑툴을 적용한 경우를 비교하였다. 이 실험을 통해서 평균적으로 JDBC에 비해 91%의 성능을 보여주었으며 <표 2> O-R 매핑툴을 실제 서비스에 적용하기에 충분한 성능을 가지고 있는 것으로 볼 수 있다.

<표 2>



테스트 회수	JDBC 직접 코딩 시	O-R 매핑툴 적용 시
1회	6742 ms	7554 ms
2회	6543 ms	7112 ms
3회	6740 ms	7229 ms
4회	6412 ms	7167 ms
평균 시간	6609 ms	7265 ms

9. 결론

본 논문에서는 전자 상거래 시스템을 위한 범용적 O-R 매핑툴인 Object Organizer ™ 을 설계하고 구현하였다. Object Organizer ™를 설계하기 위해서 범용성을 요구하는 적절한 타겟 애플리케이션 도메인을 모색하였으며 그 결과 전자 카탈로그가 선정되었다. 전자 카탈로그는 설계되는 O-R 매핑툴이 범용성을 확보할 수 있도록 요구 사항을 제공해 주었을 뿐 아니라 O-R 매핑툴의 적용 효과를 실험해 볼 수 있는 테스트 베드의 역할을 해주었다.

본 논문에서는 O-R 매핑툴의 설계 과정에서 일반적인 개발 프로세스를 분석하고 애플리케이션 개발 과정을 일반화 하는 접근 방법을 취하였다. 이러한 접근 방식은 개발 효율성을 향상시킬 수 있는 기법을 O-R 매핑툴 설계에 효과적으로 반영할 수 있게 해주었다는 점에서 의의가 있다. 또한 설계 과정에서 기존 O-R 매핑툴이 가지고 있었던 사용자 편의성과 수행 성능 면의 문제점을 개선하기 위하여 단순화된 API, 통합 설정 파일, 사용자 인터페이스를 보완하였다. 마지막으로 Object Organizer ™를 이용해 전자 카탈로그 애플리케이션을 구현하였으며 이를 기존 개발 방식의 구현 내용과 비교 분석하였다. 이를 통해 Object Organizer ™가 높은 개발 효율성과 뛰어난 수행 성능을 가진 O-R 매핑툴임을 확인할 수 있었다.

10. 참고 문헌

[1] R. G. G. Cattell, Douglas K. Barry, Mark Berler, Jeff Eastman, David Jordan, Craig Russell, Olaf Schadow, Torsten Stanienda, and Fernando Velez. The Object Data Standard : ODMG 3.0, Morgan Kaufman, 2000

[2] G.M. Bierman. Using XML as an Object Interchange Format, Department of Computer Science University of Warwick, 2000

[3] Craig Russell. Java Data Object Architecture, http://access1.sun.com/jdo/, 2002

[4] Robin M. Ross. Java Data Object Write Once, Persist Everywhere, http://www.ogilviepartners.com/, 2002

[5] Linda G. DeMichiel, L. .mit Yal.inalp, Sanjeev Krishnan. Enterprise Java Beans ™ Specification, P125-242, http://java.sun.com/, 2001