한국산업정보학회논문지 제6권 제2호 (2001. 6)

# Web환경에서 멀티미디어 기반 문제은행 시스템의 구현†

# (Implementation of a Multimedia based ExamBank System in Web Environments)

남 인 길* 정 소 연*
(In-gil Nam So-yean Jung)

**요 약** 본 논문에서는 웹 상에서 멀티미디어를 기반으로 한 문제은행 시스템을 제안하였다. 제안된 문제 은행 시스템은 객체-관계(Object-Relation)형 모델로 데이터베이스를 설계하고, 웹 환경에서 다수 클라이언트에 대한 무결성을 위해 독립적 실행이 되도록 Java 언어로 응용프로그램을 구현하였다. 문제 엔티티들은 객체로 정의하고, 이들간의 관계를 사용자 정의유형과 타입으로 설계 구현하였으며, DBMS의 스키마 객체와 JAVA Class를 매핑하여 DBMS와 어플리케이션 서버간에 체계화된 동작으로 객체를 전달할 수 있도록 하였다.

**Abstract** In this paper, we proposed multimedia based ExamBank system in web environments. In the proposed system, the database was designed based on the object-relation model and the application program was implemented with Java such that independent execution would be possible to guarantee no fault for multi-client in Web environments. We defined the Exam entities as objects, and implemented those inter-relationships as user definition and type. In addition, by mapping the schema object of DBMS and JAVA class, it becomes to possible transferring the object systematically between DBMS and application server.

## 1. 서 론

멀티미디어 데이터는 텍스트, 이미지, 오디오, 비디오, 그래픽 등과 같이 복잡한 구조와 정형 및 비정형 데이터가 다양한 관계로 맺어진 복합 객체적 성격을 가진다. 또한 멀티미디어 데이터는 물리적으로 대용량 데이터라는 특성을 가진다. 이런 데이터를 지원하기 위해서는 저장 시스템에 데이터 크기 제한이 없어야하며 데이터의 효율적인 연산을 제공할 수 있어야 한다. 멀티미디어 기반이라는 것은 기본적인 데이터가 텍스트 뿐만 아니라 이미지, 오디오, 비디오, 그래픽, 동영상과 같은 여러 형태의 매체를 포함하는 것을 말한다.

따라서 문제은행 시스템에서 하나의 문제를 구성하는 복잡한 엔티티들을 명확하게 표현하기 위해서는 새로운 종류의 데이터형과 이러한 데이터 형들간의 관계가 필요하다. 기존 관계형 데이터 베이스는 정형화된 레코드형식에 적합하도록 발달해 왔기 때문에 엔티티들의 집합을 표현할 수 있는 직접적인 기능을 제공하지 않는다. 그리고 객체 중심 데이터 베이스는 현실세계의 엔티티만을 인식하며 관계성은 단순히 속성으로만 취급하고 모든 관계성은 두 엔티티 간의 참조관계로 표현된다[1,2,3].

그러나 멀티미디어를 기반으로 하는 문제은행 시스템의 데이터베이스에서는 두 엔티티 이상 여러 엔티티 사이에 관계성이 존재한다. 이러한 특징은 관계형 데이터베이스에서 다양하고 복잡한 설계를 하기에는 적

* 대구대학교 정보통신공학부

† 이 논문은 2000년도 대구대학교 학술 연구비 지원에 의한 논문임

합하지 않다. 또한 객체 중심 데이터베이스시스템들은 데이터 조작기능을 절차적인 객체중심 또는 프로그래밍 언어의 개념에 의존함으로써 데이터베이스만이 가지고 있는 데이터조작의 특수성을 만족시켜 주지 못한다[8,9].

따라서 본 논문에서는 멀티미디어를 기반으로 한 문제은행 데이터베이스를 설계하는데 있어 객체-관계모델을 이용하여 표현한다. 문제를 구성하는 복잡한 엔티티들의 데이터형과 데이터형의 집합은 객체-관계모델에서 객체 처리방법으로 정의하고 엔티티들 간에 발생하는 복잡한 관계들은 객체-관계 모델에서 관계 처리방법으로서 표현하였다. 또한 웹과 데이터베이스 시스템을 연동하기 위한 인터페이스로 자바언어를 사용하였다. 어플리케이션의 구현에 있어 데이터베이스의 객체와 어플리케이션간의 인터페이스 구현을 위해 Java class를 사용하였고, 웹 브라우저에서 제공될 다양한 문제를 등록하기 위해서 Sun사의 JMF(Java Media Framework) API(Applic-ation Programing Interface)를 사용하였다. 2장에서는 멀티미디어데이터와 데이터베이스 모델연구로 데이터베이스모델별로 멀티미디어 데이터 표현의 문제점을 살펴보고, 제시된 객체-관계모델로서의 해결방안을 모색하며, 오라클의 객체-관계기술의 개념과 Java API 기술에 대해 살펴보았다. 3장에서는 객체-관계 데이터베이스의 특징에 기반하여 데이터베이스를 설계하고 4장에서는 3장에서 제시된 모델로 문제은행시스템을 구현하였다. 끝으로 5장에서는 결론 및 향후 연구과제에 대해서 연구하였다.

## 2. 멀티미디어데이터와 데이터베이스 모델연구

본 절에서는 문제은행시스템의 멀티미디어 데이터와 복잡한 관계의 표현을 위해 여러 가지 데이터베 이스 모델의 경우를 고려하고, 객체-관계형 모델에 서의 멀티미디어 데이터와 관계의 표현방법을 설명 한다. 그리고, 문제 은행시스템의 관계성을 표현할 Oracle에서 제공하는 객체-관계기술과, 웹에서의 독 립 수행과 멀티미디어 데이터의 표현을 위해 Java API를 살펴본다.

### 2.1 데이터 모델의 특성

#### 2.1.1 관계형 모델

멀티미디어 기반의 문제은행의 경우 다양한 형태 의 데이터형 즉, 숫자, 문자, 문자열은 물론이고 이 미지, 오디오, 텍스트 등을 포함한다. 이러한 다차 원의 데이터 타입은 테이블의 칼럼에 사용할 수가 없다. 또한 고정 타입과 함수만을 지원함으로 사용자 정의타입으로의 확장이 어렵다. 문제에 대한 정답의 경우도 단답형, 서술형 등과 같이 다양하게 나타나고 이를 표현하는 데이터형 역시 다양하다. 이와 같이 관계형 데이터베이스는 사용자 정의타입의 모델링이나 질의처리 부분 등의 측면에서 멀티미디어데이터의 처리가 자연스럽지 못하며, 비효율적이다. 따라서 문제은행 데이터베이스에서 문제를 이루는 엔티티들을 관계형 모델로 표현한다는 것은 적절하지 않다[11].

#### 2.1.2 객체중심 모델

객체 지향개념은 추상화, 정보의 은닉, 캡슐화, 상속성 등의 개념을 이용한 모델링 능력 때문에 멀티미디어를 위한 데이터 모델링 방법으로서 적합한 것으로 여겨지고 있다[12.13].

그러나 객체중심 데이터베이스는 관계성을 직접적으로 표현할 수 있는 구문과 시맨틱(semantic)을 제공 하지 않는다. 문제은행 데이터베이스에서는 문제의 마스터 클래스와 문제의 항목이나 해설 클래스는 상호 참조되어야 하는데 일반적인 객체 중심 모델은 두 개의 객체가 서로 참조 해야된다는 제약 조건을 표현하지 못하므로 시맨틱 정보는 상실된다 [3,12]. 특히 관계성 자체가 애트리뷰트를 가지는 경우는 이 애트리뷰트가 어느 한 엔티티에만 속한 것이 아니므로 어느 한 쪽 클래스에만 표현하기가 곤란하게된다. 따라서 특정 애트리뷰트의 참조를 표현하는데 있어 기존의 객체중심 모델은 적절하지 못하다.

#### 2.1.3. 객체-관계 모델제안

관계모델과 객체모델의 문제점들을 객체 관계형 모델로서 해결 방안을 모색한다.

객체-관계 데이터베이스는 기본적으로 관계형 데이터 베이스의 SQL92 표준 모델에 기반하여 관계형 데이터베이스가 가지는 장점을 최대로 지니면서 객체지향 개념을 지원하기 위해 확장되어진 형태이다. 이 모델은 멀티미디어 데이터 처리에 필요한 기능을 위해 객체지향 개념을 수용하였다. 기존 관계형모델에 객체지향 개념을 도입하여 발생하는 다양한 데이터 타입의

집합 문제는 사용자 정의형 객체 집합타 입으로 정의할 수 있다. 예를 들어 예문의 항목 데이터가 멀티미디어 데이터이면 멀티미디어 집합타입의 한 아이템으로 표현이 되고 이러한 타입은 문제의 마스터 객체 타입에 포함되어질 수 있다. 즉, 문제를 구성하는 각 엔티티들의 애트리뷰트들은 텍스트나 멀티미디어 데이터들의 집합으로 표현하고 마스터 객체 타입에서는 이러한 집합 데이터가 문제의 어느 구성요소에 속해져 있는지에 대한 정보를 가지면 된다. 또한 객체를 처리하는 과정에서 발생하는 관계의 표현 문제는 관계 처리방법으로서 해결이 될 수 있다. 즉, 문제의 마스터 객체타입은 문제의 구성 엔티티 객체의 식별자인 OID(Object IDentifier)를 참조하고 문제의 구성 엔티티의 한 애트리뷰트는 마스터 객체의 주키 (Primary Key)를 참조하면 된다.

## 2.2 설계 및 구현 요소기술

### 2.2.1 Oracle의 객체-관계 기술

Oracle 8i는 자체적으로 객체타입(object type)이라고 불리는 사용자 정의형을 지원한다. 이것은 C++ 나 Java에서 지원하는 클래스 메카니즘과 유사한 스키마 객체이다. 그리고 1 : N 관계를 모델링 하기 위해 Varrys와 Nested Tables의 두 집합 데이터 타입을 지원한다. Varrys는 사용자가 설정한 고정적인 수만큼의 앨리먼트(element)를 가지며 이 앨리먼트의 순서는 정의된다. Nested Tables은 어떤 수의 앨 리먼트를 가질 수 있고 정규 테이블 같이 SELECT, INSERT, DELETE등의 연산을 할 수 있다. 반면 앨리먼트의 순서는 정의되지 않는다 [6]. 또한 각각 언어들의 상속에 대한 의미가 다르므로 직접적으로 제공되지 않고 하나의 상속과 다중 상속에 대한 혼합을 사용한다. 서버측의 메소드 상속은 오라클 8i Java VM(Virtual Machine)을 통해 Java로 제공된다. 그러나 타입이나 그것의 서브 타입에 대한 저장이나 쿼리 인스턴스에 대한 SQL 차원의 상속은 지 원하지 않는다. 내장데이터 타입인 REF를 이용하여 그것이 참조하는 객체에 접근하여 갱신, 삭제 등 연산을 수행할 수 있고, 같은 타입의 다른 객체로 REF를 변경할 수 있으며, Null Value를 할당할 수 도 있다. 즉, REF Type을 통해 객체 사이를 쉽게 항해 할 수 있는 메카니즘을 제공한다 [6].

### 2.2.2 Java API

Java의 사용은 웹 클라이언트 측이나 서버 측의 실행에 있어 보다 유연하고 동적인 시스템을 설계 할 수 있다. 또한 플랫폼 독립적인 수행을 가능 하게하며 다중 사용자를 위한 세션 관리 및 쓰레드 관리가 용이하다.

#### 2.2.2.1 Java Servlet

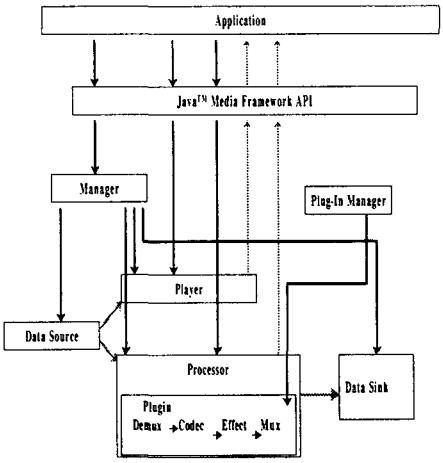
자바 서블렛은 일반적인 서버 익스텐션 (서버로서의 기능을 확장하기 위해 동적으로 적재될 수 있는 자바 클래스)이다. 서버 측 자바개발의 주요 컴포넌 트인 서블렛은 애플릿과는 달리 웹브라우저 안에서의 자바에 대한 지원을 필요로 하지 않는다. 또한 서블렛은 지정된 서버머신에서만 동작하며, Java의 오류부분이나 일관성이 없는 부분은 사용하지 않는다. 따라서 이식성이나 안정성, 효율성, 확장성 등을 가진다. 서블렛은 라이프사이클 동안 데이터베이스 연결이 오픈 되어진 상태로 유지되므로, 시간이 줄어든다. 이것은 다수의 접근과 다양한 플랫폼에서의 일관성과 안정성을 기하고, 또한 3 계층 모델에서 미들티어에 위치하여 서버 측 데이터 베이스 접근에 있어 연결을 단순화하고 안전하게 하는데 도움을 준다. 서블렛을 구동하기 위한 엔진에는 부가형 엔진 중 Java Apache의 JServ를 사용하였다[7].

#### 2.2.2.2 JMF(Java Media Frame Work)

1996 JavaOne$^{SM}$ 컨퍼런스에서 JMF는 제안되었다. 이것은 오디오나 비디오 또는 시간에 의존하는 다른 데이터들의 동기화를 맞추는 통합구조를 간단하게 만들어 놓고 이것을 재생하는 API이다.

압축해제와 랜더링이 빈번하게 일나는 멀티미디어 특성상 JVM으로는 높은 성능을 내기가 힘들다.

JMF는 상이한 플랫폼이라도 일관된 방식으로 멀티미디어 파일을 재생한다. 그러므로 모든 플랫폼에서 동작하는 멀티미디어 타입을 만들고 그것을 재생할 수 있다. <그림 1>은 JMF와 어플리케이션 사이의 상호작용을 나타낸 것이다[7].




<그림 1> JMF와 어플리케이션 기반의 자바 기술 상호작용

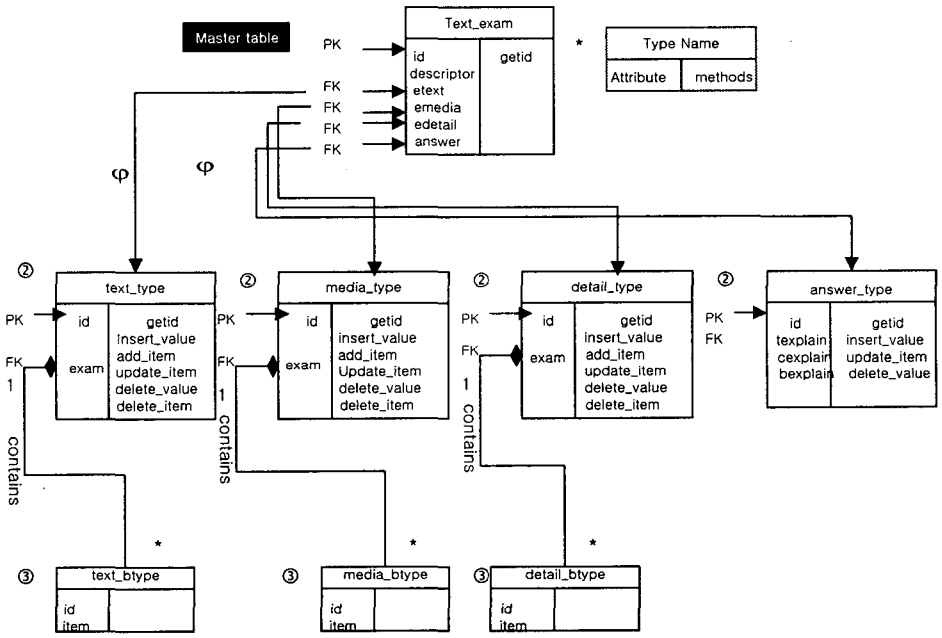
### 3.1. 객체-관계형 모델 구성도

멀티미디어 데이터를 기반으로 한 문제은행 시스템은 <그림 2>와 같은 구조를 가진다. 각 데이터의 특성에 따라 테이블이 각각 구성되어지며, 문제의 마스터 객체타입은 문제의 구성 엔티티 객체의 식별자인 OID를 이용하여 ①과 같이 참조하고, 문제를 구성하는 각 엔티티들의 애 트리뷰트는 ②와 같이 구성되어 마스터 객체의 주키를 참조하여 마 스터테이블을 중심으로 관계를 맺는다. ②의 exam 필드는 ③에서 선언 된 각 타입이 Nested table로 구성되어졌다.

### 3.2. 데이터 타입 정의

문제 마스터 객체는 간단한 텍스트의 집합을 가지는 Detail_Type, 대용량의 텍스트 집합을 가지는 Text_Type, 미디어 데이터의 집합을 가지는 Media_Type, 문제 해설을 가지는 Answer_ Type을 참조한다. <그림 3>이 그것을 나타내고 있다.

## 3. 문제은행 시스템 설계




<그림 2> 데이터베이스 모델

```
CREATE TYPE exam_type AS object
(id                 NUMBER(9),
 description        VARCHAR2(512),
 etext      REF     text_type,
 emedia     REF     media_type,
 edetail    REF     detail_type,
 answer     REF     answer_type
 MAP MEMBER FUNCTION getid RETURN NUMBER);
);
```

<그림 3> 문제 및 해설관련 데이터 Type 정의

마스터 테이블에서 멀티미디어 데이터를 표현할 객체 필드는 media_type을 참조하게 된다. 이 타입은 마스터테이블과 관계를 맺어주는 id와 exam이 라는 필드로 구성되어지며, exam은 멀티미디어 데이터의 집합을 가진다. 그리고 이것을 저장할 자료형으로 BLOB을 사용하였다.<그림 4>가 타입정의 과정을 보여주고 있다.

```
CREATE TYPE media_btype AS object
(id         NUMBER,
 item       BLOB);

CREATE TYPE media_nbtype AS TABLE OF media_btype;

CREATE TYPE media_type AS object
(id         NUMBER,
 exam       media_nbtype);
```

<그림 4> 미디어 데이터를 표현할 Type 정의

### 3.3. 객체테이블 정의

사용자 정의형 객체를 저장하기 위해서 객체테이블을 정의하여야 한다. 마스터 테이블에는 id를 제외한 모든 필드에 외래 키를 선언하여 각 데이터형 들의 선언되어진 테이블을 참조하도록 한다.<그림 5>은 Media 타입 객체 테이블을 정의한 것이다.

```
CREATE  TABLE media_objtab OF media_type
(  id  PRIMARY KEY )
  Object id PRIMARY KEY
NESTED TABLE exam STORE AS media_ntab
(
   ( PRIMARY KEY(id)  )
   ORAGNIZATION  INDEX COMPRESS
);
```

<그림 5> Media Type 테이블

각 데이터형대로 테이블이 구성되어진 다음 마스터 테이블과 관계성을 맺기 위해 테이블에 외래 키를 설정한다. <그림 6>은 만들어진 Exam 테이블의 id를 참조하기 위하여 외래키(Foreign Key)를 선언하는 부분이며 그 중의 한 예이다.

```
ALTER TABLE media_objtab ADD(FOREIGN KEY(id)
REFERENCES exam(id) ON DELETE CASCADE);
```

<그림 6> 기존 객체 테이블의 FK 추가

### 3.4. 메소드 설계

일반적으로 메소드는 어떠한 행위를 생성하기 위해 어플리케이션에서 호출하는 함수나 프로시저를 말한다. 제안되어진 문제은행 시스템에서 메소드는MEMBER나 STATIC의 키워드를 사용하여 하나 의 객체타입 안에서 서브 프로그램으로 선언되며 MEMBER 메소드는 인스턴스(instance_expression. method()) 형식으로 호출되어지고, STATIC 메소드는 객체 타입의(object_type_name.method()) 형식으로 호출되어진다. <그림 7>은 PL/SQL로 구현되어진 Text_type의 메소드의 일부분이다.

```
CREATE OR REPLACE TYPE BODY text_type AS
STATIC PROCEDURE insert_value(pid IN NUMBER, ptntype IN text_nbtype) IS
BEGIN
 INSERT INTO text_objtab VALUES(pid, text_nbtype());
 FOR i IN 1..ptntype.COUNT LOOP
            INSERT INTO TABLE(SELECT t.exam FROM text_objtab t WHERE t.id=pid)
            VALUES(ptntype(i).id, ptntype(i).item);
 END LOOP;
 UPDATE exam_objtab SET etext=(SELECT REF(t) FROM text_objtab t WHERE t.id=pid)
 WHERE id=pid;
 END;
```

<그림 7> 메소드 설계

## 4. 문제은행 시스템의 구현

본 절에서는 3절에서 제안 설계한 웹상의 멀티 미디어 기반 문제은행시스템을 실질적으로 구현한 것에 대해 서술한다.

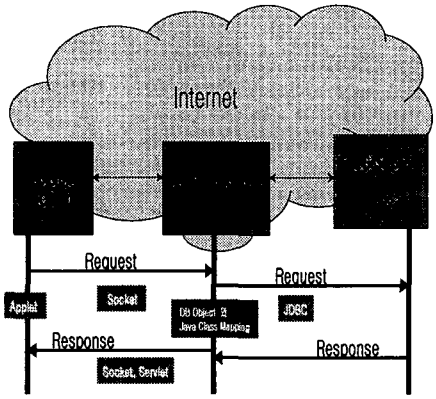
### 4.1. 개발 환경

멀티미디어 데이터를 지원하는 데이터 베이스는 대개 3단계 구조로 제안되고 있다[15,16].

문제은행 시스템의 미들 티어에는 어플리케이션서버로 Oracle Application Server1.01을 사용하였고, 운영체제는 Solaris7을 사용하였다. 데이터베이스서버의 DBMS는 ORACLE 8.1.7을 사용하였고, 운영 체제는 MS Windows NT Server4.0을 사용하였다. 클라이언트의 운영체제로는 Windows98을 사용하였다. 개발도구로는 Oracle JDeveloper 3.0과 JDK1.3을 사용하였고, 멀티미디어 데이터의 입·출력을 위해 JMF2.0을 사용하였다.

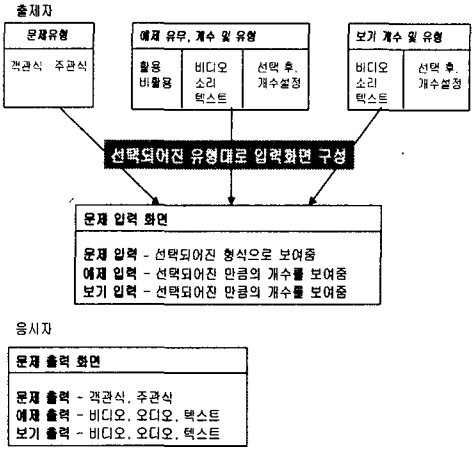
### 4.2. 시스템 구성도

시스템의 전체 구조는 물리적으로 사용자에게 문제 출제 화면과 결과를 제공하는 웹 클라이언트와 어플리케이션서버, 그리고 데이터베이스서버의 세계층으로 구성되어 있다. 웹 환경에서 다수 클라이언트에 대한 무결성의 독립적 실행이 되도록 Java언어로 구현되었고, DBMS의 스키마 객체와 Java Class를 Mapping하여 DBMS와 어플리케이션 서버 간에 체계화된 동작으로 객체를 전달할 수 있도록 하였다. 클라이언트는 소켓을 통해 어플리케이션서버에 접속을 하고, 접속 시 서버 쪽에서는 서블렛 및 소켓 프로그램으로 데이터를 받아들이고 데이터 베이스 서버로 접속을 하도록 구현하였으며, 구성도는 <그림 9>과 같이 나타낼 수 있다.




<그림 9> 시스템 구성도

### 4.3. 화면 설계




<그림 10> 화면 설계

<그림 10>은 출제자 측면과 응시자 측면 두 가지로 구성되어졌으며, 출제자가 문제의 유형 및 예제, 보기 등의 종류와 개수를 선택하면 문제 입력화면에서는 출제자가 설정한 대로 화면이 구성된다. 응시자는 출제자에 의해 만들어진 화면을 보게 된다.

### 4.4. 자바 클래스 구현

#### 4.4.1 클라이언트 시스템

문제의 출제시 일반적인 텍스트 데이터나 기존 에 작성된 이미지, 오디오, 비디오, 그래픽, 텍스트 파일은 웹 브라우저를 통해서 어플리케이션 서버로 전송되고 실시간 음성녹음이나 비디오 캡쳐는 JMF API를 이용한 애플릿에서 이루어지며 여기서 생성된 데이터의 전송은 소켓을 이용하여 어플리케이션 서버에 전송된다. 문제를 보는 사용자는 텍스트, 이미지 데이터는 웹브라우저로 전송되고 음성, 영상 데이터는 소켓을 통해 애플릿으로 전송된다. 또한 애플릿클래스에서는 실시간 비디오 및 음성을 녹화 및 녹음하고, 소켓을 통해서 어플리케이션 서버로 전송한다. <그림 11>은 애플릿 코드의 일부를 보여 준다.

```
{오디오캡쳐 소스}
 현재 이용가능한 모든 오디오 비디오 캡쳐 드라이버의 리스트를 얻어온다.
VectorvectorDevices=CaptureDeviceManagergetDevicesList(null);
CaptureDeviceInfo CaptureDeviceInfo;

 얻어진 디바이스 드라이버를 이용해서 데이터 소스를 만들고,
Datasourcedatasource=JMFUtils.createCaptureDataSource(null,null,   audioDeviceName)

 얻어진 데이터소스로부터 프로세스를 생성한다.
Processor=manager.createProcessor dataSource);

 프로세서의 출력되는 데이터의 포맷을 mp3즉, MPEG_AUDIO 로 저장한다.
Processor.setContentDescriptor(new FileTypeDescriptor(FileTypeDe   scriptor.MPEG.AUDIO))

 데이터의 저장을 위해서 DataSink 를 만드는 과정이다.
MediaLocator dest = new MediaLocator(destURL);
Try {sink = Manager.createDataSink(processor.getDataoutput(), de   st);



{Client socket 소스 }

 지정된 호스트, 주소, 포트로 소켓을 생성한다.
Socket socket = null;
try {
socket = new socket(host port);

 input, output.stream 을 사용하여 파일의 내용을 buffer 에 저장 서버소켓으로 전송
FileInputStream fi = new FileInputStream(   "33.mp3");
BufferedInputStream b is = new BufferedInputStream(fi);
BufferedOutputStream out = new BufferedOutputStream(socket.getO   utputStream());
~
~
while(b is.read(buffer)!=-1)
 {
out.write(buffer);
}
```

<그림 11> 애플릿 자바소스코드 예제

### 4.4.2 어플리케이션 서버 시스템

어플리케이션 서버란 공유할 수 있는 비즈니스(business logic)을 위한 미들티어의 시스템 소프트웨어를 말한다. 제안되어진 시스템에서는 클라이언트에서 전송된 Data는 서블렛이나 소켓 서버에서 객체 포맷으로 변환 후 JDBC를 통해 데이터베이스 서버로 전송하고, 데이터베이스 서버는 물리적인 공간에 객체를 저장한다. 마찬가지로 클라이언트에서 요청이 오면 어플리케이션 서버는 데이터베이스 서버에 요청을 해서 객체를 리턴받고 이를 자바 클래스로 매핑한 후 클라이언트로 데이터를 전송한다.

#### 4.4.2.1 사용자정의형 DB 엔티티 클래스

데이터베이스 객체 타입, 참조 타입, 집합 타입 등 사용자 정의형 데이터베이스 엔티티와 자바 응용 프로그램간의 객체 매핑을 위한 클래스를 구현하여야 한다. 일반적으로 구현하는 방법에는 두 가지가 있 는데 오라클 8i의 CUSTOM DATUM, CUSTOM DATUM FACTORY 인터페이스나 Java의 SQL DATA 인터페이스를 이용하여 직접적으로 클래스 를 구현할 수 있고 오라클의 jpublisher 유틸리티를 이용하여 .JAVA나 .sqlJ 파일을 만들 수 있다. <그림 12>는 Text_type 객체 타입에 대한 Text_ type.sqlJ 파일의 일부분이다.

```
<생략>
import java.sql.SQLException;
import oracle.jdbc.driver.OracleConnection;
import oracle.jdbc.driver.OracleTypes;
<생략>
/* constructors */
Public TextType()
{
 _struct = new MutableStruct(new Object[2], _sqlType, _factory);
 try
 {
  _ctx = new_Ctx(DefaultContext.getDefaultContext());
 }
 catch (Exception e)
 {
  _ctx = null;
 }
}
/* accessor methods */
Public java.math.BigDecimal getId() throws SQLException
{ return (java.math.BigDecimal) _struct.getAttribute(0) ; }

Public void setId(java.math.BigDecimal id) throws SQLException
{ _struct.setAttribute(0, id) ; }

Public void insertValue (   /* member method */
  java.math.BigDecimal pid,
  TextNbtype ptntype)
throws SQLException
{
 #sql [_ctx] { CALL TEXT_TYPE.INSERT_VALUE (
    :pid,
    :ptntype)    };
}
```

<그림 12> 사용자 정의형 DB엔터티 클래스

#### 4.4.2.2 Data 연산 클래스

서블렛은 삽입, 갱신, 삭제 등 모든 객체의 연산을 포함하고 서버 소켓은 비디오나 음성데이터의 삽입 연산을 수행한다.

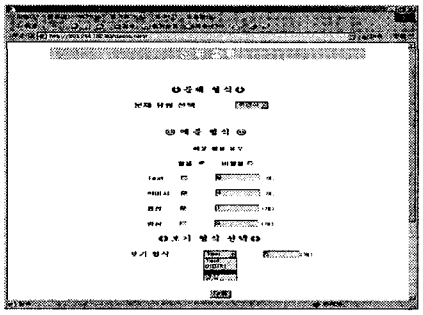
```
try {
 DriverManager.registerDriver(new oracle.jdbc.driver.OracleDri   ver());
 conn = DriveManager.getConnection(   "jdbc:oracle:1521:@exombank.ans.taegu.ac.kr   " user,Password);
 CallableStatement Cstmt = conn.prepareCalli   "begin Text_Type Insert_value(?,?);end;   ");
 Cstmt.SetInt(1,pid);
 Cstmt.SetObject(2,PtnType);
 Cstmt.execute();
}

catch(SQLException e;
{
 out.printLn(e.toString());
}
```

<그림 13> 서블렛 소스코드 예제

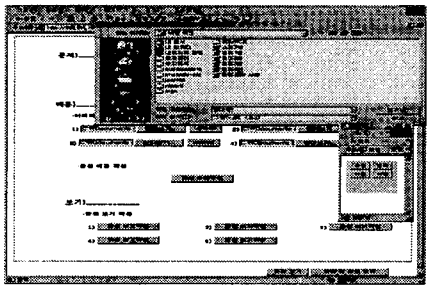
<그림13>은 서블렛에서 Text_type의 메소드를 호출하는 소스코드의 일부분이다.

### 4.5. 구현 결과

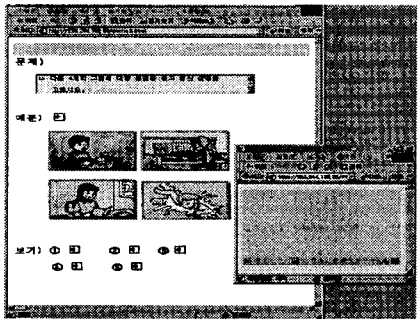
다음은 본 논문에서 구현한 시스템의 화면들이다 <그림 14에서 그림16 까지>. <그림 14> 초기화면에서 문제 출제자는 제출할 문제의 유형을 선택한 후 다음 버튼을 클릭하면 <그림 15>와 같이 선택된 형식에 따라 문제의 데이터를 입력할 수 있는 인터페이스를 제공한다.



<그림 14> 시스템 초기화면



<그림 15> 입력 인터페이스



<그림 16> 출제된 문제 보기 예제

<그림 15>에서 문제의 데이터를 입력한 후 제출한 문제를 볼 수도 있고, 정답 및 해설 입력 버튼을 클릭하면 해당하는 데이터를 입력할 수 있는 인터페이스를 제공한다. 그리고 출제된 문제를 보는 화면에서는 문제를 푼 후 정답 및 해설을 볼 수 있다

## 5. 결론 및 고찰

본 논문에서는 웹 상에서 멀티미디어 기반 문제은행 시스템을 객체 관계형 모델로 데이터베이스를 설계하고 Java언어로서 응용프로그램을 구현하였다.

문제은행 데이터베이스에서 발생하는 다양한 문제의 종류와 데이터 타입을 일관되게 표현하기 위해 객체로 정의하였고 문제의 마스터 객체 타입과 디테일 객체 타입의 관계를 표현하기 위해 객체-관계형 모델을 이용하였다. 이로서 관계형 모델로 설계될 때 발생하는 복잡성을 해결하고 객체중심 모델의 관계 표현의 부족함을 해결하였다. 또한 멤버 메소드를 두어 객체의 데이터를 명확하게 조작할 수 있었다. 향후 연구 방향은 문제의 구성 엔티티들이 수정 및 삭제되더라고 부가적인 연산이 발생하지 않는 방법을 제공해야 할 것이다. 또한 제출자의 정보를 저장하여 한번 제출한 문제의 타입은 계속 이용할 수 있는 기능과 문제의 검색에 있어 에이전트 기능을 추가하여 문제검색에 있어 효율성을 증가시켜야 할 것이다.

### 참 고 문 헌


[1] Cattell, R.G.G. and Rogers, T.R., "Combining Object-Oriented and Relational Models of Data, "Intl Workshop on Object-Oriented Database Systems, 1986.

[2] Rumbaugh. J., "Relations as Semantic Co nst-ructs in an Object-Oriented Language ", Proc. of ACM OOPSLA'87 Conference, Oct.1987.

[3] Tsichritzis, D.C. and Nierstrasz, O.M.,"Fitting Round Object Into Square Database," Lecture

Notes in Computer Science 322(ECOOP'88), Springer-Verlag,1988.

[4] 류시원, 조태경, 차광호, 정진완. "객체지향 기법을 이용한 멀티미디어데이터베이스의 모델링" 정보과학회 논문지(B) Vol.24, No.10, pp.1017-1030. 1997.

[5] 황수찬, 이석호. "객체 중심 데이터베이스에서 관계성의 관리 및 연산에 관한 연구"정보과학회논문지(B) Vol.17, No.3, pp.282-291. 1990.

[6] Oracle Technology Network http://technet.oracle.com

[7] The Source for JavaTm Technology http://java.sun.com/products/java-media/jmf/

[8] Banerjee, J., Kim, W., Kim, K., "Queries in Object-Oriented Database", Proc. 4th Intl. Conf. on Data Engineering, Feb. 1988.

[9] Copeland, G. and Maier, D., "Making Small talk a Database System",ACM SIGMOD,1984.

[10] O. Deux et al., "The O2 System", Communications of The ACM, Vol.34, No.10, 1991.

[11] 박동주, 김형주. "SOP OODBMS에서 멀티미디어 데이터 지원을 위한 클래스 라이브러리의 설계 및 구현"정보과학회 논문지(B) Vol.25, No.8, pp.1148-1158. 1998.

[12] Ullman, J.D., "Principles of Database and Knowledge-base Systems", Computer Science Press, Vol. 1, 1988.

[13] W. Klas, E.J.Neuhold and M. schrefl, "Using an Object-Oriented Approach to Model Multimedia Data," Computer Communications, Vol.13, No. 4, 1990.

[14] 박상원, 김형주 "관계형 데이터베이스의 객체지향적 인터페이스를 위한 게이트웨이의 설계 및 구현" 정보과학회 논문지(C) Vol.3, No.4, pp.333-342. 1997

[15] 이석호, Multimedia DBMS의 Logical Mo-del에 관한 연구, 최종보고서, 한국전자통신연구소, 1992.

[16] Klas, W., Neuhold, E.J., and Schrefl, M., "Using an object-oriented approach to model multimedia data," Computer Communication, Butterworth, Vol.13, No.4, pp.204-216, 1990.

[17] 나연묵, "멀티미디어 응용을 위한 객체-관계 모델에 관한 연구," 박사학위논문, 서울대학교, 1993.

[18] 복중효, 김광명, 이연식, "순수 자바를 이용한 웹 기반 멀티미디어 데이터 서비스 구조 설계," '2001 춘계 한국정보과학회 학술발표논문집(B) pp.232-234, 2001.

**남 인 길** (Nam In-Gil)
1978년 경북대학교 전자공학과 졸업 (공학사)
1981년 영남대학교 대학원 전자공학과 (계산기전공 공학석사)
1992년 경북대학교 대학원 전자공학과 (전산공학전공 공학박사)
1980년~1990년 경북산업대학교 전자계산학과 부교수
1980년~현재 대구대학교 정보통신공학부 교수
1996년~1997년 미국 루이지애나 주립대학 교환교수
관심분야 : 데이터베이스, GIS

**정 소 연** (Jung So-Yean)
1998년 대구대학교 전산정보학과 졸업 (공학사)
2000년~현재 대구대학교 컴퓨터정보공학과 (석사과정)
관심분야 : 웹 데이터베이스, 데이터웨어하우스