분산환경에서 컴포넌트 자동생성 시스템 설계 및 구현

천상호*, 권기현** , 최 형진***
(주) 오픈시스템서비스* · 동원대학 인터넷정보과** · 강원대학교 전자계산학과***

요 약

본 논문에서는 분산환경에서 MVC(Model View Controller) 모델을 기반으로 하는 Model 2 프레임웍에 입각하여 소프트웨어 컴포넌트와 관련된 요소를 자동 생성하여 웹 애플리케이션 구축을 지원하는 방법에 대해 제시한다. Model 2 프레임웍은 웹 애플리케이션에서 Model, View, Controller를 이용하여 MVC 구조로 작성하는 방법으로 기능을 캡슐화하여 변경에 대해 영향을 최소로 하는 구조이기 때문에 확장성, 유지보수성 면에서 좋은 방법으로 인식되고 있다. Model 2 프레임웍 하에서 소프트웨어 컴포넌트와 관련된 요소를 자동 생성하기 위해 MVC 모델, 컴포넌트, Model 2 프레임웍, 디자인 패턴 등에 대해 살펴보고 구현 환경에 적용하기 위한 개선된 Model 2 프레임웍을 제시하고 제시된 프레임웍에 기반하여 소프트웨어 컴포넌트와 관련된 요소를 자동 생성하는 시스템을 설계하고 구현한다.

Design and Implementation of Automated Component Generation System on Distributed Environment

Cheon Sang-Ho* · Kweon Ki-Hyeon** · Choi Hyung-Jin***

ABSTRACT

This paper presents the automated component generation system to support development of web application by the Model 2 framework on distributed environment. Model 2 framework is based on MVC(Model View Controller) model and this model capsulate the functionality of web application and have the benefits like extensibility, maintainability, resuability. In this paper, we propose a framework which is adapted in JSP environment and implement the automated component generation system. This system can efficiently utilized for web application development which require extensibility, maintainability, resuability as well as rapid web application development.

1. 서 론[1]

본 논문에서는 분산환경에서 Model 2 프레임웍에 기반하여 웹 애플리케이션을 구축하는데 요구되는 컴포넌트, 커스텀 태그 및 관련 프로그램 요소를 자동 생성하는 방법을 제시한다. Model 2 프레임웍은 웹 애플리케이션의 적응성, 확장성, 유지보수성을 극대화하는 방법으로 소프트웨어 공학의 MVC 모델에 기초를 두고 있다. MVC 모델은 통합된 시스템을 데이터인 모델, 사용자 인터페이스인 뷰 그리고 컨트롤러로 분해하여 시스템을 구현하는 것을 의미한다. 전통적으로 MVC 모델은 이벤트 위주의 윈도우 프로그램에서 적용되어 왔으나 분산 인터넷 환경에서 MVC 구조를 분산환경에 적합한 형태로 변환하여 사용될 수 있다. MVC 모델의 목표는 데이터와 사용자 인터페이스를 분리하여 시스템의 확장 및 유지 보수를 용이하게 하고 웹 페이지 작성자와 소프트웨어 개발자의 작업을 분리하여 대규모 프로젝트에서 작업을 효율적으로 분담하고 작업을 병행하여 수행할 수 있게 한다.

기존의 MVC 모델과 분산 기반의 MVC 모델의 차이점과 Model 2 프레임웍에 대해 설명하고 이를 기반으로 웹 애플리케이션에서 사용되는 서버 측 컴포넌트 및 클라이언트 측의 요소를 자동 생성하는 시스템을 설계하고 구현한다. 논문의 내용은 2장에서는 MVC 모델과 분산 디자인 패턴 그리고 Model 2 프레임웍에 관한 이론적인 내용을 설명한다. 3장에서는 환경 설정을 하고 환경에 적합한 구조로 Model 2 프레임웍을 개선한 내용에 대해 컴포넌트 자동생성 시스템을 설계하고, 4장에서는 설계한 내용을 기반으로 구현한 결과를 제시하며, 마지막으로 결론 및 향후 연구 방향을 제시한다.

※ 본 연구는 한국과학재단의 2001년도 이공계 교수 산업현장 지원사업에 의해 수행됨

2. 관련연구

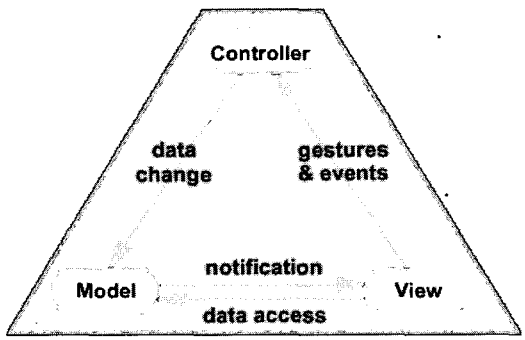
2.1 MVC 모델

2.1.1 Smalltalk MVC 모델

Smalltalk 언어는 1970년대 후반에 XEROX PARC에서 개발되었고 MVC 모델은 Smalltalk의 그래픽 객체를 관리하기 위해 도입되었다[6]. MVC의 구성 요소는 다음과 같다.

- Model: 그래픽 객체의 데이터를 관리하는 부분이다. 체크 박스의 상태나 텍스트필드의 내용이 이에 해당한다.
- View: 그래픽 객체의 외관을 처리하는 부분으로 한 종류의 데이터에 대해서도 다양한 뷰를 가지게 한다.
- Controller: 그래픽 객체에서 발생하는 이벤트의 처리를 처리하여 Model에 갱신하고 View에게 전달하는 역할을 수행한다.

MVC를 이용하면 데이터와 뷰(view)를 분리하여 데이터에 대한 효율적인 관리를 할 수 있는 장점이 있다.



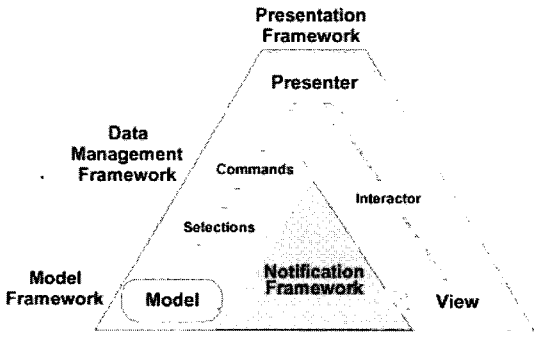
(그림 1) Smalltalk MVC 모델

2.1.2 분산 기반의 MVC 모델

Smalltalk의 MVC 모델을 확장하여 분산 기반에 적용하기 위해서는 Model, View, Controller의 분산 환경의 기능을 정의하는 것이 필요하다.

분산환경의 MVC에는 그래픽 사용자 인터페이스에 대한 부분이 없으므로, 이벤트 위주의 Controller도 의미가 없어지게 된다. 그러므로, 컨트롤러라는 용어 대신에 View와 Model을 중재하는 Presenter로서 역할을 하게 된다. 따라서, MVP(Model View Presenter) 모델에 대한 구성 요소는 다음과 같다.

- Model: 서버측의 데이터를 처리하는 부분으로 데이터베이스 및 데이터베이스에 대한 데이터를 관리하는 부분이 된다.
- View: 뷰는 처리 결과에 대한 출력을 처리하는 부분으로 텍스트 위주의 출력 또는 브라우저에 전달될 HTML이 뷰가 될 수 있다.
- Presenter: View와 Model 사이에서 데이터에 대한 관리 및 처리 결과 출력, 예외처리 및 세션관리 등을 위한 처리를 담당한다.



(그림 2) 분산 기반의 MVC 모델

2.2 3-계층 구조와 컴포넌트

확장된 MVC 구조는 컴포넌트를 사용하는 3계층 구조에 적당하다고 할 수 있다[1].

2.2.1 컴포넌트 기반 구조

컴포넌트 기반 애플리케이션으로 미들 계층을 설계하면 다음과 같은 이점을 얻을 수 있다[2].

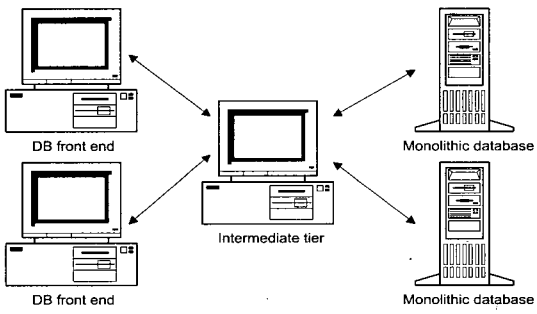
- 적은 스텝으로 대형 애플리케이션을 개발 가능
- 애플리케이션은 컴포넌트들을 재사용 가능
- 적은 스텝으로 대형 애플리케이션을 개발 가능
- 쉽고 안전하게 데이터와 기능에 접근 가능
- 애플리케이션은 기존 컴포넌트들과 결합 가능
- 컴포넌트 환경은 노화하지 않음

컴포넌트 유형에는 컴포넌트가 호출되면 어떤 인스턴스가 필요한지 결정하고 이것을 데이터베이스에서 검색한 후에 데이터베이스를 갱신하는 상태 없는(stateless) 컴포넌트와 상태가 있는 컴포넌트는 클라이언트들이 유일한 객체 구분자를 사용하는 특정 객체 서비스를 요청하는 상태 있는(stateful) 컴포넌트 두 가지가 있다.

2.2.2 3계층 클라이언트 서버 구조

3계층에서 클라이언트에서는 GUI를 제공하고 원격 서비스나 메소드를 통해 서버와 상호 작용한다. 애플리케이션 로직은 중간 계층에 나타난다. 3계층에서 중간 계층의 비즈니스 프로세스들은 사용자 인터페이스 및 데이터베이스와 분리해 관리하고 전개할 수 있다. 비즈니스 로직은 이제 고유의 분리된 계층을 가지고 하나 이상의 서버 상에서 실행할 수 있게된다. 3-계층(계층)의 장점은 다음과 같다[4][5].

- 대규모 애플리케이션의 요구에 적합
- 대부분의 코드가 서버 상에서 실행
- 네트웍 트래픽을 최소화
- 클라이언트는 비즈니스 로직만을 호출
- 데이터베이스를 클라이언트에 노출하지 않음

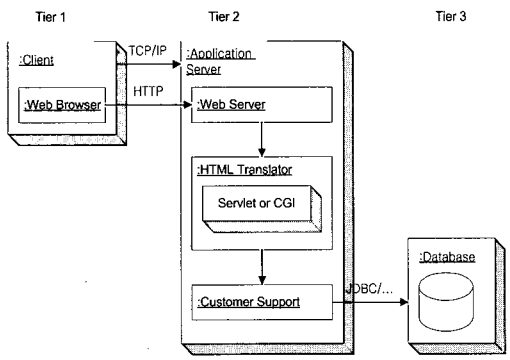


(그림 3) 3-계층 구조

2.3 인터넷 컴포넌트 환경의 디자인 패턴

인터넷 환경에서 시스템을 구현할 때 컴포넌트를 이용하여 사용되는 디자인 패턴을 UML(Unified Modeling Language) deployment 다이어그램으로 나타내면 다음과 같다[3].

2.3.1 웹서버와 CGI 및 서블릿 이용 분산

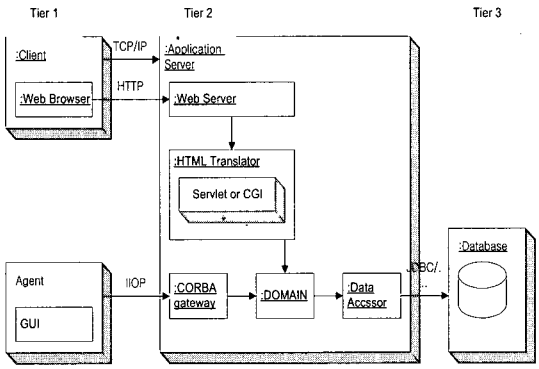


(그림 4) 웹서버와 CGI 및 서블릿 패턴

Tier 1은 브라우저, Tier 2는 서블릿이나 CGI,로 연동하는 애플리케이션 서버 그리고 Tier 3은 데이터베이스 스토어로서 동작한다. 이 경우는 모든 입출력 처리가 브라우저에 의한 인터페이스로 가능한 경우이다.(그림 4)

2.3.2 웹서버와 CORBA 이용 분산 패턴

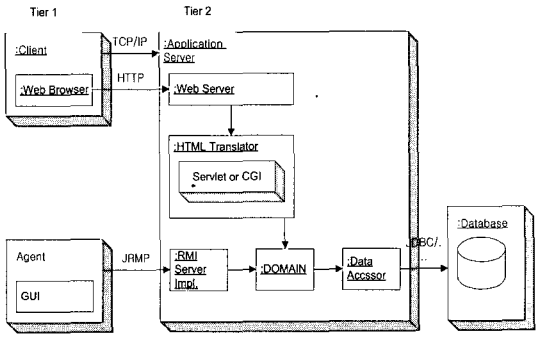
Tier 1은 브라우저와 CORBA 클라이언트, Tier 2는 서블릿과 CORBA 구현 객체 그리고 Tier 3은 데이터베이스 스토어로서 동작한다. 이 경우는 이질적인 환경이나 기존의 이질적인 (legacy) 시스템과의 통합이 요구된다.(그림 5)



(그림 5) 웹서버와 서블릿, CORBA 패턴

3) 웹서버와 RMI 이용 분산 패턴

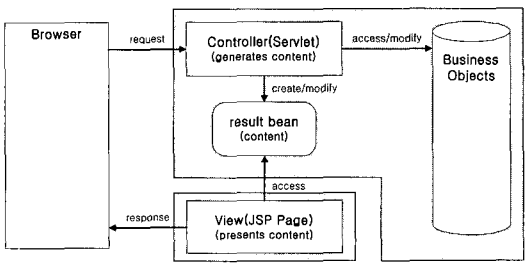
Tier 1은 브라우저와 RMI(Remote Method Invoca- tion) 클라이언트, Tier 2는 서블릿과 RMI 구현 객체 그리고 Tier 3은 데이터베이스 스토어로서 동작한다. 이 경우는 사용자 인터페이스 환경이 동적인 경우에 사용할 수 있으며 자바로 구성된 시스템에 사용될 수 있다.(그림 6)



(그림 6) 웹서버와 서블릿, RMI 이용 패턴

2.4 Model 2 프레임웍

Model 2 프레임웍은 모델, 뷰, 컨트롤러를 이용하여 MVC(Model-View-Controller)의 구조로 작성하는 방법으로 기능을 캡슐화하여 변경에 대해 영향을 최소로 하는 구조이기 때문에 확장성, 유지보수성 면에서 좋은 방법으로 인식되고 있다[7]. 모델 2 구조는 View와 비즈니스 객체를 분리하여 웹 개발 프로젝트를 작성하는 것을 기본적인 내용으로 한다. 개발 중에 비즈니스 객체의 상태가 변경될 요소가 많은 경우에는 비즈니스 객체의 개발이 뷰와 연관되지 않도록 하기 위해서 모델 2 구조를 사용하는 것이 바람직하게 된다. 그림 7의 Model 2 프레임웍 구조에서 내용을 표현하는 부분과 내용의 생성 작업 부분이 분리되어 처리됨을 알 수 있다.



(그림 7) Model 2 프레임웍 구조

3. 컴포넌트 자동 생성 시스템 설계

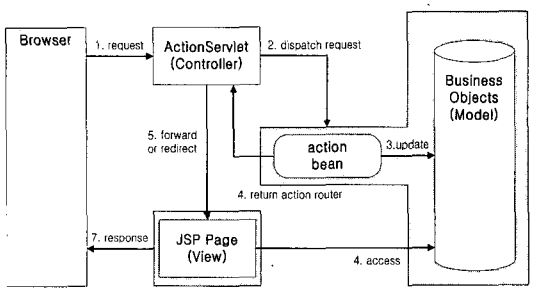
컴포넌트 자동 생성 시스템을 구현하기 환경을 다음과 같이 설정하고 설계한다.

3.1 구현 환경에 따른 Model 2 프레임웍 구조

컴포넌트 자동 생성 시스템의 구현 환경은 Java 기반의 웹 애플리케이션으로 설정하며 Model 2 프레임웍에 대한 MVC 부분은 다음과 같다.

- Model : Java Bean
- View : JSP(Java Server Page)
- Controller : Java Servlet
- 서블릿 명세 2.2
- JSP 명세 0.92
- JDK 1.3

위와 같이 JSP, 서블릿 및 Java 빈 환경에서 컴포넌트 자동 생성 시스템을 위한 Model 2 프레임웍은 다음과 같이 개선될 수 있다.



(그림 8) 구현환경의 Model 2 구조

이 프레임웍에서는 하나의 서블릿을 컨트롤러로서 사용한다. 이 서블릿을 액션 서블릿이라고 하며 모든 HTTP 요청은 액션 서블릿에 의해 처리되어 요청(request)을 자바 빈으로 전달하게 된다.

액션 빈은 비즈니스 객체를 갱신하고 제어를 액션 서블릿으로 리턴한다. 액션 서블릿에서는 처리 결과를 표시하기 위해 포워드(forward)하거나 재지향(redirect)를 통해 JSP 페이지를 호출하게 된다. 이 JSP 페이지에서는 비즈니스 객체를 참조하거나 비즈니스 객체를 참조하는 커스텀 태그를 정의하여 페이지를 구성한 후에 브라우저에 응답하는 구조이다.

3.2 자동 생성될 컴포넌트 및 요소

3.1의 Model 2 프레임웍 구조에 입각하여 컴포넌트 자동 생성 시스템에서 생성될 컴포넌트 및 요소는 다음과 같다.

3.2.1 기본 문서 생성(View)

- index.jsp
- menu.jsp
- 입출력 관련 jsp
- 목록보기 화면 jsp

3.2.2 자바빈 생성(Model)

- Type 클래스 파일
- DAO 클래스 파일
- JspBean 클래스 파일

3.2.3 기타 파일(Controller)

- Controller Servlet & Action Servlet
- Action Router
- Action Factory
- 커스텀 태그

3.2.4 DDL 생성

데이터베이스 table 생성을 위한 SQL 파일이 생성된다.

3.3 데이터베이스 테이블 구조

본 컴포넌트 자동 생성 시스템에서 사용하는 데이터베이스 테이블은 다음과 같다.

3.3.1 TB_STATUS Table

TB_STATUS 테이블은 데이터소스명과 데이터베이스 테이블 수를 저장하는 테이블이며 구조는 다음과 같다.

〈표 1〉 TB_STATUS

필드명	의 미	타입	길이	비고
uid	일련번호	int	11	PK, NN
sname	DataSource	varchar	100	
tblcnt	테이블 수	int	11	
dbmsno	DB 종류	int	11	

3.3.2 TB_INFO Table

TB_INFO 테이블은 테이블 명칭, class명칭, 화면명칭, 총컬럼수 등을 저장하는 테이블이며 구조는 다음과 같다. statusno는 TB_STATUS의 uid의 값을 유지하는 외래키이다.

〈표 2〉 TB_INFO

필드명	의 미	타입	길이	비고
uid	일련번호	int	11	PK,NN
statusno	작업번호	int	11	FK
tblname	테이블명	varchar	100	
clsname	클래스명	varchar	100	
pkgname	패키지명	varchar	100	
srcname	화면표시명	varchar	100	
totalcolno	컬럼수	int	11	
totalpkno	PK 수	int	11	

3.3.3 TB_RECORD Table

〈표 3〉 TB_RECORD

필드명	의 미	타입	길이	비고
uid	일련번호	int	11	PK,NN
infno	테이블번호	int	11	FK
object_field	객체 멤버명	varchar	100	
table_column	테이블 컬럼명	varchar	100	
screen_name	화면표시명	varchar	100	
field_type	타입	int	11	
field_size	필드 크기	int	11	
field_presition	소수점	int	11	
listing	리스팅 여부	int	11	
ispk	PK 여부	int	11	

TB_RECORD 테이블은 해당 테이블의 각 필드에 대한 설정 내용을 저장하는 테이블이며 구조는 다음과 같다. infno 필드는 TB_INFO의 uid의 값을 유지하는 외래키이다.

4. 컴포넌트 자동 생성 시스템 구현

4.1 컴포넌트 자동 생성 시스템 구현 환경

컴포넌트 자동 생성 시스템에서 클라이언트는 JSP(Java Server Page)의 HTML 출력을 표현하는 웹 브라우저이고 이 시스템 역시 Model 2 프레임웍에 기반하여 작성하였다. 그리고 데이터베이스는 MySQL 3.2를 사용하였으며 JDBC (Java Database Connectivity)를 이용 데이터베이스 연결 풀에 의해 처리한다.

Model 2 프레임웍 구조에 따라 Model은 데이터베이스와 질의 관련 컴포넌트가 되고 View는 JSP의 출력이 되며 Controller는 액션 서블릿으로 구현하여 데이터베이스에 관련된 처리를 포함하여 세션(session) 관리 사용자 입력 정보의 보관 등을 처리한다.

4.2 소스코드 생성에 관련된 리소스 파일

컴포넌트 자동 생성과 관련된 리소스 파일 및 기능은 다음과 같다.

4.2.1 Model 생성 관련 파일

DAO 클래스를 생성하기 위해 DAO class전체 골격을 갖추는 파일(DAO.template), SQL insert(DAO.create.template), 목록 조회에 관련한 메쏘드파일(DAO.list.template), SQL remove(DAO.remove.template), SQL select (DAO.select.template), SQL update(DAO .update.template) 등이 사용된다.

4.2.2 View 생성 관련 파일

한건의 레코드를 반영하는 화면(JSP.single. template), 여러건의 목록을 반영하는 화면 (JSP.list.template)과 부가 페이지 생성 파일로 에러 발생시 반영될 에러 페이지(JSP.error. template), 프레임파일(JSP.index. template), 좌측의 각 테이블에 대한 프로그램별 링크 메뉴 (JSP.menu.template), 초기 화면으로 보여질 비어 있는 페이지(JSP.right. template) 등이 사용된다.

4.2.3 Controller 생성 관련 파일

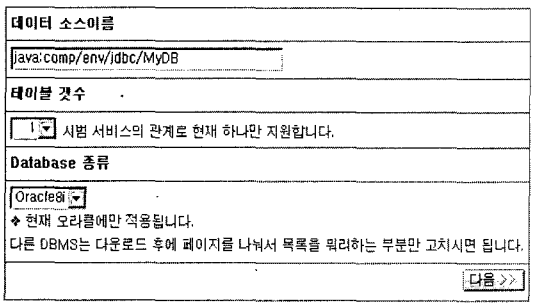
SingleJspBeans을 생성하기 위해 SingleJSP 에서 넘어오는 요청을 처리하는 JSP Bean(JSPBEAN. single.template,JSPBEAN.single.execute.template)과 ListJSP 에서 넘어오는 요청을 처리하는 JSP Bean(JSPBEAN.list.template, JSPBEAN.list. execute.template)으로 구성된다.

4.3 컴포넌트 자동 생성 시스템의 수행

컴포넌트 자동 생성 시스템의 수행은 다음의 3 단계의 사용자 인터페이스에 의해 처리된다.

4.3.1 1 단계

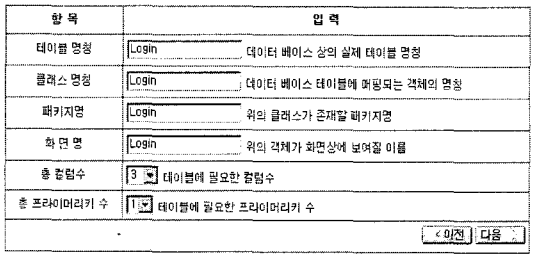
데이터 소스명과 데이터베이스 테이블 개수를 입력한다. 입력되는 데이터 소스명은 생성되는 자바 빈(Model)에서 JDBC 연동시 JDBC 2.0 스펙에 의해 사용되는 데이터베이스의 경로를 의미한다. 현재 구현 상태에서는 테이블의 수를 하나로 하였으나 Model 2 프레임웍에 응용하기 위해서는 같은 작업을 각 엔터티에 대해 수행해 주면 된다. 데이터베이스를 지정하는 부분은 JDBC 연동시 사용될 SQL의 작성에 기준이 된다.



(그림 9) 사용자 인터페이스 1 단계

4.3.2 2 단계

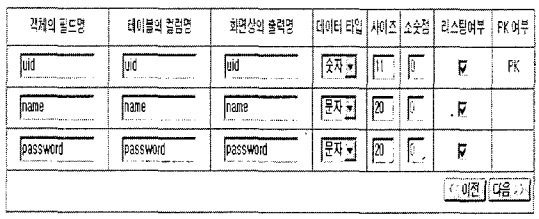
테이블 명칭, class명칭, 화면명칭, 총 컬럼 수 등을 입력한다. 테이블 명칭은 DB의 SQL에서 사용될 이름이며 class명칭은 Model 부분에 사용되기 위해 생성되는 자바빈의 클래스 이름으로 사용되고, 패키지 이름은 생성되는 산출물이 자바빈 이외에 3.2 절에서 언급한 파일들이 모두 생성되므로 패키지화하기 위해 요구된다. 그리고, 화면명칭은 JSP(View) 생성시 사용되는 이름이며 총 컬럼 수 및 총 프라이머리키 수는 3단계 작업에서 SQL 테이블의 기본 명세를 작성할 때 테이블의 크기를 정하는데 사용된다.



(그림 10) 사용자 인터페이스 2 단계

4.3.3 3 단계

해당테이블의 각필드에 대한 설정 후 생성된 zip 파일 다운로드한다. 이 단계에서 입력되는 SQL 테이블 작성에 필요한 내용을 작성하는데 이 내용은 SQL 테이블과 자바빈(Model) 생성 및 관련 모듈 생성에 기본 자료로 사용된다.



(그림 11) 사용자 인터페이스 3 단계

4.3.4 수행결과 산출물

RDBMS 데이터베이스 명세를 이용하여 컴포넌트 자동 생성 시스템에서 생성될 컴포넌트 및 요소는 다음과 같다. 다음 표는 기준 이름이 "Emp"인 경우이다.

〈표 4〉 MVC의 각 Component 별 구성요소

구 분	분 류	필수요소	명명법
Model	Entity	SingleEntity	Emp.java
		ListEntity	EmpList.java
	Entity Control	DAO	EmpDAO.java
View	jsp	SingleJSP	Emp.jsp
		ListJSP	EmpList.jsp
Controller	JspBeans	SingleJspBean	EmpJspBean.java
		ListJspBean	EmpListJspBean.java

수행결과 산출물들 중에서 가장 기본이 되는 자바 빈 클래스(Entity)의 내용은 다음과 같다.

```
package Login;

public class Login implements
java.io.Serializable
{
 private long uid;
 private java.lang.String name;
 private java.lang.String password;

 public long getUid(){
        return uid;
 }

 public java.lang.String getName(){
        return name;
 }

 public java.lang.String getPassword(){
        return password;
 }

 public void setUid(long uid){
        this.uid=uid;
 }

 public void setName(java.lang.String name){
        this.name=name;
 }

 public void setPassword(java.lang.String password){
        this.password=password;
 }
}
```

(그림 12) 생성된 자바 빈 클래스

5. 결론 및 연구 방향

본 연구에서는 웹 애플리케이션에 개발시에 자주 이용되는 컴포넌트와 관련 요소들을 자동 생성하기 위해 환경에 적합한 Model 2 프레임웍을 제시하고 이 프레임웍에 기반하여 소프트웨어 컴포넌트와 관련 요소들을 생성하는 시스템을 설계하고 구현하였다. Model 2 프레임웍은 웹 애플리케이션을 MVC 구조로 작성하기 때문에 기능을 캡슐화하여 변경에 대해 영향을 최소로 하는 구조이므로 확장성, 유지보수성 면에서 좋은 방법이다. 본 연구에서 설계하고 구현한 컴포넌트 자동생성 도구는 웹 애플리케이션 개발시에 활용되어 생산성 향상에 기여할 수 있다.

연구 방향으로는 데이터의 호환성 및 표준화를 위해 XML과 DTD를 정의하는 부분과 국제화 처리 부분 등을 보완하고 웹 애플리케이션의 템플릿 기능을 고려하는 것이 필요하다.

참고 문헌

[1] Component Methodologies, "Component Development Strategies", Vol.8 No.11, November 1998.

[2] Brown, "Tool Support for Enterprise scale CBD, Component Strategies", pp. 22-31, 1998.

[3] Alan W, Brown and Keith Jaeger, "The Future of Enterprise Application Development with Component and Patterns", STERLING SOFTWARE, Aug., 1998.

[4] Orfali, Harkey and Edwards, "The Essential Distributed Objects Survival Guide", Wiley Press, 1996

[5] Component Methodologies, "Component Develop- ment Strategies", Vol.8 No.11, November 1998.

[6] Steve Burbeck, "Application Programming in Smalltalk-80: How to use Model-View-Controller (MVC). Available at http://st-www.cs.uiuc.edu/users/march/st-docs/mv.html, 1992.

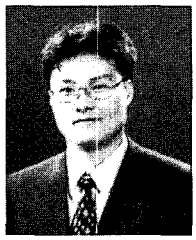
[7] David M. Geary, "advanced Java Server Pages", PH Press, 2000.

[8] Gamma, Helm, Johnson, Vlissides, *Design Pattern*. Addison-Wesley, 1995.

[9] Fields and Kolb. *Web Development with JSP*, Manning 2000.

천 상 호

1986년 서강대학교 수학과 졸업(이학사)
1995년 강원대학교 대학원 컴퓨터과학과 석사과정 수료
1985년 ~ 89년 한국화약 전산실
1989년 ~ 97년 주)선경유통
1997년 ~ 현재 주)오픈시스템서비스 대표이사
관심분야: 객체지향 방법론, 컴포넌트 시스템, 멀티미디어 컨텐츠, 분산 시스템



권 기 현

1993년 강원대학교 전자계산학과 졸업(이학사)
1995년 강원대학교 대학원 전자계산학과 졸업(이학석사)
2000년 강원대학교 대학원 컴퓨터과학과 (이학박사)
1998년 ~ 현재 동원대학 인터넷 정보과 조교수
관심분야: 객체지향 방법론, 컴포넌트 시스템, 컴포넌트 방법론, 분산 시스템

최 형 진

1982년 영남대학교 물리학과 졸업 (이학사)
1987년 일본동경공업대학 정보공학과 (공학석사)
1990년 일본동경공업대학 정보공학과 (공학박사)
1990년 ~ 91년 ETRI 선임연구원
1991년 ~ 현재 강원대학교 전자계산학과 부교수
관심분야: 인공지능, 화상처리, 패턴인식, 컴퓨터그래픽