EJB 컴포넌트 기반 WAP 응용 생성기☆

A EJB Component-based WAP Application Generator

음 두 헌* 강 이 지**
Doohun Eum Izzy Kang

요 약

무선 인터넷 사용자의 급증과 함께 모바일 전자상거래가 활성화 되고 있다. 이러한 모바일 전자상거래에 대한 수요를 만족시키기 위해 모바일 전자상거래를 지원하는 모바일 응용의 생산성 향상이 요구된다. 본 논문에서 소개하는 WAPSiteGen은 데이터베이스부터 구축해야 하거나 기존 데이터베이스를 사용하는 모바일 응용의 생산성 향상을 위해 3-계층 구조를 갖는 WAP 응용을 자동 생성한다. WAPSiteGen은 3-계층 중, 비즈니스 로직 층을 담당하는 EJB 컴포넌트들과 함께 질의 처리 및 푸쉬 서비스 기능을 제공하는 JSP 웹 컴포넌트들을 자동 생성하고 프리젠테이션 층으로는 처리된 결과를 보여주는 관리자용 HTML 폼과 고객용 WML 데크들과 이 데크를 구성하는 카드들을 자동 생성한다. 또한, WAPSiteGen은 EJB와 JSP 등의 컴포넌트 기술을 도입함으로써 3-계층 구조를 갖는 WAP 응용의 확장성, 재사용성, 이식성 등도 향상시킬 뿐 아니라 고객이 원하는 시점과 장소에 정보를 제공할 수 있는 WAP 푸쉬 서비스를 갖는 응용을 생성한다. WAPSiteGen은 비슷한 기능을 지원하는 상용 도구들과 달리 생성하는 응용이 질의의 대상인 개체와 함께 이 개체와 연관된 모든 개체들의 집합을 한 단위로 검색 할 수 있어 연관된 정보들을 신속하게 제공한다. 본 논문에서는 WAPSiteGen의 기능 및 구현 원리를 설명하고 상용화된 모바일 응용 생성기들과의 기능을 비교하여 WAPSiteGen의 장점을 설명한다.

Abstract

Mobile e-business is in wide use with the rapid growth of wireless internet users. To meet the growing requests for mobile e-business the productivity increase of mobile applications has been demanded. The WAPSiteGen, introduced in this paper, automatically generates a 3-tier WAP application that needs a new application database or that already has an application database ,to increase the productivity. The WAPSiteGen generates the EJB components that handle business logic and the JSP Web components that process user-made queries and the WAP Push feature. For a presentation layer, it also generates the HTML forms for an application manager and the WML decks for end users, which consists of many cards. The WAPSiteGen enhances extendibility, reusability and portability of generated 3-tier applications by comprising such component technologies as EJB and JSP. Furthermore, it supports the WAP Push services for generated applications that provide necessary information to wherever and at whenever a user wants. Since the applications generated by the WAPSiteGen provide the information on an interested entity as well as the information on all the directly or indirectly related entities to the interested one, it shows faster information accessibility. In this paper, we explain the functionality and implementation of the WAPSiteGen and then show its merits by comparing the WAPSiteGen to commercial WAP application generators.

☞ Keyword : automatic generator, software productivity, components, WAP applications, mobile applications, EJB

1. 서 론

최근 무선 인터넷 사용자가 급증하고 있다. 국민은행에서 이미 시행하고 있는 '뱅크온'이라는 모바일 뱅킹 서비스를 시작으로 모바일 전자상거래 응용의 수요가 증대될 것으로 예측된다[1]. 따라서, 이러한 모바일 전자상거래에 대한 수요를 만족시키기 위해 모바일 전자상거래를 지원하는

* 정 회 원 : 덕성여자대학교 컴퓨터공학부 교수
dheum@duksung.ac.kr(제 1저자)
** 준 회 원 : 아주대학교 중앙전산원
izzy@duksung.ac.kr(공동저자)
[2004/09/06 투고 - 2004/10/04 1차 - 2004/12/16 2차 - 2005/04/25 심사완료]
☆ 본 연구는 덕성여자대학교 2004년 자연과학연구소 연구비 지원으로 이루어졌음

모바일 응용의 생산성 향상이 요구된다. 모바일 전자상거래 응용의 핵심 요소는 모바일 기반 데이터베이스 응용 기술과 무선 이동 장비를 통한 사용자 인터페이스 기술이다[2].

무선 이동 장비를 이용하는 무선 통신 기술 중 대표적인 것으로는 WAP 포럼에서 제정한 WAP (Wireless Application Protocol)[3]이 있다. WAP을 기반으로 한 WAP 응용은 무선 인터넷 마크업 언어인 WML(Wireless Markup Language)[4]을 사용해 실현되며 모바일 응용의 대표적인 형태이다. WML에서 카드(card)는 무선 이동 장비의 화면 표시 단위이고 데크(deck)는 하나의 WML 문서로서 카드들의 집합을 말한다.

EJB는 컴포넌트를 기반으로 하는 분산 비즈니스 응용의 개발과 배치를 위한 표준 아키텍처이다. EJB에서는 3-계층 구조를 가지는 응용의 비즈니스 로직을, 데이터 로직을 처리하는 엔티티 빈(entity bean)과 순수 비즈니스 로직만을 담당하는 세션 빈(session bean)의 두 가지로 나눈다[5]. 엔티티 빈은 데이터베이스의 엔티티 개체에 대한 맵핑으로 볼 수 있다. 세션 빈은 서로 다른 빈 간의 상호작용을 담당하는 비즈니스 메소드를 가지며 엔티티 빈과는 달리 빈 내부에서 사용자 정의 트랜잭션 처리도 가능하다. 웹 컴포넌트인 JSP(Java Server Page)는 동적 HTML 및 WML 코드를 생성하는 과정을 간단하게 만드는 서블릿 컴포넌트 모델을 확장한 것이다. JSP는 JDBC(Java Database Connectivity)를 이용하거나 엔티티 빈을 이용해 데이터베이스에 접근할 수 있다. JSP가 동적으로 생성하는 HTML 및 WML 코드는 자바 애플릿에 비해 용량이 매우 작고 기본 플랫폼에서도 실행되며 전자상거래에서 보안을 위해 설치한 방화벽을 통과할 수 있기 때문에 JSP는 3-계층 응용의 프리젠테이션 로직을 담당하는 컴포넌트로 적합하다.

현재의 모바일 전자상거래 응용은 기존에 유사한 응용이 있어도 이를 재사용하지 못하고 처음부터 새로 작성되고 있다. 그러나 소프트웨어의 부품화, 분산화, 통합화 및 개방화를 지원하는 컴포넌트 기술을 도입하면 새로운 응용을 기존의 컴포넌트들로부터 신속하고 효율적으로 생산할 수 있다. 본 논문에서는 컴포넌트 기반이면서 3-계층(3-tier) 구조를 갖는 WAP 데이터베이스 응용을 자동 생성하여 모바일 응용의 대표적인 형태인 WAP 응용의 생산성 향상은 물론 재사용성과 확장성을 지원하는 WAPSiteGen을 소개한다. WAPSiteGen은 3-계층 중 데이터베이스 층을 담당하는 데이터베이스는 클래스 다이어그램을 이용해 새롭게 구축하거나 기존의 데이터베이스를 사용한다[6]. 비즈니스 로직 층을 담당하는 EJB 컴포넌트들과 함께 질의 처리 및 푸쉬 서비스 기능을 제공하는 JSP 웹 컴포넌트들을 자동 생성하고 프리젠테이션 층으로는 질의에 대한 결과 데이터를 보여주는 관리자용 HTML 폼과 고객용 WML 데크를 자동 생성한다. 푸쉬 기술은 클라이언트/서버 모델을 기반으로 하지만, 사용자가 정보를 가져오기 위해 서버에 등록을 하고 이후, 서버는 사용자의 요청 없이도 정보를 자동으로 보낼 수 있다. 인터넷상에서 푸쉬 생성자(Push Initiator)는 푸쉬 메시지를 초기화 하고, PPG(Push Proxy Gateway)로 푸쉬 초기화 메시지를 전송한다. PPG는 푸쉬 생성자로 푸쉬 제출(push submission) 승인 또는 거절 메시지를 전송하게 된다. 그리고, OTA(Over The Air) 프로토콜을 사용해서 무선 장치로 푸쉬 메시지를 전송한다. 마지막으로 PPG는 푸쉬 생성자로 결과를 전송한다[7].

잘 알려진 상용 WAP 응용 생성기로는 Microsoft의 Visual Studio .NET[8]과 AnyBil의 Any Builder Enterprise[9], 그리고 Speedware의 Mobile Dev[10] 등이 있으며 모두 2-계층 구조의 응용을 생성한다. Visual Studio .NET, AnyBuilder, 그리고 MobileDev는 GUI를 사용해 WML 데크들을 반자동으로 생성한다. 생성된 WML 데크는 GUI 설계 시 지정된 테이블에 관한 질의 처리 결과만을 담을 수 있다. 반면, WAPSiteGen은 응용 데이터베이스로부터 자동으로 WAP 응용을 생성할 수 있을 뿐만 아니라 생성된 응용은 현재 질의의

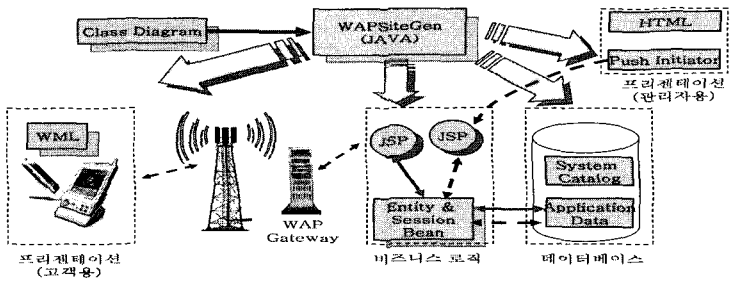
대상인 닻 개체(anchor entity)는 물론 닻 개체와 일대일 또는 다대일 관계 타입으로 연관되는 첨부 개체(appended entity)들과 일대다 또는 다대다 관계 타입으로 연관되는 확장 개체(expended entity)들의 집합을 한 단위로 검색한다. 검색된 정보는 하나의 데크로 구성되어 무선 이동 장비에 전송되어 사용자 인터페이스로 제공된다. 따라서, 관심의 대상인 닻 개체와 연관된 정보들을 한번의 질의로 신속하게 접근할 수 있다.

2절에서 WAPSiteGen의 개요를 설명하고 3절에서 상용 WAP 응용 생성기들과의 기능을 비교한 후, 4절에서는 WAPSiteGen의 구현을 간략히 설명한다. 마지막 절에서는 본 논문의 결론과 앞으로의 연구 방향을 정리한다.

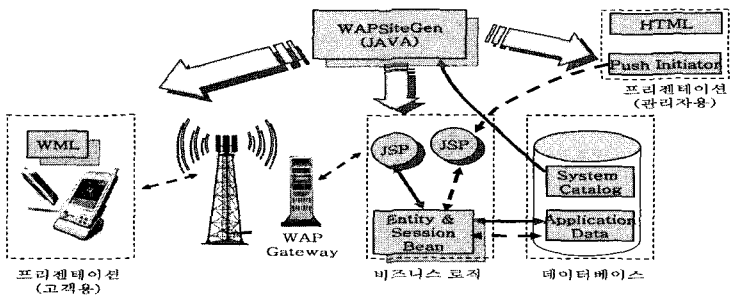
2. WAPSiteGen의 개요

본 절에서는 WAP 데이터베이스 응용을 자동으로 생성하는 도구인 WAPSiteGen의 개요를 소개한다. WAPSiteGen의 기능을 설명하기 위해 본 논문에서는 간단한 전자상거래 응용(Titan 유람선 예약 시스템[11])의 생성을 예로 든다.

Java로 구현된 WAPSiteGen은 그림 1과 같이 두 가지 방식으로 프리젠테이션, 비즈니스 로직, 데이터베이스의 3-계층 구조를 가지는 WAP 응용을 자동 생성한다. 그림 1의 (a)는 WAP 응용을 위한 클래스 다이어그램으로부터 카디널리티와 참조 무결성 조건 등의 정보를 추출해 필요한 응용 데이터베이스를 구축하고 이 데이터베이스와 연동하여 비즈니스 로직을 담당할 EJB 컴포넌트들과 이 EJB 컴포넌트들을 사용하여 질의를 처리하고 푸쉬 메시지를 전송할 수 있는 JSP 웹 컴포넌트들을, WAPSiteGen이 자동 생성하는 첫 번째 방식을 나타낸다. 프리젠테이션 층으로는 유선으로 연결될 관리자의 질의 및 처리를 위한 HTML 폼들을 생성하며 무선으로 연결될 고객의 질의 및 처리를 위한 WML 데크들을 생성한다. 두 번째 방법은 그림 1의 (b)와 같이, WAPSiteGen이 기존



(a) 클래스 다이어그램으로부터 응용 생성



(b) 기존 데이터베이스로부터 응용 생성

〈그림 1〉 WAPSiteGen이 생성하는 응용의 구조

데이터베이스의 시스템 카탈로그로부터 스키마 정보를 추출하여 그림 1의 (a)에서 생성하는 것처럼 데이터베이스와 연동하여 비즈니스 로직을 담당할 EJB 컴포넌트들, JSP 웹 컴포넌트들, HTML 폼과 WML 데크들을 생성한다. WAPSiteGen은 현재 ORACLE8i 서버[12]와 MS-SQL 서버 7.0[13]으로 구축된 데이터베이스를 이용할 수 있다. 본 논문에서는 ORACLE8i 서버의 사용을 가정한다.

그림 2는 WAPSiteGen으로 작성한 Titan WAP 데이터베이스 응용의 UML 클래스 다이어그램이다. Titan 응용은 다음을 가정한다. 데이터베이스와 연동될 엔티티 빈들의 방향성은 양방향으로 한다.

1) 고객(CUSTOMER)은 고객의 이름, 주소, 신용카드에 관한 정보로 관리된다. 각 고객에게는 아이디가 기본 키로 주어지고, 고객은 신용카드(CREDIT_CARD)를 하나 가지거나 가지지 않을 수 있고, 하나 이상의 전화(PHONE)와 하나의 주소(ADDRESS)를 가지며 한 개 이상의 예약(RESERVATION)을 할 수 있다.
2) 예약은 지불된 요금, 예약날짜, 그리고 다수의 객실(CABIN)을 가지며 하나의 선박여행(CRUISE)과 연계되고 다수의 고객을 포함할 수 있다.
3) 선박여행은 하나의 선박(SHIP)이 배정되고 한 개 이상의 예약을 가진다.
4) 선박은 선박의 이름, 무게, 용적과 함께 다수의 객실을 보유한다.
5) 신용카드는 카드번호, 유효기간, 발급기관, 이름, 그리고 카드를 소유한 한 명의 고객으로 구성된다.
6) 전화는 전화번호, 타입, 그리고 해당 번호를 사용하는 한 명의 고객으로 구성된다.
7) 주소는 거리, 도시, 주, 우편번호, 그리고 해당 주소에 거주하는 한 명의 고객으로 구성된다.

2.1 프리젠테이션 층의 생성

WAPSiteGen은 사용자 인터페이스 층을 위해 다음의 그룹화 규칙을 통해 한 번의 질의 처리로 하나의 고객용 WML 데크 또는 관리자용 HTML 폼에 담을 수 있는 개체 타입들을 결정할 수 있다.

규칙 1. 현재 질의의 대상인 개체를 닻 개체라고 한다.

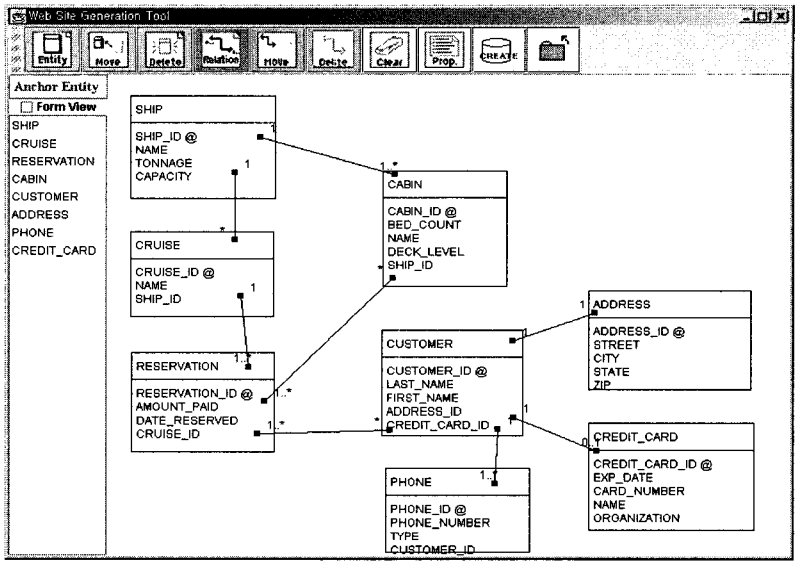
규칙 2. 닻 개체와 일대일 또는 다대일 관계인 개체를 첨부 개체라 하고 닻 개체와 일대다 또는 다대다 관계인 개체를 확장 개체라 한다.

규칙 3. 첨부 연산은 모든 개체(닻 개체, 첨부 개체 또는 확장 개체)에 적용 가능하며 하나의 WML 데크 또는 HTML 폼 내에서 첨부 연산의 회수에는 제한이 없다.

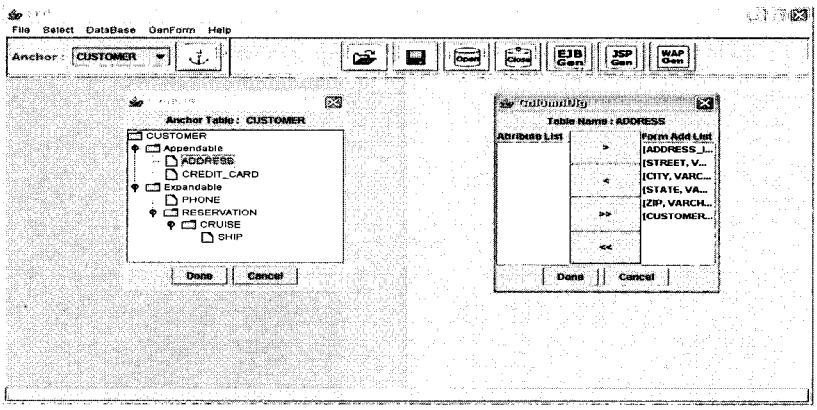
규칙 4. 확장 연산은 닻 개체 또는 닻 개체에 직·간접적으로 첨부된 개체로부터 확장할 때 사용된다. 하나의 WML 데크 또는 HTML 폼 내에는 한번의 확장을 통해 도달할 수 있는 개체들만이 포함될 수 있다.

WAPSiteGen이 생성하는 WML 데크 또는 HTML 폼의 종류는 Insert/Search 데크/폼, Select 데크/폼, Update/Delete 데크/폼, Result 데크/폼의 네 가지이다. Insert/Search 데크/폼은 데이터를 삽입하거나 검색할 때 사용된다. Select 데크/폼은 Insert/Search 데크/폼으로 검색한 결과 목록을 제공할 때 사용된다. Select 데크/폼의 목록 중 하나의 항목을 선택하면 Update/Delete 데크/폼이 생성되어 선택된 항목(닻 개체)과 연관된 정보들(첨부개체 및 확장 개체)을 제공한다. 사용자는 이 데크/폼들을 통해 데이터를 갱신하거나 삭제할 수 있다. Result 데크/폼은 수정, 삭제에 대한 결과를 보여준다. Titan 응용에서는 3개의 Update/Delete WML 데크 또는 HTML 폼들(CUSTOMER, CABIN, CREDIT_CARD)을 얻을 수 있다.

그림 3은 그림 2의 Titan 응용에서 고객(CUSTOMER)이 닻 개체로 선택된 경우, WAPSiteGen

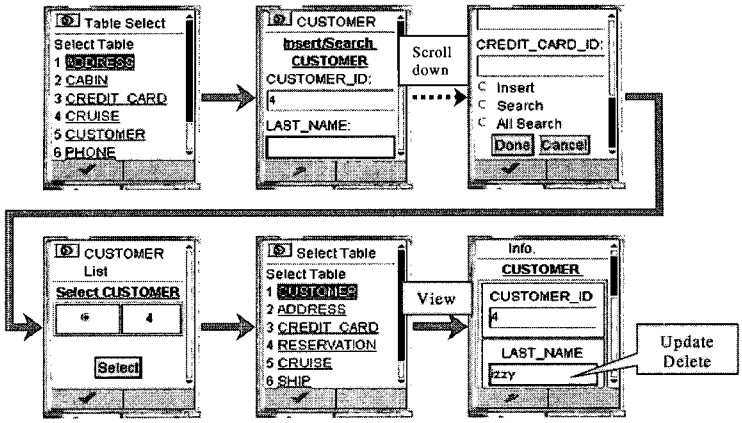


〈그림 2〉 Titan 응용의 클래스 다이어그램



〈그림 3〉 CUSTOMER를 닻 개체로 선택 시 TreeDlg와 ColumnDlg 다이얼로그 박스

이 그룹화 규칙에 의해 생성한 다이얼로그 박스이다. 닻 개체로 선택된 고객은 신용카드(CREDIT_CARD)와 일대일 관계이고, 주소(ADDRESS)와도 일대일 관계이므로 신용카드와 주소는 첨부 개체가 되어 TreeDlg 창의 Appendable 노드 밑에 표시된다. 고객과 일대다 관계 타입인 전화(PHONE)와 예약(RESERVATION)은 확장 개체가 된다. 첨부 개체에는 그 개체의 첨부 개체와 확장 개체가 다시 추가될 수 있고 확장 개체에는 그 개체의 첨부 개체가 추가될 수 있다. 따라서, 예약의 첨부 개체인 선박여행(CRUISE)과 선박여행의 첨부 개체인 선박(SHIP)도 TreeDlg 창의 Expandable 노드 밑에 추가된다. 트리에서 주소 개체를 선택하면 속성들이 ColumnDlg 창에 보여진다. ColumnDlg 창에서 검색할 속성으로 Address_ID와 Street, City, State, Zip의 모든 속성이 선택된다. 이렇게 선택되어진 개체와 속성들은 한 단위로 검색된다. 또한, TreeDlg 창에서 고객 테이블을 선택함으로써 푸쉬 서비스를 제공할 테이블로 고객이 지정되어 푸쉬 메시지가 제공된다.



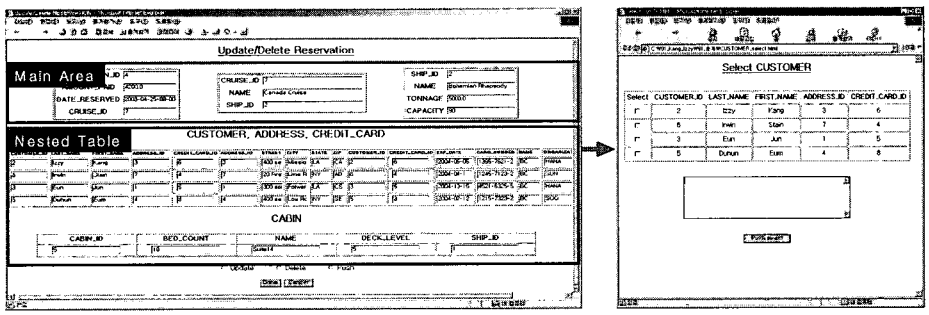
〈그림 4〉 WAPSiteGen이 생성한 Titan 응용의 WML 데크들

그림 4는 WAPSiteGen이 그림 3의 다이얼로그 박스를 통해 생성한 Titan 응용의 고객용 WML 데크들의 동작원리를 보인다.

생성된 응용을 실행하면 첫 화면에 데이터베이스 내의 모든 테이블 이름들이 브라우저 창에 보인다. 그 중 질의 대상 테이블(개체 타입)로 고객(CUSTOMER)을 선택하면 고객에 대한 정보를 입력하거나 검색하는 카드 화면이 보인다. CUSTOMER_ID 입력란에 4를 입력하고 스크롤바를 내려 Select를 선택한 후 Done 버튼을 누르면 CUSTOMER_ID가 4인 개체(닻 개체)와 그룹화 규칙에 의해 그림 3과 같이 닻 개체와 연관된 모든 개체들에 관한 정보가 한번의 질의로 하나의 Update/Delete 데크에 담긴다. 비슷한 방법으로 CABIN과 CREDIT_CARD를 닻 개체로 하는 Update/Delete CABIN 데크와 Update/Delete CREDIT_CARD 데크도 생성된다.

그림 5는 WAPSiteGen이 생성한 관리자용 Reservation Update/Delete HTML 폼이다.

Reservation 폼은 주 영역(Main Area)과 테이블(Nested Table)영역으로 구성된다. 주 영역에는 닻 개체인 Reservation과 Reservation의 첨부 개체인 Cruise와 Ship이 표시된다. 내포된 테이블에는 확장 개체 및 이 확장 개체의 첨부 개체가 표시된다. Reservation과 다대다 관계로 확장 개체인 Customer와 Customer의 첨부 개체인 Address와 Credit_Card, Reservation과 다대다 관계로 확장 개체인 Cabin이 내포된 테이블에 표시된다. 폼에서 Push 라디오 버튼을 선택하고 Done 버튼을 누르면, 원하는 Customer들을 선택하여 푸쉬 메시지



〈그림 5〉 Reservation Update/Delete HTML 폼과 Push 폼

를 전송할 수 있는 Push 폼이 생성된다.

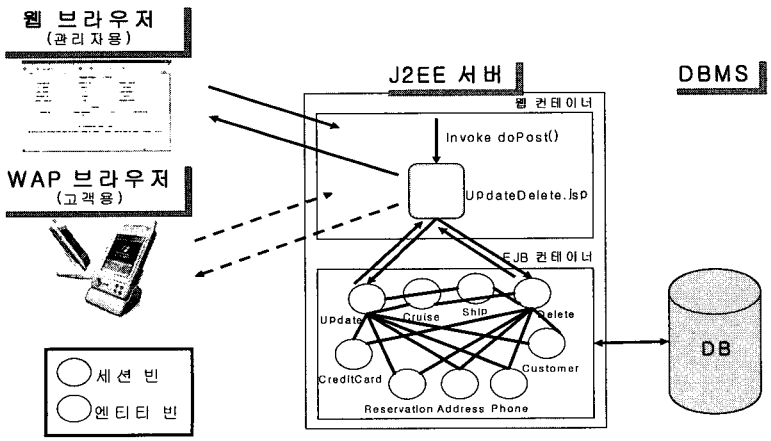
2.2 비즈니스 로직 층의 생성

WAPSiteGen은 그림 3의 TreeDlg 창과 ColumnDlg 창을 통해 얻은 정보를 토대로 고객용 사용자 인터페이스 데크들과 관리자용 HTML 폼들을 생성 한 후, 이들을 통해 이루어질 질의를 처리할 EJB 및 JSP 컴포넌트들을 생성한다. 생성될 EJB 컴포넌트들은 데이터베이스 내의 모든 개체 타입에 대한 데이터 로직을 담당하는 엔티티 빈들과 응용의 비즈니스 로직을 담당하는 세션 빈들이다. 세션 빈은 각 개체 타입 별로 InsertBean, SelectBean, UpdateBean, DeleteBean의 4가지씩 생성된다. 생성될 JSP 컴포넌트는 사용자 인터페이스인 WML 데크 또는 HTML 폼과 함께 질의 결과를 WAP 브라우저 또는 웹 브라우저에 보내 사용자에게 보여주거나 이들로부터 받은 질의를 처리하기 위해 적절한 세션 빈을 호출한다. JSP 컴포넌트는 각 개체 타입 별로 사용자 인터페이스 폼에 따라 Insert.jsp, Select.jsp, UpdateDelete.jsp, Result.jsp의 4가지씩 생성된다.

WAPSiteGen이 생성하는 WAP 응용 중, EJB 컴포넌트는 RMI-IIOP 프로토콜을 이용해 분산 객체 간의 통신을 담당하고, JSP 컴포넌트는 방화벽을 통과하기에 적합한 WML 또는 HTML 코드를 동적으로 생성한다. 따라서 생성된 컴포넌트들은 분산과 방화벽 문제를 모두 해결할 수 있는 조합이 된다[14].

그림 6은 WAPSiteGen이 생성한 Titan 응용의 구조도이다. WAPSiteGen은 J2EE 서버 환경을 가정한다. J2EE 서버의 웹 컨테이너엔 JSP 컴포넌트들이 위치하고 EJB 컨테이너엔 EJB 컴포넌트들이 위치한다. 사용자 인터페이스인 WML 데크 또는 HTML 폼으로부터 질의를 받게 되면 해당 JSP 컴포넌트가 호출되고 이 JSP 컴포넌트는 질의 처리 기능을 가진 해당 세션 빈을 호출한다. 호출된 세션 빈은 데이터베이스 테이블들과 연동되는 엔티티 빈들을 조작하여 질의 결과를 얻게 되고 이는 JSP 컴포넌트를 통해 다시 WAP 브라우저 또는 웹 브라우저에 전송되어 표시된다.

그림 6에서, 그림 4의 CUSTOMER 데크와 같은 WML 데크로부터 데이터 갱신을 위한 질의를 입력받게 되면 갱신 작업을 수행하기 위한 JSP 컴포넌트인 UpdateDelete.jsp가 호출되고 이는 다시 갱신 작업을 위한 세션 빈인 UpdateBean을 호출한다. 호출된 UpdateBean은 닻 개체에 해당되는 엔티티 빈과 그 닻 개체와 연관된 첨부 개체와 확장 개체에 해당하는 엔티티 빈들의 데이터를 갱신하는 비즈니스 로직을 수행하고 이는 해당 데



〈그림 6〉 Titan 응용의 구조

이터베이스 테이블에 바로 반영된다. 삭제 질의도 비슷한 과정으로 수행된다. 데이터 입력과 검색을 위한 질의를 처리할 때도 마찬가지로 Insert.jsp와 Select.jsp 등이 사용된다.

WAPSiteGen이 생성하는 컴포넌트들은 이식성과 확장성이 좋으며 그 중 엔티티 빈들은 다른 응용을 위한 재사용성이 매우 높다. 또한, 다양한 비즈니스 로직을 수행하는 세션 빈들과 조립된다면 생성된 데이터베이스 응용을 완전한 WAP 응용으로 쉽게 확장할 수도 있다.

3. 상용 WAP 응용 생성기와의 기능 비교

본 절에서는 WAPSiteGen과 타 WAP 응용 생성기들의 기능을 비교한다. 표 1은 WAPSiteGen과 상용화된 WAP 응용 생성 도구인 AnyBuilder Enterprise, Visual Studio .NET, MobileDev의 기능을 비교한 것이다.

AnyBuilder Enterprise는 Anybil사의 제품이다. 데이터베이스와의 연동은 ODBC 드라이버를 이용하여 ORACLE, MSSQL, MySQL, MDB의 데이터베이스를 지원한다. 사용자 인터페이스로는 WML, HDML, mHTML 코드를 생성하며 ASP, JSP, PHP 질의 처리 코드를 반자동으로 생성하여 웹서버 환경에 대한 개발을 지원한다. AnyBuilder Enterprise는 폼에 나타낼 테이블과 그 테이블의 속성들을 선택적으로 구성하여 반자동으로 데크를 생성할 수 있다. 한 데크에 표현 가능한 개체는 닻 개체이다. 또한, AnyBuilder Enterprise는 데이터베이스를 새로 생성하지 못하고 기존 데이터베이스만 사용이 가능하다.

MobileDev는 Qeury Wizard를 사용하여 질의 처리를 하는 ASP, PERL 코드를 반자동으로 생성한다. 하나의 데크에는 닻 개체만 표현 가능하며 기존에 구축된 데이터베이스만 이용이 가능하다.

Visual Studio .NET은 폼에 필요한 GUI 요소들을 선택적으로 구성하여 HTML, WML, cHTML의 코드를 생성해주는 WAP 응용 생성기이다. 데크는 사용자의 선택에 따라 반자동으로 생성되며 하나의 데크는 AnyBuilder Enterprise 및 MobileDev와 같이 닻 개체만 표현할 수 있다. Visual Studio .NET도 새로 데이터베이스를 생성할 수 없고 기존에 구축된 데이터베이스를 이용하는 것만 가능하다. 그러나 AnyBuilder와 MobileDev와는 달리 HTML 폼의 생성도 가능하다.

AnyBuilder Enterprise, MobileDev, Visual Studio .NET으로 생성한 응용은 모두 2-계층 구조를 가지기 때문에 WAPSiteGen이 생성하는 응

〈표 1〉 상용 WAP 응용 생성기와의 기능 비교

		WAPSiteGen	AnyBuilder Enterprise	MobileDev	Visual Studio .NET
프리젠테이션	한 데크에 표현 가능한 개체	닻 개체 첨부 개체 확장 개체	닻 개체	닻 개체	닻 개체
	데크 생성	자동	반자동	반자동	반자동
비즈니스 로직	컴포넌트 사용유무	O	X	X	X
	생성코드	JSP와 EJB 컴포넌트	ASP, JSP, PHP	ASP, PERL	HTML, WML, cHTML
데이터베이스	데이터베이스 생성	클래스 다이어그램 이용	X	X	X
	기존 데이터베이스 사용	O	O	O	O
전체	생성된 응용의 구조	3-계층	2-계층	2-계층	2-계층
	생성된 응용의 재사용성 · 확장성 · 이식성	+++	+	+	+

용에 비해 프리젠테이션과 비즈니스 로직이 합쳐져 성능 면에서 비효율적이며 메모리가 제한적인 무선 이동 장비에 비즈니스 로직을 담당하는 프로그램이 설치되어야 한다. 또한, 컴포넌트 기반이 아니기 때문에 재사용성, 확장성, 이식성 등이 떨어진다. 상용 도구들이 2-계층 구조의 응용을 생성하는 이유는 시장이 아직 컴포넌트 기술을 기반으로 하는 3-계층 응용을 수용할 만큼 성숙되어 있지 않기 때문일 것이다.

표 1에서 보듯이 WAPSiteGen은 데이터베이스에 대한 비즈니스 로직을 담당하는 EJB 컴포넌트들과 프리젠테이션 로직을 담당하는 고객용 WML 데크 및 관리자용 HTML 폼, 질의를 처리하는 JSP 컴포넌트들을 자동으로 생성한다. 상용 생성기들은 하나의 데크에 닻 개체의 정보만을 담는데 반해 WAPSiteGen이 자동으로 생성한 데크 또는 HTML 폼은 닻 개체뿐만 아니라 첨부 개체와 확장 개체의 정보까지 한번의 질의로 하나의 데크 또는 HTML 폼에 포함할 수 있다. 따라서 닻 개체와 작간접으로 일대일, 다대일, 일대다, 다대다로 연관된 개체들을 한꺼번에 조작할 수 있다는 장점이 있다.

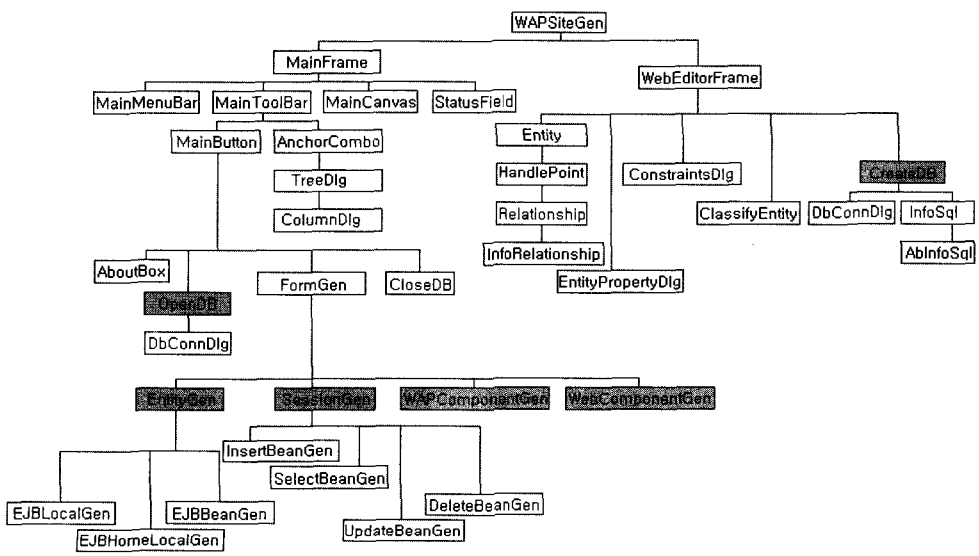
4. WAPSiteGen의 구현

본 절에서는 WAPSiteGen 프로토타입의 구현을 간략히 소개한다. WAPSiteGen은 Java로 구현되었으며 JSP 컴포넌트들과 함께 WebLogic 서버 6.0에 배치 가능한 EJB 2.0 컴포넌트들을 생성한다. 데이터베이스는 ORACLE8i 서버 및 MSSQL 서버 7.0을 사용할 수 있다. WAPSiteGen의 컴포넌트 계층구조 및 각 클래스에 대해 설명한 후, WAPSiteGen이 자동 생성하는 EJB 컴포넌트와 JSP 컴포넌트에 대해 설명한다.

4.1 WAPSiteGen의 컴포넌트 계층구조

그림 7은 WAPSiteGen의 컴포넌트 계층구조이다. 주요 클래스에 대한 설명은 다음과 같다.

WAPSiteGen : 이 클래스는 WAPSiteGen의 MainFrame을 생성하는 메인 메소드를 포함한다. 데이터베이스부터 구축할 것인지 아니면 기존 데이터베이스를 사용할 것인지에 따라 WebEditorFrame 객체가 생성되거나 MainFrame 객체가 생성된다.



〈그림 7〉 WAPSiteGen의 컴포넌트 계층구조

WebEditorFrame : 이 클래스는 JFrame의 서브클래스이다. Entity, EntityPropertyDlg, ConstraintsDlg, CreateDB의 네 개의 컴포넌트들을 포함한다. 클래스 다이어그램을 그릴 수 있는 JPanel 클래스의 drawingPanel 객체를 생성한다. JPanel에 ActionListener를 이용하여 Entity와 Relation을 그리고 각 Entity의 Property 및 무결성 제약조건 등의 정보를 입력 받기 위해 Entity, Relationship, EntityPropertyDlg, ConstraintsDlg 등의 객체를 사용한다.

Entity : 데이터베이스의 시스템 카탈로그 테이블에 있는 스키마 정보를 테이블 단위로 저장한다. entityName, primaryKey, foreignKey, foreignKey_table, atts, appendEntities, expandEntities 등을 속성으로 갖는다.

CreateDB : 이 클래스는 JFrame의 서브클래스이다. MSSQL 7.0이나 ORACLE8i 데이터베이스 중 하나를 선택하게 하고, DbConnDlg 컴포넌트로부터 구축할 데이터베이스의 정보를 입력 받은 후, 클래스다이어그램을 그릴 때 생성된 Entity, Relationship, ConstraintDlg 등의 객체를 이용하여 데이터베이스를 구축한다.

DbConnDlg : 이 클래스는 JDialog의 서브클래스이다. 구축할 데이터베이스에 대한 IP 주소, JDBC 포트, 데이터베이스의 사용자 아이디, 패스워드 정보를 입력 받는다.

MainFrame : 이 클래스는 JFrame의 서브클래스이다. 기존의 데이터베이스를 이용하여 웹 응용을 생성하기 위한 MainMenuBar, MainToolBar, MainCanvas, StatusField 등의 네 개의 컴포넌트들을 포함한다.

MainMenuBar : 이 클래스는 MenuBar의 서브클래스이다. 메뉴 항목으로는 File, Select, DataBase, GenForm, Help를 포함한다. File 메뉴는 Open, Save, Exit 명령을 갖는다. Select 메뉴는 Class Diagram과 Database 항목이 있으며 데이터베이스부터 구축할 것인지 또는 기존 데이터베이스를 사용할 것인지에 따라 사용자가 필요한 선택을 할 수 있도록 한다. DataBase 항목은 OpenDB와 CloseDB 항목을 갖는다. GenForm 항목은 GenEJB, GenForm 항목을 갖는다.

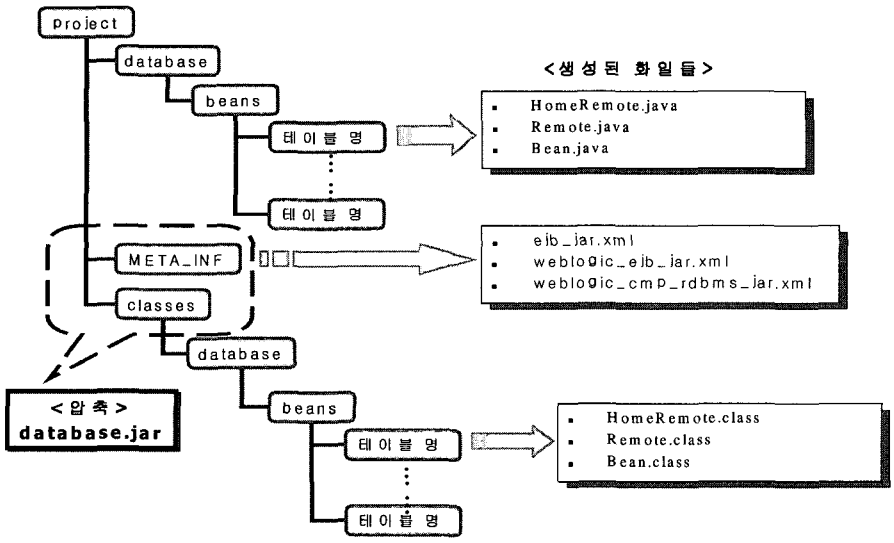
MainToolBar : 이 클래스는 JFrame의 서브클래스이다. AnchorCombo와 MainButtons의 두 컴포넌트들을 포함한다. 이 컴포넌트들은 MainToolBar의 JPanel에 의해 붙여지며 MainMenuBar와 같은 명령을 수행할 아이콘들로 구성된다.

OpenDB : 이 클래스는 JFrame의 서브클래스이다. 데이터베이스를 선택하게 하고, DbConnDlg 컴포넌트에서 입력 받은 정보들을 이용하여 데이터베이스에 연결한다. 데이터베이스의 시스템 카탈로그 테이블에 있는 스키마 정보는 테이블 단위로 Entity 객체들에 저장된다. Entity 객체에는 테이블명, 각 테이블의 속성들, 그 속성의 타입들, 기본 키, 외래 키가 저장되며 또한 Statement 객체의 executeQuery() 메소드를 사용하여 ALL_CONSTRAINS 테이블로부터 검색된 각 테이블과 첨부 및 확장 관계에 있는 테이블명들이 저장된다.

AnchorCombo : 데이터베이스와 연결 후, 콤보박스에 테이블명이 설정된다. OpenDB 객체에서 executeQuery() 메소드를 사용하여 USER_TABLES 테이블로부터 검색된 테이블명은 AnchorCombo 객체의 JComboBox에 첨부된다. 첨부된 테이블명의 리스트에서 사용자가 닻 개체를 선택하게 된다.

TreeDlg : 이 클래스는 JDialog의 서브클래스이다. 그림 3의 TreeDlg와 같은 Dialog 박스를 생성한다. TreeDlg는 닻 개체의 이름을 보여주는 테이블을 포함한다. OpenDB에서 생성된 Entity 객체들의 getAppendEntities()와 getExpandEntities() 메소드들을 이용하여 닻 개체와 첨부 및 확장 관계에 있는 테이블들을 트리 구조로 나타낸다.

ColumnDlg : 이 클래스는 JDialog의 서브클래스이다. 그림 3의 ColumnDlg와 같은 Dialog



〈그림 8〉 WAPSiteGen이 생성하는 EJB 파일들

박스를 생성한다. ColumnDlg에는 TreeDlg에서 선택된 테이블의 속성들을 나타낸다. 사용자는 폼에 포함시킬 속성들을 선택할 수 있다.

EntityGen : 사용자가 EJBGen 버튼을 클릭 함으로써 이 클래스의 인스턴스가 생성된다. EntityLocalGen, EntityHomeLocalGen, EntityBeanGen 컴포넌트 등의 해당 엔티티 빈들을 생성한다.

SessionGen : 사용자가 EJBGen 버튼을 클릭 함으로써 이 클래스의 인스턴스가 생성된다. InsertBeanGen, SelectBeanGen, UpdateBeanGen, DeleteBeanGen 컴포넌트 등의 해당 세션 빈들을 생성한다.

WAPComponentGen : 사용자가 WAPGen 버튼을 클릭 함으로써 이 클래스의 인스턴스가 생성된다. Insert/Search 데크, Select 데크, Update/Delete 데크, Result 데크 등의 네 개의 데크를 자동 생성한다. WAPSiteGen이 생성한 EJB 컴포넌트들은 이 WML 데크들을 통해 이루어질 질의를 처리한다.

WebComponentGen : 사용자가 WebGen 버튼을 클릭 함으로써 이 클래스의 인스턴스가 생성된다. Insert/Search 폼, Select 폼, Update/Delete 폼, Push 폼, Result 폼 등의 다섯 개의 HTML 폼을 자동 생성한다. WAPSiteGen이 생성한 EJB 컴포넌트들은 이 HTML 폼들을 통해 이루어질 질의를 처리한다.

4.2 EJB 컴포넌트의 자동 생성

그림 8은 WAPSiteGen이 EJB 컴포넌트들을 위해 생성하는 파일들과 이 파일들이 저장될 폴더의 구조이다. 생성되는 EJB 컴포넌트들은 버전 2.0인 엔티티와 세션 빈들이다. EJB 컴포넌트들을 상용 제품인 WebLogic 서버 6.0에 배치하기 위한 디플로이먼트 디스크립터(deployment descriptor)도 생성한다. 하나의 EJB 컴포넌트인 엔티티 빈 또는 세션 빈에 대해 두 개의 인터페이스와 하나의 빈 클래스를 생성한다. 각 개체 타입의 개수만큼 엔티티 빈이 생성되며 세션 빈으로는 데이터베이스에 대한 질의처리를 수행하기 위한 InsertBean, SelectBean, UpdateBean, DeleteBean을 생성한다.

WAPSiteGen이 EJB 컴포넌트들을 위해 생성하는 파일들은 project/database/beans 폴더 밑에 저장된다. Titan 응용인 경우, beans 폴더 밑에 custom

```
<%@ page contentType = "text/html; charset=euc-kr" %>
<%@ page import="java.sql.*, java.util.*, java.util.Iterator, javax.transaction.*, javax.rmi.*, javax.naming.*" %>
<%@ page import="java.text.*" %>
<%@ page import="database.beans.CustomerSelect.*" %>
<%@ page import="javax.ejb.EJBException,javax.rmi.PortableRemoteObject,java.rmi.RemoteException" %>

<%--Select Data Relation Query--%>
<%
String s = request.getParameter("selectNum");  //anchor

Context jndiContext = null;
Object ref = null;
Vector selectV = null;
Vector anchorV = null;
Vector appendSetV = new Vector();
Vector expandSetV = new Vector();

try {
        jndiContext = new InitialContext();
        ref = (CustomerSelectHomeRemote)jndiContext.lookup("CustomerSelectHomeRemote");

        CustomerSelectHomeRemote anchorSelectBean =
 (CustomerSelectHomeRemote)PortableRemoteObject.narrow(ref, CustomerSelectHomeRemote.class);
        CustomerSelectRemote select= (CustomerSelectRemote) anchorSelectBean.create();

        selectV = (Vector) select.select(s);

        anchorV = (Vector)((Vector)selectV.elementAt(0)).elementAt(0); // Vector안의 Vector이므로..
        appendSetV = (Vector)selectV.elementAt(1);
} catch (Exception e) {
        e.printStackTrace();
}
%>

<%@ page contentType="text/vnd.wap.wml"%>
<?xml version="1.0" encoding="ks_c_5601-1987"?>
<!DOCTYPE wml PUBLIC "-//PHONE.COM//DTD WML 1.1//EN" "http://www.phone.com/dtd/wml11.dtd">

<%-- cache delete --%>
<%
response.setHeader("Pragma", "No-cache");
response.setDateHeader("Expires", 0);
response.setHeader("Cache-Control", "No-cache");
%>

<html xmlns="http://www.w3.org/1999/xhtml"xml:lang="en">

        <card id="Select" title="Select Table">
        <p>Select Table</p>
        <ol>
                <li><a accesskey="1" href="#CUSTOMER">CUSTOMER</a></li>
                <li><a accesskey="2" href="#ADDRESS">ADDRESS</a></li>
                <li><a accesskey="3" href="#CREDIT_CARD">CREDIT_CARD</a></li>
                <li><a accesskey="4" href="#RESERVATION">RESERVATION</a></li>
                <li><a accesskey="5" href="#CRUISE">CRUISE</a></li>
                <li><a accesskey="6" href="#SHIP">SHIP</a></li>
                <li><a accesskey="7" href="#PHONE">PHONE</a></li>
        </ol>
        </card>
        :
        :
```

〈그림 9〉 CUSTOMER UpdateDelete.jsp

er, credit_card, address, phone, ship, cruise, reservation, cabin 등의 서브폴더가 생성되고 각 폴더에 해당 엔티티 빈에 대한 두 개의 인터페이스와 하나의 빈 클래스인 HomeRemote.java, Remote.java, Bean.java 파일들이 생성되어 저장된다. 생성된 엔티티 빈들은 Titan 응용 데이터베이스 내의 모든 개체 타입에 대한 데이터 로직을 담당한다. Customer를 닻 개체로 선택하였을 때, 세션 빈들에 대해서는 beans 폴더 밑에 CustomerInsert, CustomerSelect, CustomerUpdate, CustomerDelete 등의 서브폴더가 생성되고 각 폴더에 Bean.java, HomeRemote.java, Remote.java 파일들이 생성되어 저장된다. 세션 빈은 데이터베이스에 대한 질의 처리를 수행하기 위한 비즈니스 로직을 담당한다.

이렇게 생성된 EJB 컴포넌트들을 상용 제품인 WebLogic 서버 6.0에 배치하기 위한 디플로이먼트 디스크립터도 project 폴더 밑의 META-INF 서브폴더에 생성된다. 생성된 빈들은 컴파일 되어 디플로이먼트 디스크립터에 의해 디플로이 되면서 분산 객체 간의 통신을 위한 스텁(stub)과 스켈레톤(skeleton)이 생성된다. 컴파일 된 클래스 파일, 디플로이먼트 디스크립터, 스텁, 그리고 스켈레톤은 "database.jar"라는 파일로 압축된다. 압축된 파일은 project 폴더 밑에 저장되며 바로 WebLogic 서버 6.0에서 사용이 가능하다.

4.3 JSP 컴포넌트의 자동 생성

WAPSiteGen은 WML 데크 또는 HTML 폼으로부터의 질의 처리를 위해, 생성되는 폼의 구조에 맞게 Insert.jsp, Select.jsp, UpdateDelete.jsp, Result.jsp의 4가지의 JSP 웹 컴포넌트들을 생성한다. 이들은 세션 빈들과 엔티티 빈들을 이용해 질의를 처리한다. 예를 들어, 데이터베이스에 데이터를 삽입하기 위한 질의를 처리하는 Insert.jsp는 세션 빈인 InsertBean을 호출하여 질의를 처리하고 처리된 결과를 WML 데크 또는 HTML 폼으로 전송하여 화면에 보일 수 있도록 한다.

그림 9는 Titan 응용에서 Customer를 닻 개체로 선택하였을 때, 닻 개체와 첨부 및 확장 관계에 있는 개체 타입들을 세션 빈인 CustomerSelect를 이용해 검색하고 그 결과를 Update/Delete 데크에 보여주기 위해 생성된 JSP 컴포넌트의 일부분이다. 먼저, JNDI를 이용해 CustomerSelect 세션 빈이 위치한 주소를 찾아 홈 오브젝트의 참조를 획득한 후, CustomerSelect 세션 빈의 새로운 홈 인터페이스 객체인 home을 생성한다. 생성된 홈 인터페이스 객체를 이용하여 JSP 컴포넌트가 비즈니스 메소드인 select()를 호출할 수 있도록 하기 위해 EJB 객체인 Select를 생성하고 이 객체에 대한 스텁을 획득한다. JSP 컴포넌트는 닻 개체에 대한 검색을 수행하기 위해 스텁을 통해 CustomerSelect 세션 빈의 select() 메소드를 호출한다. select() 메소드는 닻 개체의 기본 키를 파라미터로 갖는다. 메소드 수행 결과는 벡터 타입으로 받아 WML 데크를 동적으로 작성 한 후, 사용자 측에 전송되어 보여지게 된다.

5. 결 론

무선 인터넷 사용자의 급증과 함께 모바일 전자상거래가 활성화 되고 있다. 이러한 모바일 전자상거래에 대한 수요를 만족시키기 위해, 모바일 전자상거래를 지원하는 모바일 응용의 생산성 향상이 요구되고 있다. 본 논문에서는 컴포넌트 기반이면서 3-계층 구조를 갖는 WAP 데이터베이스 응용을 자동 생성하여 모바일 응용의 생산성 향상을 지원하는 WAPSiteGen을 소개하였다. WAPSiteGen은 다음과 같은 소프트웨어 컴포넌트들을 자동 생성한다.

1) 3-계층 중, 비즈니스 로직을 담당하는 EJB 컴포넌트들과 함께 질의 처리 및 푸쉬 서비스 기능을 제공하는 JSP 웹 컴포넌트들
2) 3-계층 중, 프리젠테이션 층으로는 질의 결과를 보여주는 관리자용 HTML 폼(유선용)과 고객용 WML 데크(무선용)
3) 3-계층 중, 데이터베이스 층은 클래스 다이

어그램을 이용해 새롭게 구축하거나 기존 데이터베이스를 사용

표 1에 상용 WAP 응용 생성기들과의 기능을 비교를 하였으며 WAPSiteGen의 주된 비교 우위 사항들을 요약하면 다음과 같다.

1) WAPSiteGen이 자동 생성한 프리젠테이션 층(HTML 폼 및 WML 데크)은 닷 개체뿐만 아니라 이와 연관된 첨부 개체 및 확장 개체들을 포함한다. 따라서 한번의 질의로 연관된 개체들을 한번에 검색할 수 있다.
2) WAPSiteGen이 생성한 응용은 3-계층 구조를 가지며 컴포넌트 기반이기 때문에 확장성 및 재사용성이 우수하다.

참 고 문 헌

[1] Chi-Wei Lan, Chun-Chou Chien, Meng-Yen Hsieh, Irene Chen, "A Mobile e-Commerce Solution", Proceedings of the International Symposium on Multimedia Software Engineering, p. 215-222, 2000.

[2] Doohun Eum and Toshimi Minoura, Web-SiteGen : Web-Based Database Application Generator, IEICE Trans. on Information & Systems, Vol. E86-D, No.6, pp. 1001-1010, Jun. 2003.

[3] 권오성, Converter : Integration between wired Internet and Wireless Internet, http://matilda.snu.ac.kr/doc/analysis/wap_converter.htm

[4] Wireless Markup Language, WAP Forum, Apr. 30. 1998. http://www.wapforum.org

[5] 정화영 외1, "Implementation and Development of Seat Reservation System based on EJB for E-Business", 『정보처리학회 2002년 추계학술대회』, VOL. 09, NO. 02, 2002.10.

[6] 강이지 박은희 음두헌, MAG : WAP 푸쉬 서비스를 제공하는 모바일 응용 생성기, 한국정보과학회 춘계 학술발표회 제31권 1호 (B), 2004.4. pp. 463-465

[7] WAP, Bluetooth, and 3G Programming: Cracking the Code, Dreamtech Software Team, 2002, Dreamtech Software Team

[8] Visual Studio .NET, http://www.microsoft.com/korea/msdn/vstudio/.

[9] AnyBuilder Enterprise, http://www.anybil.com.

[10] MobileDev, http://mobiledev.speedware.com.

[11] Richard Monson-Haefel, Enterprise JavaBeans, O'REILLY, 2001. 10.

[12] ORACLE: Oracle Transparent Gateway for DB2 Installation and User's Guide, http://technet.oracle.co.kr/docs/oracle78/gateways/tg4db2/toc.htm.

[13] MSSQL: MS SQL 2000 Books Online, http://ddart.net/mssql/sql2000/html/.

[14] 신동규·신동일·차석일·장철수·이경호·김중배, "EJB기반 모바일 전자상거래 컴포넌트의 설계 및 구현", 『정보처리학회논문지D』, Vol.9-D, No.4, 2002.8.

◐ 저 자 소 개 ◑

음 두 헌 (Doohun Eum)
1984년 서강대학교 전자공학과 졸업(학사)
1987년 오레곤주립대학교 대학원 컴퓨터공학과 졸업(석사)
1990년 오레곤주립대학교 대학원 컴퓨터공학과 졸업(박사)
1991년 ~ 1992년 전자통신연구원 인공지는연구실 선임연구원
1992년 ~ 현재 덕성여자대학교 컴퓨터공학부 교수
관심분야 : 객체지향 시스템, 분산 및 모바일 응용, etc.
E-mail : dheum@duksung.ac.kr

강 이 지 (Izzy Kang)
2003년 덕성여자대학교 수학과 졸업(학사)
2005년 덕성여자대학교 대학원 전산 · 정보통신학과 졸업(석사)
2005년 ~ 현재 아주대학교 중앙전산원
관심분야 : 객체지향시스템, 데이터베이스, etc.
E-mail : izzy@duksung.ac.kr