Class Diagram 의 Class 를 EJB Bean 으로의 Mapping 기법

°허진선, 김수동
숭실대학교 컴퓨터학과

A Technique for Mapping Classes to EJB Beans

°Jin-Sun Her, Soo Dong Kim
Dept. of Computing, Soongsil University

요 약

소프트웨어 산업계에서 재사용 단위가 객체보다 더 큰 컴포넌트 기반의 개발에 관심이 집중되고 있다. 그래서 모델링 언어인 UML과 컴포넌트가 운용되는 유연하고 확장성 높은 기반 아키텍처인 EJB를 이용한 기업형 시스템 개발이 요즘 기업에서 활발해지고 있다. UML과 EJB 각각에 대한 연구는 많이 진행되었지만, UML Model을 이용한 EJB Model구현시의 mapping기법에 관한 연구는 아직 미흡한 실정이다. 그래서 본 논문에서는 UML Modeling 을 통해 Class diagram에서 추출된 Class 들이 EJB로 구현될 때 실제로 어떤 Bean으로 Mapping 되는지에 대해 제시한다.

1. 서론

컴포넌트 기반 소프트웨어 개발 기법은 고품질의 소프트웨어를 경제적으로 개발하는 새로운 기법으로 부각되고 있다. Sun사에 의해 제안된 Enterprise JavaBeans(EJB) 아키텍처는 컴포넌트 기반의 분산 업무 어플리케이션의 개발과 배치를 위한 컴포넌트 아키텍처이다. EJB를 사용함으로써, 복잡한 프레임워크에 대한 작성 없이 확장성과 신뢰성이 높고 안전한 어플리케이션의 작성이 가능하게 된다. EJB는 어떤 회사의 엔터프라이즈 미들웨어에서도 이동성 있고 재사용이 가능한 어플리케이션을 지원하도록 설계되었다. 컴포넌트 기반의 소프트웨어 개발 프로젝트의 핵심사항은 프로젝트에 적합한 컴포넌트 모델의 선택이다. 컴포넌트 모델은 소프트웨어 개발자로 하여금 하위 레벨의 시스템 프로그래밍이나 내부 구조보다는 컴포넌트에 관심을 갖도록 하는 기술과 규칙의 집합이다. 그러나, 컴포넌트 모델 중 하나인 EJB를 위한 설계기법이나 방법론은 소개되어있지 않다. 그래서, 프로젝트 진행 시 클래스를 빈으로 매핑하는 데에 어려움이 있다. 본 논문에서는 클래스를 빈으로 매핑하는 기법을 제안하고자 한다[1].

본 논문의 2장에서는 관련연구로서 클래스 타입과 빈 타입에 대해서 살펴본다. 3장에서는 MVC stereotype 모델링 과정에 대해서 설명하고, 4장에서는 MVC Class에서 EJB Bean으로 기법에 대해서 설명한다. 마지막으로 5장에서는 결론을 내리고 향후 연구 방향을 제시한다.

2. 관련연구

2.1 Class Type

MVC라는 개념은 Smalltalk에서 소프트웨어 계층을 구분하는 기준으로 소개되었다. 즉, 클래스를 [그림 1]처럼 Model, View, Controller로 분류한다[2,3].

- Model : MVC에서 가장 하위 계층에 있으며, 어플리케이션의 영구적 데이터를 가진 객체를 가리킨다. Model은 데이터의 값을 가지고 있다. 이 데이터는 Controller 객체가 접근 사용한다.
- Controller : Controller는 Model과 View사이에서 위치하며, Model 데이터를 처리하는 업무 로직을 구현한 객체 계층이다. 사용자의 입력에 대해 사용자 인터페이스(UI)가 어떻게 반응할 지를 정의하고 있다.
- View : Controller를 통하여 처리 결과를 사용자에게 보여 주기나, 사용자의 입력을 받아 하부에 전달하는 객체 계층이다. 하부 계층 객체(Model)의 데이터를 여러 가지 방법으로 사용자에게 보여줄 수 있다.

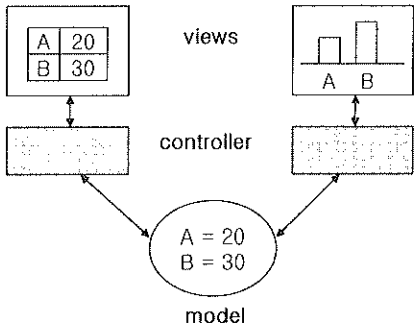

그림 1. MVC

2.2 Bean Type

EJB 에서의 빈은 기본적인 컴포넌트 단위로 정의하고 있다. 빈에는 두 종류가 있는데 세션 빈(Session Bean)과 엔티티 빈(Entity Bean)이 그것이다.

Session Bean은 자료의 지속성이 일시적인 성격을 지니는데, 이는 Entity Bean으로 처리 되지 않는 부분들을 채워주는 역할을 하며, 서로 다른 Bean 간의 상호작용, 즉 workflow를 표현해 준다. 세션 빈은 두 가지의 기본 형태, 무상태(Stateless) 세션 빈과 상태유지(Stateful)세션 빈으로 나뉘어 진다.

- 무상태 세션 빈(SLS) : 무상태 세션 빈은 메소드로 대표되는 관련된 서비스들의 집합으로, 빈은 한 번의 메소드 호출이 일어난 후 다음 번 호출까지 상태를 유지하지 않는다.

- 상태유지 세션 빈(SFS) : 상태 유지 세션 빈은 클라이언트 어플리케이션의 연장이다. 클라이언트 대신 작업을 수행하고, 그 클라이언트와 관련된 상태를 유지한다. 상태유지 세션 빈과 클라이언트 간의 지속적인 대화를 나타낸다고 하여 이 상태를 대화 상태(Conversational State)라고 한다. 상태유지 세션 빈에 호출된 메소드들은 이 대화 상태로부터 데이터를 읽고, 쓸 수 있는데 이 대화 상태는 해당 빈의 모든 메소드들에 의해서 공유된다.

세션 빈에 비해 엔티티 빈은 보존의 성격을 가진다. 엔티티 빈은 데이터베이스의 데이터를 표현하고 데이터베이스를 기반으로 한 데이터를 갱신하고, 트랜잭션에 관여하여 동시에 여러 사용자들에게 사용될 수 있다. 자료의 지속성을 보장하는 방법에는 빈 관리 지속성(Bean-Managed Persistence)과 컨테이너 관리 지속성(Container-Managed Persistence)으로 나뉠 수 있다.

- 빈 관리 지속성(BMP) : 빈 개발자가 직접 소스코드 상에서 데이터베이스를 접근하여 자료의 지속성을 보장하며, 빈 인스턴스와 데이터베이스 사이에서 상태를 관리함에 있어 유연성을 준다.

- 컨테이너 관리 지속성(CMP) : 컨테이너에서 프라이머리 키(Primary Key) 클래스를 이용하여 자동적으로 관리해 줌으로서, 빈 개발자가 별도의 지속성 관리를 위한 코딩을 해주지 않아도 된다 [4,5].

3. MVC Stereotype 모델링 과정

어플리케이션 구축 시 도메인의 요구사항에 따른 분석으로 시작한다. UML을 이용한 분석단계에서 나온 Conceptual Class Diagram(CD)에다 각 클래스의 특성에 따라 MVC 타입으로 분류하여 Stereotype을 추가한다. 표기는 [그림 2]과 같이 한다.

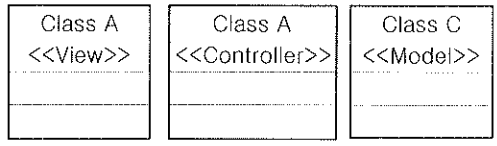

그림 2. MVC Stereotype을 추가한 CD

지속적인 속성(attribute)이 있는 클래스는 <<Model>>로 표기한다. 한 도메인에서 분석된 클래스가 지니는 속성 중 대부분이 이런한 속성을 지니는 경우가 많다. 객체를 실체화 하면서 자신의 속성을 지니게 되며 이러한 속성을 장기적인 목적을 위해서 데이터베이스나 파일 시스템을 이용하여 보조기억장치에 보존되는 경우이다. 객체가 없어지더라도 기억된 속성들은 다시 복구되어 이용될 수 있다.

둘째, 영구적인 속성이 하나도 없는 클래스는 <<View>>로 표기한다. 또한, 클라이언트나 actor가 initiate하는 클래스들도 여기에 포함된다. 어플리케이션 구현 시 사용자 인터페이스에 해당하는 부분이 여기로 매핑된다. 셋째로, Model 클래스를 호출하고 View 클래스로부터 호출받는 클래스는 <<Controller>>로 표기한다. 또한, 구현하고자 하는 시스템의 기능이 포함되어 있는 비즈니스 로직이 들어 있는 경우나 계산이 들어있는 클래스도 여기로 매핑된다.

4. MVC class 에서 EJB Bean 으로의 Mapping

3장의 MVC Type을 추가한 Class Diagram에서 EJB 구현을 고려한 한 단계 상세한 Class Diagram을 [그림 3]과 같이 작성한다. <<View>> Class는 대부분 Applet, JSP나 Servlet 형태로 구현되며, 사용자와 서버 사이에서 정보 교류 역할을 담당한다. 예를 들면, Servlet으로 View를 구현하며 웹 브라우저 상의 입.출력을 Servlet의 Get, Set 함수를 통하여 서버의 Session Bean 이나 Entity Bean과 정보 교류를 하게 된다.

<<Controller>> class는 Stateless Session Bean이나 Stateful Session Bean으로 Mapping된다. 이는 EJB의 Session Bean이 영구적 데이터를 직접 가지고 있지 않고 Entity Bean으로부터 받은 데이터를 처리하는 업무 로직을 가지고 있어, Controller 역할과 일치한다.

Stateless 와 Stateful Session Bean 의 결정은 그 Controller 클래스에 해당되는 Use Case 명세(Description)을 보고 판단할 수 있다. 즉, 해당 클래스에 적용되는 기능들이 사용자와의 세션 과정과 중

간 값, 즉 상태를 유지할 필요가 없을 경우 Stateless Bean 을 사용한다. 사용자와의 세션을 유지, 기억해야 하는 기능들인 경우 Stateful Bean 으로 구현한다. Stateless Bean 이 실행 시 더 가볍고 효율적이므로, 가능한 Stateless Bean 으로 구현을 시도한다.

해당 Use Case 명세가 상세하지 않거나 부정확한 경우, 상태전이도 (Statechart Diagram)을 작성하여, 상태 유지의 필요성을 판단한다. 즉, 상태전이도에서 초기 상태와 최종 상태 사이에 유지해야 하는 객체 고유의 상태가 발견되면 Stateful 로 구현한다. Stateless 로 구현되는 클래스의 상태전이도는 중간 상태가 없거나 하나 정도 발견되므로, 쉽게 판단할 수 있다.

<<Model>>class 는 EJB 의 영구 데이터를 가지고 있는 Entity Bean 으로 Mapping 된다. 즉, Model 클래스의 영구적 데이터 속성들을 Entity Bean 안에 정의하며, JDBC 를 통하여 관계형 데이터베이스 테이블에 저장한다. Entity Bean 과 테이블 사이의 저장을 위하여 EJB 에서는 Bean-Managed Persistence(BMP)와 Container-Managed Persistence(CMP) 2 가지 방식을 제공한다.

한 Entity Bean 에 적용되는 Home Interface 의 find 함수들의 유형을 살펴보면, BMP 와 CMP 유형을 구분할 수 있다. 즉, 데이터베이스 함수들이 하나의 Bean 안의 여러 인스턴스들 간의 질의가 대부분인 경우는 CMP 로 구현한다. CMP 의 경우는 SQL 질의의를 프로그램 안에 포함하지 않고, Descriptor 라는 외부 설정자에 정의하여 사용함으로 Entity Bean 구현이 용이하며, 질의의 내용이 수정될 경우, Bean 을 수정하지 않고 외부의 Descriptor 만 수정하면 되므로 이식성과 유지보수성이 우수하다.

그러나, 데이터베이스 함수들이 복수 개의 Bean 들에 관여되는 Join 성격의 기능을 필요로 하면, BMP 방식으로 Entity Bean 을 구현한다.

BMP entity Bean 이나 CMP entity Bean 으로 mapping 된다. CMP 가 더 좋으나, class 를 여러 개 join 해야 할 경우 BMP 로 mapping 한다. 이는 복수 Bean 들에게 적용되는 *SQL* 질의의를 *BMP* 프로그램안에 포함할 수 있기 때문이다. BMP 로 구현할 경우의 테이블과의 저장 과정을 직접 통제하고 검사할 수 있기 때문에 보다 상세한 테이블 매핑이 요구될 경우 BMP 로 Model 클래스를 구현한다[1].

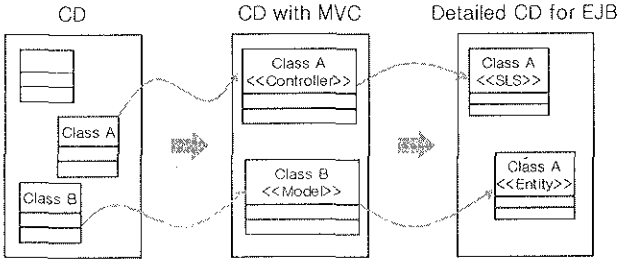

그림 3. Class Diagram 설계 단계

5. 결론

본 논문에서는 개략적인 Class Diagram 으로부터 EJB 구현 직전의 Class Diagram 작성 과정을 통해서 Class 를 Bean 으로 매핑 하는 기법에 대해서 설명하였다. 개략적인 CD 에서 어떤 클래스들이 M,V,C 타입이 되는지 판단해서 MVC Stereotype 을 추가하는 과정을 제안했다. 그리고, 이 MVC Class Diagram 의 View 타입 클래스는 서비사이드 프로그램인 EJB 에서 특별히 매핑 되는 빈은 없고 일부만이 SLS 빈으로, Model 타입 클래스는 Entity Bean 으로, Controller 타입 클래스는 Use Case 명세를 참조한 특성 판단으로 SLS 빈이나 SFS 빈으로 매핑 됨을 살펴보았다. 지금까지 Class 를 Bean 으로 매핑 하는 기법에 대한 수요가 높은 만큼 연구나 지침들이 논의된 적은 있었지만 어플리케이션 구현 시 실무적인 가이드가 될만한 지침이 부족했다. 본 논문에서 제시하는 매핑 기법이 보다 실무적인 적용이 가능할 것으로 기대된다.

참 고 문 헌

[1] 박종성, 객체를 EJB 기반 엔터프라이즈 빈으로의 매핑 기법, 숭실대학교 석사학위 논문, 2000 년 12 월.

[2] Erich Gamma, Richard Helm, Ralph Johnson, John Vlissides, Design Patterns, Addison-Wesley Press, 1995.

[3] Ivar Jacobson, Object-Oriented Software Engineering, Addison-Wesley Press, 1992.

[4] Enterprise JavaBeans™ Specification, Version 1.1, Sun Microsystems, 1999.

[5] 김수동, "Enterprise JavaBeans(EJB) 기반의 컴포넌트 프로그래밍", 정보처리학회 논문지 Vol.4 No7, 2000 년 7 월.