순차도에서 EJB 빈 으로의 매핑 기법

°이원규, 김수동
숭실대학교 컴퓨터학과
wklee@selab.soongsil.ac.kr, sdkim@comp.ssu.ac.kr

A Technique for Mapping Sequence Diagram to EJB Beans

Won-Kyu Lee°, Soo Dong Kim
Dept. of Computing, Soongsil University

요 약

Unified Modeling Language(UML) 은 개발하고자 하는 시스템을 Visualizing, Specifying, Constructing, Documenting 화 해주는 graphical language 이다. 우리는 이러한 모델링 언어를 통하여, 객체지향 분석/설계 과정에 이용하고 있다. 그러나, EJB 로 Mapping 하기 위한 충분한 연구가 이루어지지 않아 개발자들이 EJB 로 UML 을 설계 및 구현을 명확하게 옮기지 못하였다. 본 논문에서는 이 점에 중점을 두고 UML Modeling 을 통해 설계 및 구현된 내용중 UML 의 Sequence Diagram 의 요소들을 EJB Bean 으로의 구현시 Mapping 지침을 제시한다.

1. 서론

UML 기법은 기존의 모델링 기법인 Booch 방법론, OMT, Jacobson 방법론을 집대성하여 향상시킨 객체지향 분석/설계방법이다. UML 기법은 객체지향 모델링이 가져야 할 필수 기능으로 일반화된 표기법을 지원하거나 모델링 요소간의 관계를 명시해 주는 등의 기준을 잘 갖추고 있다. Sequence Diagram(SD)은 객체들 간에 상호작용의 유형을 설명하기 위한 다이어그램이다. Class Diagram 은 시간적인 정보를 전혀 고려하지 않은 반면 SD 는 시간적인 순서로 객체들 사이에 보내지는 메시지의 상호작용을 보여 준다. 객체지향 분석 과정에서의 SD 는 하나의 Use Case 안에서 기능을 수행하기 위해 객체들간에 어떻게 상호작용하는지를 나타낸다. SD 는 흐름에서 객체의 역할을 명확하게 해주고 클래스의 기능과 인터페이스를 결정하기 위한 기본적인 자료를 제공한다. SD 의 이러한 성질이 Enterprize Java Bean(EJB) 의 상태 세션 빈, 무상태 세션 빈, 컨테이너 관리 엔티티 빈, 빈관리 엔티티 빈 중에 어느것으로 Mapping 을 할지 결정하는데 좋은 근거가 되리라 생각이 되어진다.

본 논문의 구성은 다음과 같다. 2 장에서 SD 의 일반적인 특성과 EJB 빈의 유형별 특성을 설명하고, 3 장에서는 SD 을 EJB 빈으로 Mapping 하기 위한 단계를 설명하고, 마지막 4 장에서는 결론을 제시한다.

2. 관련연구

2.1 Sequence Diagram 의 일반적인 특성

어떤 시스템에 속해 있는 한 객체가 다른 객체와 어떻게 교류(Interact) 하는 지를 시간에 걸쳐 나타낼 때 사용한다. 객체는 위부분에 위치하며, 시간은 위에서 아래로 흐른다. 한 객체에서 다른 객체로 전달되는 메시지는 화살표로 나타낸다.

객체는 SD 의 가장 윗부분에 위치하며, 왼쪽에서 오른쪽으로 배열된다. 각 객체로부터 아래로 뻗어 가는 쇄선은 객체의 생명선이라 불린다. 생명선(lifeline)을 따라 좁다란 사각형이 드문드문 나타나는데 이 부분은 실행(activation)이라 하며 객체가 수행하는 오퍼레이션이 실행되고 있음을 나타낸다. 그림 1.은 객체, 생명선, 그리고 실행을 보이고 있다.

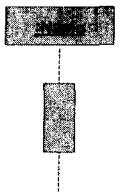

그림 1. SD 의 객체, 생명선, 실행

메시지는 단순(simple) 메시지, 동기(synchronous) 메시지, 비동기(asynchronous) 메시지로 전송될 수 있다. 단순메시지는 한 객체에서 다른 객체로 제어 흐름이 이동하는 것이다. 동기 메시지는 한 객체가 다른 객체로 보내는 메시지 객체로서, 수신 객체로부터 그 메시지를 받았다는 답신이 와야 자신의 작업을 계속할 수 있다는 특성이 있다. 비동기 메시지의 경우, 동기메시지와 달리 그 메시지를 전송한 후에 수신 객체로부터 답을 기다리지 않고 자신의 작업을 계속한다. 그림 2.는 SD 의 메시지 기호이다.

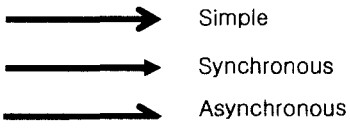

그림 2. SD 의 메시지 기호

2.2 EJB 빈의 특징

EJB 에는 두 종류의 빈이 있다. 첫째, 세션 빈은 상태유지 세션 빈과 무상태 세션 빈으로 구분되고, 둘째, 엔티티 빈은 영속성 관리 방법에 따라서 컨테이너 관리 엔티티 빈과 빈관리 엔티티 빈으로 구분된다. 빈들의 특성을 비교하면 세션빈은 하나의 클라이언트의 작업을 목적으로 하며, 하나의 클라이언트에서만 공유 접근이 가능하다. 상태유지 세션빈의 경우 특정 클라이언트에게 할당되는것에 반해, 무상태 세션빈은 특정 클라이언트에게 공유 접근이 할당되지 않는다. 세션빈은 영속성을 갖지 않는다.
상태유지 세션빈은 클라이언트를 대신하여 워크플로우를 캡슐화하고 관리하는 단순한 인터페이스를 제공하며, 무상태 세션빈은 하나의 매소드로 전체작업을 처리하며 스와핑의 부하를 줄일 수가 있다. 엔티티 빈은 영속성 저장소에 존재하는 객체표시를 목적으로 하며, 하나 이상의 클라이언트에게 공유 접근을 허용한다. 엔티티빈은 세션빈과 달리 영속성을 갖는다.

2.3 MVC 의 개념

MVC 는 객체지향언어인 Smalltalk 에서 도입되었으며 JDK 1.1 에서는 Observe/Observable 객체로 많이 사용이 되었다. Model - View - Controller 구조이며, View 와 Controller 를 합쳐 Delegate 라 표현하기도 한다. 이것은 어떻게 보여지고, 사용자 입력에 어떻게 반응하며 데이터가 어떻게 표현되어지는지를 나타낸다.

(1) Model : 프로그램 상태에 대한 논리적인 표현이다. 데이터가 변경되었을 경우에는 이를 View 에게 통보한다.
(2) View : 변하는 데이터에 대한 시작적인 표현을 제공하며 응답메커니즘으로 Controller 를 사용한다.
(3) Controller : 사용자 입력을 어떻게 다뤄야 할지를 명세한다.

3 Sequence Diagram 과 EJB 빈과의 관계

3.1 객체의 종류

EJB 명세서에는 개체를 빈으로 구현하기 위한 세가지 객체를 표현한다.

(1) 상태가 없이 서비스를 제공하는 객체

(2) 특정 클라이언트와 일정시간 동안 작업을 하는 객체 : 한 클라이언트로부터 여러 번 호출된 메소들간에 상태를 유지한다.

(3) 여러 사용자간에 공유될 수 있는 비즈니스 객체를 표현하는 엔티티 객체

3.2 MVC 개념의 Sequence Diagram 을 EJB 빈으로의 매핑

MVC 중 View 의 관점에 해당되는 Controller 는 EJB 빈으로 구현시 사용자의 UI 부분으로 시각적인 표현을 제공하는 응답 메커니즘이다.

이러한 SD 의 Controller 부분을 EJB 빈으로 매핑할 경우 상태가 없이 서비스를 제공하는 객체와 특정 클라이언트와 일정 시간 동안 작업을 하는 객체로 매핑이 가능하다. 이는 Servlet/JSP 로도 구현이 될 수 있으며, 세션빈으로 매핑이 가능하다.

Controller 의 관점에 해당되는 객체들은 EJB 빈으로 구현시 사용자 입력을 어떻게 다뤄야 할지를 명세한다. 매핑시에는 Description 을 보고 상태유지/무상태 세션빈으로 결정한다.

Model 의 관점에 해당되는 객체들은 EJB 빈으로 구현시 여러 사용자간에 공유될 수 있는 비즈니스 객체를 표현하는 엔티티 객체로 매핑된다. 단일 클래스에 대하여 fi( )인 경우 컨테이너에 의해 영속성이 자동관리되는 컨테이너 관리 엔티티빈으로 매핑이 좋으나, fi( )가 복수개 클래스들간의 쿼리가 필요하는 경우 빈관리 엔티티빈으로 매핑하는 것이 바람직하다.

3.3 MVC 개념의 Sequence Diagram

그림 3.은 MVC 관점에서 살펴본 SD 의 형태를 나타내고 있다. 사용자가 Controller 에게 메시지를 보내면 View Layer 인 Controller 가 그 메시지 f1( )을 Obj:ClassA 에 보내고 있다. 이

때 f1( ) 메시지는 Business Logic Layer 이므로 영속성을 가지지 않는 객체이다. Obj1:ClassA 에서 처리되어진 영속성을 가지지 않는 객체가 또 다시 UI 을 거치지 않고 Obj2:ClassB 에게 메시지 f2( )을 보내고 있다. Obj2:ClassB 에서 처리되어진 메시지는 DB Layer, 즉 영속성을 가지는 객체 Obj3:ClassC 로 메시지 f3( )을 보내고, Obj3:ClassC 에서 처리되어진 메시지는 그값을 View Layer 인 Controller 에게 보낸다. 이는 사용자가 UI 에서 처리되어진 메시지가 내부적으로 처리가 되어지면서 DataBase 에서 처리 된 후 그 처리 결과를 사용자에게 다시 알려주는 것이다.

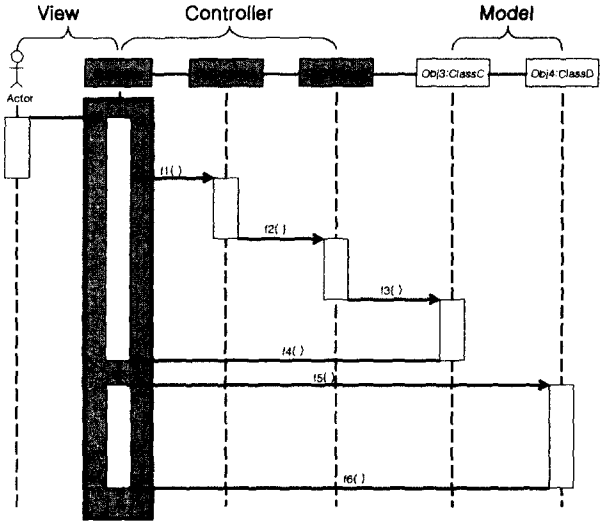

그림 3. MVC 개념의 SD

위의 사용되어진 메시지들은 비동기적이다. 모든 컴포넌트들은 메시지를 보내고 난 후에 답신을 기다리지 않고 바로 자신의 작업을 계속한다. 윈도우 운영체제용 애플리케이션을 몇 개 사용해 보았다면, 아마 비동기 통신이 어떤 것인지도 경험해 보았을 것이다. 문자 키를 누르자 마자 문자가 바로 화면에 나타나는 경우도 있지만, 몇 개의 키를 눌러도 아무 반응이 없다가 눌린 키에 해당하는 문자들이 갑자기 와르르 나오는 경우가 바로 비동기 통신의 예이다.

사용자의 UI 에서 Obj4:ClassD 의 메시지 f5( ) 와 f6( )은 영소성을 갖는 객체로의 메시지이다. UI 에서 DB Layer 로 메시지가 일어나는 것은 UI 부분에서 찾는 과정이 일어난다고 가정하는 것이다.

표 1. MVC, SD, EJB Bean 과의 관계

MVC	SD	EJB Bean
View	Controller	Servlet/JSP Session Bean
Controller	영속성을 갖지 않는 객체	Session Bean
Model	영속성을 갖는 객체	Entity Bean

4. 결론

본 논문에서는 Sequence Diagram 의 요소들을 EJB 로 구현시 매핑되는 지침을 제시하였다. 지금까지의 여러 가지 연구들을 통해서도 UML 의 Diagram 들을 EJB 빈으로 매핑하는 방법에 대해 논의 된바 있으나 본 논문에서 제시하는 Sequence Diagram 을 EJB 빈으로 매핑하는 지침은 이전보다 나은 좀 더 체계적이고 정확한 방법의 검증이 이루어질 것이다.

앞으로는 이러한 방법을 통해서 Diagram 들이 EJB 빈으로 매핑되는 기법이 더욱 상세화 되고 규칙화 되어야 할 것이다.

그러기위해서는 기존의 방법들 뿐만 아니라 더욱더 새롭고 진보된 기술의 개발도 필요로할 것이다.

참 고 문 헌

[1] 김수동, 실무자를 위한 소프트웨어 공학, 에드텍, 1999.

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

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

[4] Grady Booch, James Rumbaugh, Ivar Jacobson, The Unified Modeling Language User Guide, Addison - Wesley, 2000.

[5] Grady Booch, James Rumbaugh, Ivar Jacobson, The Unified Modeling Language Reference Manual, Addison - Wesley, 2000.

[6] Grady Booch, James Rumbaugh, Ivar Jacobson, The Unified Modeling Language Distilled Second Edition, Addison - Wesley, 2000.