컴포넌트 분해에 의한 EJB 빈 추출 기법

허윤호[0] 이연숙 류성열
숭실대학교 컴퓨터학부
{hyh0408, ysbucket}@sclab.ssu.ac.kr

EJB Bean Extracting Method through Component Decompose

Yun-Ho Huh[0] Yeon-Sook Lee Sung-Yul Rhew
School of Computing, Soongsil University

요 약

소프트웨어 산업의 급성장에 의해 소프트웨어의 재사용에 대한 요구는 더욱 증가하고 있다. 현재 객체지향 패러다임보다 더 큰 재사용의 규모를 가지는 컴포넌트 기술이 업계에서 점차 각광을 받고있다. 본 논문에서는 객체지향 언어인 자바언어를 기반으로 개발된 어플리케이션을 자바 분산 컴포넌트 기술인 EJB 기반의 어플리케이션으로 전환하는 기법을 소개하고, 컴포넌트 단위로 구성된 클래스들을 EJB의 빈으로 매핑하는 기법을 제시한다.

1. 서론

최근 몇 년 동안에 자바언어는 다양한 분산 환경에서의 프로그래밍을 수행함에 있어서 그 강력함을 보여주고 있다. 기업의 상당수가 클라이언트/서버 형태의 어플리케이션, 분산 어플리케이션 및 웹 어플리케이션을 자바 기반으로 개발하고있다. 이렇게 개발된 어플리케이션의 대부분이 객체지향 패러다임에 근거한 클래스 단위의 재사용성을 가지고 있지만 소프트웨어 산업이 급속도로 복잡해지고 다양해지고 있기 때문에 클래스 단위의 재사용이 주목할 만한 결과를 보이지 못하고 있다. 그래서 최근 학계에서나 업계에서는 클래스보다 확장된 재사용단위인 컴포넌트에 관심이 모아지고 있다[5]. 최근 자바 언어를 기반으로 하는 분산 환경에서의 프로그래밍은 선(Sun)사에서 제안하는 분산 컴포넌트 아키텍처인 EJB(Enterprise Java Beans)를 이용한 개발이 권장되고 있다. 본 논문에서는 기존의 클래스단위의 재사용성을 가지는 어플리케이션을 EJB의 컴포넌트 단위인 빈(bean) 단위의 재사용성을 가지는 어플리케이션으로 전환하는 기법 중 핵심이 되는 빈 추출 기법을 4단계로 제시한다. 2장에서는 관련 연구로서 EJB에 대해 기술하고, 3장에서는 빈 추출 4단계를 각 단계별로 제시하며, 4장에서는 본 논문에서 제시한 빈 추출 단계를 채팅 시스템에 적용해 본다.

2. 관련연구

2.1 EJB

EJB는 서버 상에서 자바 플랫폼으로 운영되는 컴포넌트 기술로, 객체 컴포넌트 개발에 필요한 스펙들로 구성되어 있다[1]. Enterprise Bean은 Session Bean과 Entity Bean 두 가지로 나뉘어 진다. Entity Bean은 데이터베이스와 연결되어 지속성 있는 데이터를 표현한다. Session Bean은 비즈니스 프로세스를 위한 로직을 포함하며 Entity Bean과는 달리 데이터에 엑세스하는 역할을 한다. Session Bean은 두 개의 서브타입 Stateful Session Bean과 Stateless Session Bean으로 나뉜다[2].

3. EJB 빈 추출

그림 1은 기존의 어플리케이션을 EJB기반의 어플리케이션으로 전환하는 순서를 제시한 프로세스이다.

그림 1 어플리케이션 전환 프로세스

그림 1에서 컴포넌트 식별이 수행된 후의 산출물로서 잘 정의된 인터페이스와 컴포넌트 명세서를 얻게 된다. 이것을 기반으로 각각의 컴포넌트는 빈을 추출하게 되는데 본 논문에서 제시하고 있는 EJB 빈 추출 기법은 다음 그림과 같다.

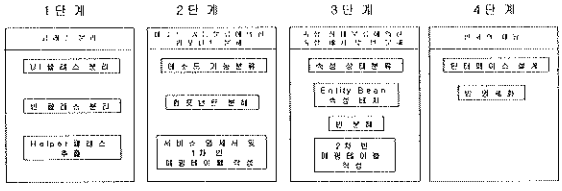


그림 2 빈 추출 기법 4단계

일반적인 재사용성의 목적에 의해 클래스 단위로 분류된 컴포넌트는 EJB의 빈과 직접 매핑이 되지 않는다. 그것은 기존의 어플리케이션이 EJB기반의 어플리케이션을 목적으로 설계되어진 것이 아니기 때문에 식별된 컴포넌트는 대개 두 종류의 빈(Session/Entity Bean)의 성격을 모두 포함하게 된다. 또한 어떤 하나의 클래스 조차도 두 종류의 빈의 성격을 모두 갖게 되는 경우가 있다. 그러므로 성격에 맞는 빈과의 매핑을 위해서는 클래스 보다 하위 단위인 메소드 단위의 분석에 의해 식별된 컴포넌트를 분해할 필요성이 있다.

3.1 클래스 분리

이 단계에서는 컴포넌트를 분해하기 전 재사용 가능한 클래스와 분해 될 클래스로 분리한다.

- **UI 클래스(재사용 가능한 클래스)** : UI를 위한 메소드와 속성들로만 구성되어있는 클래스.
- **빈 클래스(분해 될 클래스)** : EJB의 빈과 매핑될 메소드와 속성들로 구성되어있는 클래스이며,

다음과 같은 특성을 가진다.

- DB와 트랜잭션이 이루어지는 클래스
- 컴포넌트 식별 프로세스에서 정의된 인터페이스와 직접 상호작용을 하는 클래스
- UI 클래스와 직접 상호작용을 하는 클래스.

● Helper 클래스(재사용 가능한 클래스) : 빈 클래스를 도와주는 클래스로서 UI 클래스와 빈 클래스 외에 빈 클래스에서의 일방적인 관계를 이루는 클래스 또는 Helper 클래스와 관계를 이루는 클래스를 말한다.

그 밖의 비즈니스 로직을 포함하는 UI 기능을 가진 클래스 또는 빈 클래스로의 관계를 이루는 클래스들은 빈 클래스에 속한다. 다음은 위의 설명을 그림으로 나타낸 것이다.

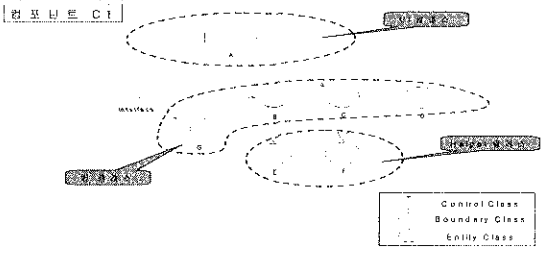


그림3. 클래스 분리

3.2 메소드 기능 분류에 의한 컴포넌트 분해

분리된 클래스를 기준으로 컴포넌트는 UI 컴포넌트, 두 종류의 빈, 빈을 도와주는 helper 클래스들의 패키지, 그리고 서비스 명세서로 분해된다. 여기서 빈 클래스는 빈과의 매핑을 위해서 메소드 기능 분류에 의해 분해 되어지는데 본 논문에서는 그림 4와 같이 4가지로 분류한다.

- UI 메소드:UI에 작용 또는 UI와 관련된 메소드
  예) getBackground() - 채팅 화면의 배경색을 바꿔준다.
- transaction 메소드:DB에 transaction이 이루어지는 메소드
  예) check() - 사용자와 패스워드를 체크한다.
- service 메소드:어플리케이션 서버에서 제공되어지는 기본적인 서비스에 해당하는 메소드
  예) ChattingServer() - 여러 클라이언트의 채팅을 위한 서버 역할을 한다.
- business 메소드:그밖에 일반적인 비즈니스 로직을 구현하는 메소드
  예) ReadMessage() - 메시지를 리턴한다.
  WriteMessage() - 메시지를 보낸다.
  Login() - 로그인한다.

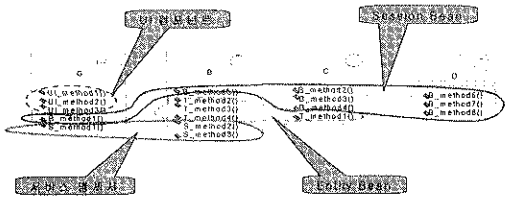


그림4. 메소드 기능 분류에 의한 컴포넌트 분해

먼저 빈 클래스 중 UI기능을 가지는 클래스는 UI 메소드와 관련 속성들을 추출하여 새로운 UI클래스로 구성이 되고, 이 클래스는 그림 3의 UI 클래스에 속하게 된다. 이렇게 구성된 UI 클래스를 묶어서 하나의 UI 컴포넌트로 분해 되어지는데 이 때 컴포넌트 식별 프로세스에서 정의된 인터페이스와 컴포넌트 명세서를 참고하여 새로운 인터페이스를 정의하고 컴포넌트 명세서를 작성한다. 다음으로 두 종류의 빈을 구성하기 위해서 트랜잭션 메소드들을 추출하여 Entity Bean을 구성하고 비즈니스 메소드들을 추출하여 Session Bean을 구성한다. 이렇게 구성된 빈은 표 1의 1차 빈 매핑 테이블에 기록한다. 빈을 도와주는 Helper 클래스는 모두 묶어서 하나의 패키지를 만든다. 서비스 명세서는 service 메소드들의 각각의 기능과 속성들의 기술을 작성하여 빈 배치자에게 제공한다.

표 1 1차 빈 매핑 테이블

컴포넌트 이름	빈 이름	메소드 시그네쳐	사용하는 메소드	사용하는 속성	메소드 기능
C1 컴포넌트	Entity Bean	T_method1()	B_method2()		DB에 ... 추가한다.
		T_method2()			DB에 ... 검색한다.
		T_method3()	B_method5()	b	DB에 ... 검색한다.
		T_method4()			DB에 ... 삭제한다.
	Session Bean	B_method1()			... 기능을 수행한다.
		B_method2()	T_method1() B_method6()		... 기능을 수행한다.
		B_method3()			... 기능을 수행한다.
		B_method4()	T_method1()	a	... 기능을 수행한다.
		B_method5()	T_method4()	b	... 기능을 수행한다.
		B_method6()		d,e	... 기능을 수행한다.
		B_method7()	B_method3() B_method8()	f	... 기능을 수행한다.
		B_method8()		g	... 기능을 수행한다.

3.3 속성 상태분류에의한 속성 배치 및 빈 분해

메소드의 기능에 의해 분해된 빈은 각 메소드들이 사용하는 속성을 배치 시켜야한다. Session Bean의 경우 그 속성의 상태에 따라 두 가지의 빈으로 분해한다. 본 논문에서는 빈과의 매핑을 위한 속성 기능을 2가지로 분류한다.

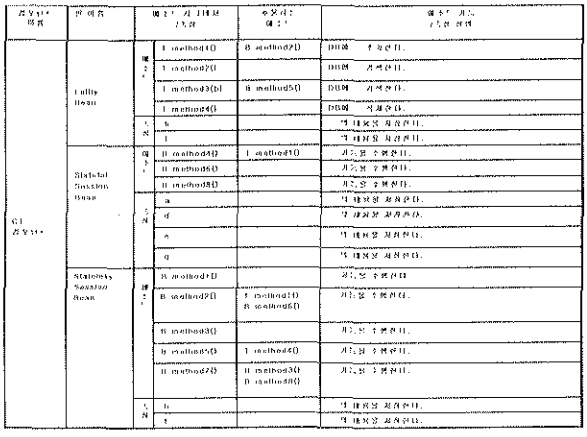
● conversational 속성 : 클라이언트와의 지속적인 대화를 나타내는 속성으로, 이것은 빈을 호출하는 클라이언트와 관련된 상태를 유지해 주는 속성이다. 거의 모든 속성들이 여기에 속한다.

● non-conversational 속성 : 빈을 사용하는 모든 클라이언트에 대해서 어떠한 상태도 유지할 필요가 없는 속성으로, 주로 final과 static으로 선언된 속성과 임시 변수와 같은 속성이 여기에 속한다.

엔티티 빈의 속성은 데이터베이스 테이블의 애트리뷰트 집합으로 구성한다. 엔티티 빈의 메소드가 사용하는 속성이 있다면 그 속성은 메소드의 파라미터에 포함을 시킨다. 세션 빈에서 conversational 속성을 사용하는 메소드들은 Stateful 세션 빈으로 그 외에 non-conversational 속성을 사용하는 메소드들과 속성을 사용하지 않는 메소드들은 Stateless Session Bean으로 구성한다. 이렇게 구성된 빈을 표 2의 2차 빈 매핑테이블에 기록한다.

conversational 속성 : g,h,k
non-conversational 속성 : a,I,j

표 2 2차 빈 매핑 테이블



3.4 빈과의 매핑

빈을 구성하기위한 2차 빈 매핑테이블을 기준으로 빈을 기술한다. 빈의 기술은 빈의 사용자와 빈, 빈과 빈간의 상호작용을 위한 홈 인터페이스와 리모트 인터페이스를 정의하고 빈 명세서에 빈의 기능을 기술한다.

4. 사례 연구

위에서 제시한 빈 추출 기법을 바탕으로 기존 RMI 기반의 채팅 시스템 중 채팅기능의 한 컴포넌트를 구성하여 EJB 빈으로 매핑하였다. 다음은 채팅기능 컴포넌트에 대한 클래스 다이어그램이다.

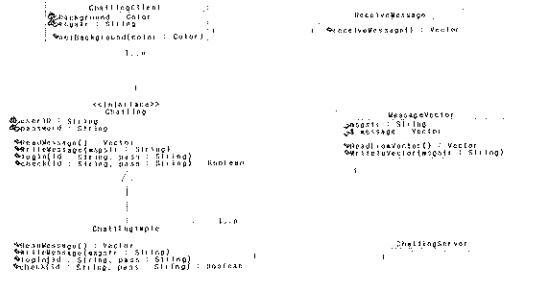


그림 3 채팅기능 컴포넌트의 클래스 다이어그램

4.1 클래스 분리

기존 채팅기능 컴포넌트의 클래스 분리는 표 3과 같다.

표 3 클래스 분류

클래스 분류	클래스 명	클래스 설명
UI 클래스	ChattingClient	
Bean 클래스	Chatting	
	ChattingImple	
	ChattingServer	
	ReceiveMessage	UI클래스와 상호 작용함
Helper 클래스	MessageVector	Bean클래스와 관계

4.2 메소드 기능 분류에 의한 컴포넌트 분해

표 3의 Bean 클래스의 메소드 기능 별 분류는 표 4와 같다. 여기서, ChattingServer 클래스는 클래스 자신이 생성자로서 service 메소드의 역할을 한 후 사라진다.

표 4 메소드 기능별 분류

컴포넌트 이름	빈 종류	메소드 시그네쳐	사용하는 메소드/속성	메소드 기능 설명
C1 컴포넌트	Entity Bean	Check()	setString()/userID, password	DB에서 사용자 유무를 리턴한다.
	Session Bean	ReadMessage()	ReadFromVector()	메세지 벡터를 리턴
		WriteMessage()	WriteToVector()/msgstr	메세지 벡터에 메시지를 추가한다.
		receiveMessage()		리턴된 벡터를 채팅창에 뿌려준다.
		Login()	Check()/userID, password	로그인 결정

4.3 속성 상태 분류에 의한 속성 배치 및 빈 분해

표 4에서 분류된 메소드들이 사용하는 속성은 다음 표 5과 같이 배치된다.

표 5 속성 상태 분류

컴포넌트 이름	빈 종류		메소드 시그네쳐	호출하는 메소드	메소드 / 기능 설명
C1 컴포넌트	Entity Bean	메소드	Check()	setString()	DB에서 사용자 유무 리턴
		속성	String userID		사용자 아이디 지정
			String password		사용자 비밀번호 지정
	Stateful Session Bean	메소드	Login()	Check()	로그인 결정
		속성	String userID		현재 사용중인 사용자 아이디 지정
			String password		현재 사용중인 사용자 비밀번호 지정
	Stateless Session Bean	메소드	ReadMessage()	ReadFromVector()	메세지벡터 리턴
			WriteMessage()	WriteToVector()	메세지벡터에 메시지 추가
			receiveMessage()		리턴된 벡터를 채팅창에 뿌려준다.
		속성	String msgstr, static Vector message		메시지의 내용 지정

4.4 빈과의 매핑

위의 표 1 부터 표 3에 걸쳐 재구성된 컴포넌트들을 빈과 매핑시킨다.

5. 결론 및 향후 연구

본 논문에서는 기존의 클래스 단위의 어플리케이션을 EJB의 컴포넌트 단위인 빈 단위의 재사용성을 가지는 어플리케이션으로 전환하는 기법 중 핵심 단계인 빈 추출 기법을 제시하였으며, 이를 채팅 시스템에 적용하여 실제 구현을 해보았다. 향후 연구 과제로는 전체적인 프로세스 각각에 대한 구체적인 연구와 신뢰성 있는 방법으로 정착하기 위하여 많은 사례연구가 이루어져야 할 것이다.

6. 참고 문헌

[1] Sun Microsystems, Inc., Enterprise JavaBeans Specification, version 2.0, 2000.

[2] Ed Roman, Mastering Enterprise JavaBeans and the Java™2 Platform, Enterprise Edition, John Wiley & Sons, Inc., 1999.

[3] Henri Jubin, Jurgen Friedrichs, and the JalapenoTeam, Enterprise JavaBeans by Example, Prentice-Hall, Inc., 2000.

[4] Sun Microsystems, Inc., J2EE Connector Architecture Specification, v1.0, June 1, 2000.

[5] 컴포넌트 생성 및 추출을 위한 기법 개발-II, 한국 전자 통신 연구원, 2000.