제 18 회 한국정보처리학회 추계학술발표대회 논문집 제 9 권 제 2 호 (2002.11)

EJB 서버 시스템에서 인스턴스 관리 방법에 관한 연구

정승욱*, 이경호*, 김중배*
*한국전자통신연구원 전자거래연구부
e-mail : swjung@etri.re.kr

A Study on the Instance Management for EJB Server System

Seung-Woog Jung*, Kyeong-Ho Lee *, Joong-Bae Kim*
*Electronic Commerce Department
Electronics and Telecommunications Research Institute

요 약

EJB(Enterprise Java Beans)는 비즈니스 로직을 컴포넌트 형태로 작성하여 재 사용성을 향상시킨 서버 측 컴포넌트 모델로서, J2EE(Java2 Enterprise Edition)의 핵심이다. EJB는 컴포넌트를 특성에 따라 일반적인 비즈니스 로직을 나타내는 세션 빈(Session Bean), 데이터베이스에 저장된 데이터를 나타내는 엔터티 빈(Entity Bean) 그리고 JMS 메시지를 처리하는 메시지 드리븐 빈(Message-driven Bean)으로 구분한다. 이러한 빈들은 EJB 서버에 의해 관리된다. 본 논문에서는 EJB 서버에서 EJB 빈의 라이프 싸이클을 관리해주는 인스턴스 관리자(Instance Manager)에 대해 논의한다.

1. 서론

J2EE는 SUN사가 자바를 기업형 환경에 적용할 수 있도록 만든 기업형 자바 미들웨어 프레임워크이며, 웹 응용 서버 표준을 일컫는다. J2EE 는 컴포넌트 기반의 기업형 비즈니스 응용을 개발하기 위해 필요한 기본 서비스를 제공함으로써, 개발자에게 비즈니스 로직 만을 개발하도록 하고 기타 복잡한 과정은 자동적으로 처리해주는 환경을 제공한다.

엔터프라이즈 어플리케이션(Enterprise Application)은 JSP/Servlet 으로 구성된 웹 컴포넌트와 EJB 빈으로 구성된 비즈니스 컴포넌트로 이루어지며, 웹 응용 서버로 배포된다. JSP/Servlet 은 웹 환경에서 사용자에게 동적 콘텐츠(Dynamic Contents)를 제공하며, EJB 빈은 비즈니스 로직을 담당하는 분산 객체 컴포넌트이다.

EJB 는 웹 응용 서버 스펙인 J2EE 의 핵심으로서, 비즈니스 로직을 컴포넌트 형태로 작성하여 재 사용성을 높이기 위한 서버 측 컴포넌트 프로그래밍 모델이다. EJB 는 컴포넌트를 특성에 따라 일반적인 비즈니스 로직을 나타내는 세션 빈(Session Bean), 데이터베이스에 저장된 정보와 같은 여러 클라이언트에 의해 공유되며 영속 장치에 저장되는 엔터티 빈(Entity Bean) 그리고 JMS 메시지를 처리하는 메시지 드리븐 빈(Message-driven Bean)으로 구분한다. 이러한 빈들은 EJB 스펙에 따라 구현된 EJB 서버 시스템에 의해 관리된다.

본 논문에서는 ETRI 에서 개발한 E504 EJB 서버에서 EJB 빈의 라이프 싸이클(lifecycle)을 관리해주는 인스턴스 관리자(Instance Manager)에 대해 논의한다.

2. EJB 컨테이너 기본 구조

EJB 빈은 주로 기업 내 비즈니스 로직을 컴포넌트 형태로 작성한 서버 측 컴포넌트이다.

EJB 를 작성하기 위해서는 리모트 인터페이스를 얻기 위한 홈 인터페이스, 리모트에서 호출할 수 있는 메소드를 정의한 리모트 인터페이스, 그리고 실제 비즈니스 로직을 구현한 빈 인스턴스 등 세 개의 클래스를 작성해야 한다. EJB 클라이언트는 JNDI 네이밍

서버에서 홈 인터페이스에 대한 레퍼런스를 얻은 후 홈 인터페이스를 통해 리모트 인터페이스에 대한 레퍼런스를 얻을 수 있으며, 이를 이용해 빈 인스턴스에 메시지를 보내게 되다. 이 메시지는 바로 빈 개발자가 작성한 빈 인스턴스에 전달되는 게 아니라 컨테이너가 중간에 가로채 보안 및 트랜잭션 처리를 수행 한 후 해당 빈에게 메시지를 전달한다.

그림 1 은 E504 컨테이너 시스템의 전체 구조를 나타낸 그림이다.

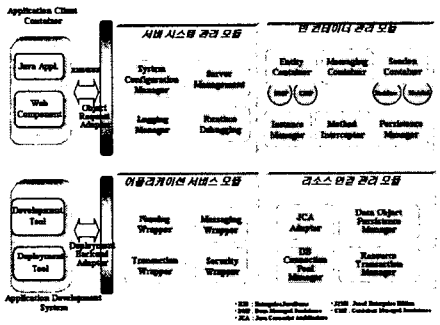

그림 1. E504 EJB 컨테이너 구조

서버 시스템 관리 모듈은 어플리케시션 개발자로부터 배포된 어플리케이션의 탑재, 컨테이너의 생성, 서버에서 구동되는 각 관리자의 상태 정보의 모니터링 및 시스템 로그 정보를 수집하는 기능을 수행한다.

빈 컨테이너 관리 모듈은 배포된 빈의 속성 정보 관리, 인스턴스 생명주기 관리, 빈 메소드에 설정된 미들웨어 서비스 적용, 빈 메소드를 실행하고, 예외 처리 기능을 수행한다.

어플리케이션 서비스 모듈은 개발자가 설정한 각종 미들웨어 서비스를 적용하기 위해서, 인증 및 접근제어 관리, 분산 트랜잭션 관리, 메시지 서버 연동 관리의 기능을 수행한다.

리소스 연결 관리 모듈은 빈과 영속성 관리자에서 사용하는 데이터베이스 연결 관리, JCA 기반 레거시 정보 시스템 연동 기능, 데이터베이스 연결의 트랜잭션 연동 관리 기능을 수행한다.

그림 2 는 클라이언트로부터 E504 EJB 컨테이너 내에 있는 빈까지의 요청 처리 과정을 나타낸 그림이다.

클라이언트의 요청이 컨테이너에 전달되면 해당 요청은 여러 인터셉터(interceptor)를 거치게 된다. 인터셉터는 트랜잭션, 보안 등의 기본 서비스를 제공하게 된다. 컨테이너가 빈의 메소드를 호출하기 전, 인스턴스 관리자에게 해당 요청을 처리할 빈을 요구한다. 컨테이너는 해당 빈을 인스턴스 관리자로부터 제공 받은 후 빈의 메소드를 호출하고 결과를 클라이언트에게 반환한다.

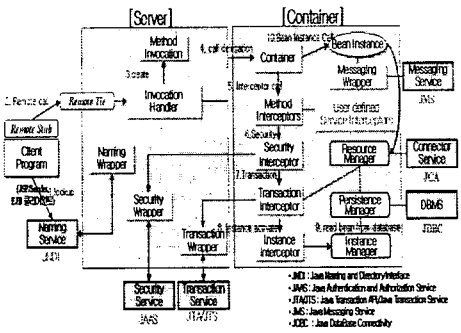

그림 2. 요청 처리 과정

상태 세션 빈은 특정 클라이언트에 종속된 빈으로서 클라이언트으로 부터 일정 기간 동안 요청이 없는 경우 활성화된 빈을 메모리에서 제거해 주도록 EJB 스펙은 규정하고 있다.

엔터티 빈의 경우에는 일정 시간 동안 클라이언트로부터 요청이 오지 않는 빈의 경우 무한정 활성 상태로 메모리에 올려 놓는 것은 메모리 부족 현상을 초래할 가능성이 높기 때문에 비활성화 시켜 주어야 한다. 또한, 프라이머리 키가 같은 엔터티 빈에 대해 다수의 클라이언트가 동시에 접근이 가능하며, 다수의 클라이언트 요청이 동시에 빈에게 전달되면 빈 내부에서 사용되는 변수가 원하지 않는 값을 가지게 되는 경우가 발생할 수 있다. 이런 경우 컨테이너가 각 클라이언트 요청을 순차적으로 수행하게 해야 한다.

클라이언트가 어떤 빈에게 메시지를 요청하였을 경우 해당 빈 인스턴스을 생성하고, 일정 시간 클라이언트로 부터 요청이 오지 않는 경우 해당 빈을 메모리에서 제거하는 등의 빈의 라이프 싸이클을 관리하는 부분이 필요하며, 이를 수행하는 모듈을 인스턴스 관리자라 한다.

빈은 빈 인스턴스 뿐만 아니라 현재 해당 빈에 적용되어 있는 트랜잭션 및 기타 다양한 부가 정보를 가지고 있어야 하며 이를 빈 컨텍스트에 저장한다. 인스턴스 관리자는 실제로는 빈 인스턴스 및 부가 정보를 가지고 있는 빈 컨텍스트를 관리한다.

3. 인스턴스 관리자의 요구사항

인스턴스 관리자의 요구 사항은 빈 타입에 따라 달라진다.

1) 무상태 세션 빈(Stateless Session Bean)

무상태 세션 빈의 경우 빈 내부에 상태를 가지고 있지 않기 때문에 빈 하나를 이용해 다수의 클라이언트 요청을 동시에 처리할 수 있다. 인스턴스 관리자는 무 상태 세션 빈을 포함하는 빈 컨텍스트를 미리 특정 개수 만큼 생성한 후, 클라이어트의 메시지 요청이 있을 경우 미리 생성된 빈을 이용해 처리한다.

2) 메시지 드리븐 빈(Message-driven Bean)

메시지 드리븐 빈도 무상태 세션빈의 한 종류이므로 무상태 세션 빈과 같다.

3) 상태 세션 빈 (Stateful Session Bean)

상태 세션 빈은 하나의 클라이언트에 대해 유일하게 하나만 생성되기 때문에, 다수의 클라이언트 요청이 동시에 수행되지 않는다. 또한 상태 세션 빈은 내부에 상태 정보를 가지고 있으며, 이를 초기 상태로 되돌리는 부분이 존재하지 않기 때문에 한번 사용된 빈은 다른 클라이언트 요청을 처리하기 위해 재사용될 수 없다.

클라이언트의 요청이 많아 지면 메모리 상에 존재하는 상태 세션 빈의 수도 많아져 메모리 용량이 모자랄 경우가 발생할 수 있다. 컨테이너는 시스템 상황을 검사하여 트랜잭션과 연관되지 않은 상태 세션 빈을 저장소에 임시 저장하여 비활성화 시키고, 클라이언트 요청이 오면 그때 다시 메모리에 활성화시켜 메모리를 효율적으로 관리해야 한다. 또한, 지정된 일정 시간 동안 클라이언트에서 요청이 오지 않으면 상태 세션 빈을 제거해야 한다.

4) 엔터티 빈 (Entity Bean)

엔터티 빈은 다수의 클라이언트에 의해 공유될 수 있다. 다수의 클라이언트 요청이 동시에 빈에게 전달되면 빈이 원하지 않는 상태가 될 수 있기 때문에 컨테이너는 엔터티 빈에 대한 동시성 제어를 수행해야 한다.

동시성 제어를 위해 주로 사용되는 방법은 아래와 같다.

첫번째 방법은 해당 프라이머리 키를 갖는 엔터티 빈을 하나만 생성하고, 여러 트랜잭션이 해당 빈을 공유하게 하는 것이다. 여러 트랜잭션이 해당 빈을 접근하려고 할 때는 순차적으로 수행될 수 있도록 컨테이너가 보장해 주어야 한다. E504 컨테이너 시스템에서는 락킹 메커니즘을 통해 해당 엔터티 빈에 대한 락을 소유한 트랜잭션 만이 해당 빈을 접근하도록 하며, 트랜잭션이 종료 되면 락을 놓아 다른 트랜잭션이 해당 빈을 접근하도록 하고 있다.

두번째 방법은 프라이머리 키가 같더라도 트랜잭션마다 하나씩 엔터티 빈을 생성하게 하는 방법이다. 이 경우는 트랜잭션마다 하나의 엔터티 빈이 생성되므로 동시성 제어를 위한 부가적은 작업을 필요로 하지 않는다. 그러므로, 첫번째 방법보다는 속도가 빠르나 메모리는 더 많이 차지하게 된다.

엔터티 빈은 클라이언트 요청이 오면 메모리에 계속 남아 있게 된다. 클라이언트 요청이 많아 지면 메모리에 올라와 있는 엔터티 빈의 수가 많아 메모리를 많이 차지하게 된다. 메모리에 올라와 있는 엔터티 빈의 수가 어느 한도 이상이 되면, 해당 빈의 상태를 데이터베이스에 기록하고 빈을 메모리에서 제거해야 한다. 메모리에서 빈을 제거하는 알고리즘은 LRU(Least Recently Used)를 기본으로 사용하며 대체 가능하다.

이상을 요약해 보면, 인스턴스 관리자는 클라이언트 요청이 오면 빈을 생성하고, 메모리에 빈 인스턴스가 많아지면 적당히 선택하여 메모리에서 제거해주는 역할을 수행한다. 또한, 동시 접근이 가능한 빈의 경우 동시성 제어 기능도 수행한다.

4. 인스턴스 관리자 구조

그림 3 은 인스턴스 관리자의 기본 구조를 나타낸 그림이다.

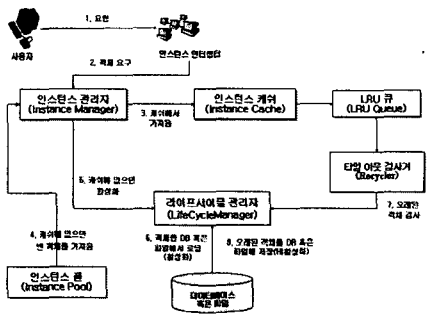

그림 3. 인스턴스 관리자 기본 구조

클라이언트의 요청이 전달되면 컨테이너는 해당 클라이언트 요청의 대상이 되는 빈 컨텍스트를 인스턴스 관리자에게 요청한다. 인스턴스 관리자는 해당 빈 컨텍스트가 캐쉬에 있는지 검사하여 있으면 해당 빈 컨텍스트를 반환하고, 없으면 풀에서 미리 생성된 빈 컨텍스트를 가져와 트랜잭션 및 기타 상태를 세팅하고 캐쉬에 삽입한 후 반환한다.

만약, 빈 컨텍스트을 캐쉬에 삽입할 때 캐쉬의 크기가 꽉 찬 경우 캐쉬 내의 빈 컨텍스트를 선택하여 비활성화 시키고 해당 빈 컨텍스트을 삽입한다. 비활성화 정책은 기본적으로 LRU(Least Recently Used)를 사용하며 다른 것으로 대체 가능하다.

인스턴스 관리자는 기본적으로 인스턴스 풀, 인스턴스 캐쉬, 타임아웃검사기, 라이프 싸이클 관리자에 대한 레퍼런스를 가지고 있으며 인스턴스 관리에 대한 기본 인터페이스를 제공한다. 인스턴스 관리자는 각 빈 타입에 따라 기본적으로 StatelessSessionInstanceManager, StatefulSessionInstanceManager, EntityInstanceManager, MultiEntityInstanceManager, MessageDrivenBeanInstanceManager 4 개가 존재하며, 각각 InstanceManager 라는 인터페이스를 구현한 추상 클래스 AbstractInstaceManager 를 상속하고 있다. EntityInstanceManager 는 빈 인스턴스가 하나만 존재하는 경우이고 MultiEntityInstanceManager 는 트랜잭션마다 빈 인스턴스가 생성되는 경우이며, 이러한 인스턴스 관리자는 배포시 배치 기술자에 기술하게 된다.

인스턴스 풀은 EJB 빈 인스턴스를 일정 개수 만큼 미리 생성하여 클라이언트가 인스턴스를 다 사용하면 다른 클라이언트가 해당 인스턴스를 재 사용할 수 있도록 보관하는 기능을 수행한다.

인스턴스 캐쉬는 현재 사용 중인 EJB 빈 인스턴스을 보관하는 기능을 수행한다.

타임아웃검사기(Recycler)는 일정 시간 마다 타임아웃된 빈을 검사하여, 타임 아웃된 빈을 메모리에서 제거하는 역할을 수행한다.

라이프 싸이클 관리자는 ejbActivate(), ejbPassivate(), ejbLoad(), ejbStore()와 같은 빈의 라이프싸이클과 관련된 요청을 처리한다.

5. 결론

지금까지 E504 EJB 서버내의 인스턴스 관리자에 대한 기능 및 간략한 구조에 대해 논의하였다.

인스턴스 관리자는 클라이언트 요청을 처리할 빈을 제공하고, 메모리에 빈 인스턴스가 많아지면 적당히 선택하여 메모리에서 제거해주는 역할을 수행한다. 동시에 접근 가능한 빈의 경우 동시성 제어 기능도 수행한다.

인스턴스 관리자는 EJB 컨테이너의 성능에 영향을 주므로, 향후 성능 향상을 위한 다양한 기법의 개발이 필요하다.

참고문헌

[1] JavaTM 2 Platform Enterprise Edition Specification, v1.4 Public Draft, SUN microsystems, July 15, 2002.

[2] Enterprise Java BeansTM Specification Version 2.0 Final Release, SUN microsystems, Auguest 22, 2001

[3] Java Naming and Directory Interface (JNDI), Version 1.2.1, Sun Microsystems, 1999

[4] Java Transaction API (JTA), Version 1.0.1, Sun Microsystems, 1999

[5] JBoss - World class J2EE technologies in open source, http://www.jboss.org

[6] OpenEJB , http://www.openejb.org