컴포넌트 뱅크 구축을 위한 EJB 배치 디스크립터의 확장

김용대° 최은만
동국대학교 컴퓨터멀티미디어공학과
{ydkim, emdhoi}@dgu.ac.kr

Extension of EJB Batch Descriptor for Components Bank Construction

Young-Dae Kim° Eun Man Choi
Dept. of Computer Multimedia Engineering, Dongguk University

요 약

사용자의 다양한 요구사항에 따른 소프트웨어 구조의 방대함과 복잡함으로 인하여 소프트웨어 부품을 재사용하는 방식의 컴포넌트 개발 방법이 최근 각광을 받고 있다. 이런 추세에 맞추어 IBM, SUN, 한국전자통신연구원 등의 여러 기관에서 공용 컴포넌트 개발, 컴포넌트 생성 및 조립기술, 컴포넌트 시험 및 품질평가 기술, 컴포넌트 유통기술 등의 컴포넌트 관련 기술 사업을 주진 중에 있다. 특히 특정 영역에서 공통으로 사용될 수 있는 공용 컴포넌트를 개발하여 공용 컴포넌트 은행에 저장하고, 컴포넌트 사용의 공용 체제를 구축하는 것은 급변하는 사용자의 요구에 따른 소프트웨어 시장 변화에 신속하게 대처할 수 있는 중요한 방안이라 할 수 있다. 이 논문에서는 EJB(Java Enterprise Beans)컴포넌트 은행 구축 시 사용자의 요구사항에 적합한 컴포넌트를 선택하고 해당 컴포넌트에 대한 올바른 이해를 돕는데 중요한 확장된 EJB 컴포넌트 명세를 제안한다.

1. 서론

현재 우리가 소프트웨어를 개발 구현하는 대부분의 방식은 라이브러리 이용 방식이다. 즉, 자신이나 다른 사람이 만들어 놓은 라이브러리를 사용하기 위해서 내부 소스의 구현을 알아야만 사용하고자 하는 부분에 적용할 수 있다.

C++나 Java등의 객체지향 언어의 확산으로 인하여 클래스 기반의 객체지향 기술이 적용 되고 있지만 객체지향의 재사용성에 대한 몇몇 회의적인 입장이 등장하면서 진정한 재사용 가능한 기술에 점차 눈을 돌리게 되었다. VB나 Dephi등의 4DL의 확산과 MS의 PC 시장의 점령으로 통한 OLE, VBX, OCX등의 기술이 점점 정착함으로 개발자 뿐 아니라 일반 사용자들까지도 컴포넌트에 대한 개념을 이해 하게 되었고, IT 마케팅에서 일어나는 추세인 아웃소싱과 패키지형 소프트웨어의 경제적인 이점이 부각되면서 더욱 컴포넌트 기반 소프트웨어 개발을 가속화 하였다.

컴포넌트 기반 개발의 핵심은 이미 개발되었거나 개발할 소프트웨어를 컴포넌트로 구성하고 이를 다시 사용하여 통합함으로 새로운 소프트웨어를 개발하자는 것이다.[1]

하드웨어 엔지니어링에서 볼 수 있었던 이런 방법을 소프트웨어 개발에 도입함으로 여러 가지 장점을 얻을 수 있다.

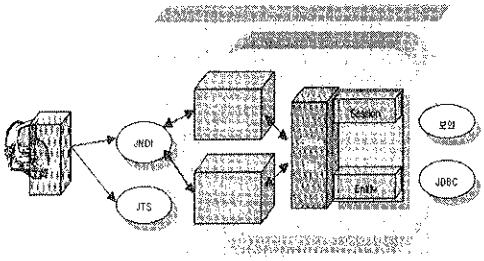
소프트웨어 컴포넌트를 재사용할 목적으로 제작된 공용 컴포넌트 은행 시스템은 재사용될 수 있는 컴포넌트들을 분류하여 컴포넌트 저장소에 저장하고, 개발자의 요구와 일치하는 재사용 가능한 컴포넌트들을 컴포넌트 저장소로부터 검색, 추출해야 한다.

따라서 본 논문에서는 EJB 컴포넌트의 특징과 용도를 표현하기위한 XML 형태의 EJB 배치 스크립트를 확장한 컴포넌트 명세를 제안하여 실제 컴포넌트 은행 시스템에서 컴포넌트 검색을 용이하게 하고, 사용자들에게는 해당 컴포넌트에 대한 정확한 이해를 돕는다. 이에 따라 2장에서는 EJB에 대하여 간략히 소개하고, 3장에서는 XML 형태의 EJB 배치 스크립트를 확장한 EJB 컴포넌트 명세를 제시하고, 4장에서 결론을 맺고자 한다.

2. 관련연구

EJB는 서버상의 자바 컴포넌트와 그 컴포넌트의 동작 환경에 대한 프레임워크로서 확장 가능한 다중 구조, 분산 애플리케이션을 만들 수 있는 컴포넌트 구조이다. 즉 컴포넌트 조합(Plug-in)을 위한 틀을 제공하며 동적으로 확장 가능한 어플리케이션 서버의 생성을 가능하게 함으로써 CORBA진영의 객체중심(메소드 호출)호출과 각 서비스(보안, 트랜잭션, 트레이더등), 스레딩 같은 개발자들이 일일이 신경 써야만 했던 부분을 컴포넌트와 어플리케이션 서버 내에서 지원하도록 한다. 따라서 프로그래머에게 전체적인 시스템의 하부구조에 대한 고심 없이 비즈니스 로직에 전념할 수 있도록 도와준다.

이러한 Frame work을 제공하는 EJB는 각각의 영역으로 나눌 수 있는데, EJB에서 작동하는 자바 컴포넌트는 엔터프라이즈 빈이라고 하며, 엔터프라이즈 빈이 기반을 두고 있는 동작 환경을 엔터프라이즈 빈 컨테이너라 한다.



[그림 1] EJB의 구조

엔터프라이즈 자바빈즈는 처리하는 데이터의 저장유무와 비즈니스 로직의 범위에 따라 세션 빈과 엔티티 빈으로 구분하고 클라이언트는 JNDI(Java Naming & Directory Interface)의 네이밍 서비스를 이용하여 엔터프라이즈 자바빈즈을 찾아내고, 빈의 원격 인터페이스를 얻어 엔터프라이즈 자바빈즈의 작업 수행결과를 얻어낸다. 또한 세션 빈과 엔티티 빈은 JDBC나, JSQL 또는 자바에서 제공하는 보안 인터페이스를 이용하여 데이터 베이스에 접근하거나 보안 기능을 수행 할 수 있고 트랜잭션 처리를 위해 JTS(Java Transaction Service)를 이용할 수 있다. 그림 2에서도 보듯이 모든 클라이언트는 실제로 엔터프라이즈 자바빈즈과 직접 통신하지는 않는다. 단지 컨테이너에서 제공하는 EJB Home Interface와 EJB Remote Interface를 통해서만 엔터프라이즈 자바빈즈에 접근할 수 있다

3. EJB 컴포넌트 명세

EJB 컴포넌트는 jar라는 압축형식을 통하여 배포되며, 이 압축형식 안의 EJB 배치 디스크립터를 통하여 해당 EJB 컴포넌트에 대한 명세를 기술한다. 하지만 기존의 SUN사에 의해 정의된 배치 디스크립트는 단순히 EJB서버가 EJB 컴포넌트를 인식하게 하기 위한 수단에 불과하다.

따라서 본 논문에서는 이 배치 디스크립트를 확장하여 재사용을 목적으로 한 다른 응용 어플리케이션 개발자가 컴포넌트 구조에 대한 정확한 이해를 하고 컴포넌트 은행 시스템 안에서 컴포넌트를 효율적으로 관리하고 검색하기 위해 컴포넌트 명세로 확장 시킬 것을 제안한다.

3.1 EJB 배치 디스크립터

EJB 컴포넌트는 보안, 트랜잭션, 네이밍, 그리고 기타 분산 객체 시스템에서 제공하는 일반적인 서비스를 EJB서버로부터 자동으로 관리된다. 배치 디스크립터는 이렇게 EJB서버가 EJB 컴포넌트를 관리하기 위해 런타임 중에 어떻게 서비스들이 각각의 EJB 컴포넌트에 적용되는지에 대한 정보를 기술한 XML문서이다. 프로퍼티 정의 파일처럼, 배치 디스크립터는 프로그램의 변경없이 런타임에 프로그램(EJB 컴포넌트)의 동작을 변경하는 것을 가능하게 한다. 프로퍼티 파일은 일반적으로 클라이언트 쪽의 응용 프로그램에서 사용하지만 배치 디스크립터는 EJB 컴포넌트에 한정적이다. 배치 디스크립터는 용도상 비주얼 베이직이나 파워빌더에서 사용하는 프로퍼티 시트와도 유사하며, 프로퍼티 시트에서 컴포넌트의 런타임 속성(배경색, 폰트 크기등)을 기술할 수 있는 것처럼, 배치 디스크립터에서는 서버측 컴포넌트의 런타임 속성(보안, 트랜잭션등)을 빈 인터페이스나 빈 클래스의 변경없이 기술할 수 있다.

[그림 2]는 Cabin 빈의 EJB 배치 디스크립터이다.[4]

```
<?xml version="1.0"?>
<!DOCTYPE ejb-jar PUBLIC "-//Sun Microsystems, Inc.//DTD Enterprise
JavaBeans 1.1//EN" "http://java.sun.com/j2ee/dtds/ejb-jar_1_1.dtd">
<ejb-jar>
 <enterprise-beans>
  <entity>
    <description>
       This Cabin enterprise bean entity represents a cabin on a cruise ship.
    </description>
    <ejb-name>CabinBean</ejb-name>
   <home>com.titan.cabin.CabinHome</home>
이하생략
```

[그림 2] Cabin 빈 배치 디스크립터

위의 예에서 사용자가 컴포넌트에 대해 얻을 수 있는 정보는 빈에 대한 간략한 기술과 컴포넌트 이름, 컴포넌트 인터페이스와 구성 클래스, 컴포넌트 접근권한 정도이다. 이러한 정보만으로는 컴포넌트 은행으로부터 추출된 컴포넌트가 사용자의 요구사항에 적합한지를 판단할 수 없으며, 또한 사용자가 해당 컴포넌트에 대해 정확히 이해하기 힘들다. 따라서 사용자가 컴포넌트의 구조나 인터페이스에 대한 이해를 높이기 위해서 컴포넌트 개발자가 컴포넌트를 적용하는데 도움이 될 수 있는 컴포넌트 명세를 제공하는 것이 중요하다.

4. EJB 배치 디스크립터의 확장

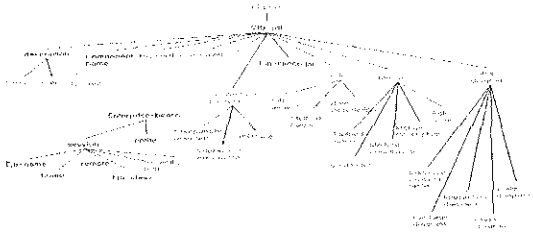
앞 절에서 설명했듯이 현재의 EJB 배치 디스크립터로는 컴포넌트에 대한 충분한 이해가 부족함으로 보다 자세한 컴포넌트 명세로의 확장을 제안한다. 확장된 컴포넌트 명세는 컴포넌트를 설명하기 위한 각 항목들로 분류하고 XML형태의 정형화된 명세 언어로 기술하여 컴포넌트 저장소로부터 원하는 컴포넌트를 쉽게 검색할 수 있을 뿐만 아니라 컴포넌트 소프트웨어의 유지보수를 용이하게 하고, 사용자에게 해당 컴포넌트에 대한 명확한 이해를 할 수 있도록 돕는다. [표 1]은 확장된 EJB 컴포넌트 명세에 필요한 항목들을 분류한 것이다.

[표 1] 확장된 EJB 컴포넌트 명세를 위한 항목

기술 항목	설 명
CLUSTER	컴포넌트 적용분야/분류코드
COMPONENT NAME	컴포넌트 이름
DESCRIPTION	컴포넌트 개요
KEYWORD	검색할 키워드
CONSTRAINTS	사용 시 요구되는 제약 사항
ARCHITECTURE DIAGRAM	컴포넌트의 구조도
ROLE	수행시 컴포넌트의 역할
METHOD	컴포넌트 메소드
DETAIL DIAGRAM	상세 설계도

[그림 3]은 기존의 EJB 배치 디스크립터로부터 확장된 EJB 컴포넌트 명세를 트리형으로 표현한 것이고, [그림 4]는 기존의 EJB배치 디스크립터 DTD와 확장된 배치

디스크립터 DTD를 비교하여 보여준다.



[그림 3] 확장된 EJB 컴포넌트 명세의 트리구조

```
<?xml version="1.0"?>
<!ELEMENT assembly-descriptor (security-role*, method-permission*,container-transaction*)>
<!ELEMENT cmp-field (description?, field-name)>
<!ELEMENT container-transaction (description?, method+, trans-attribute)>
<!ELEMENT description (#PCDATA)>
<!ELEMENT display-name (#PCDATA)>
<!ELEMENT ejb-class (#PCDATA)>
<!ELEMENT ejb-client-jar (#PCDATA)>
<!ELEMENT ejb-jar (description?, display-name?, small-icon?, large-icon?, enterprise-beans, assembly-descriptor?, ejb-client-jar?)>
<!ELEMENT ejb-link (#PCDATA)>
<!ELEMENT ejb-name (#PCDATA)>
<!ELEMENT ejb-ref (description?, ejb-ref-name, ejb-ref-type, home, remote, ejb-link?)>
<!ELEMENT ejb-ref-name (#PCDATA)>
<!ELEMENT ejb-ref-type (#PCDATA)>
<!ELEMENT enterprise-beans (session | entity)+>
<!ELEMENT entity (description?, display-name?, small-icon?,large-icon?, ejb-name, home, remote, ejb-class, persistence-type, prim-key-class, reentrant,
        cmp-field*, primkey-field?, env-entry*, ejb-ref*, security-role-ref*, resource-ref*)>
중략
```

[그림 4] 확장된 EJB 컴포넌트 명세 DTD의 확장부분

확장된 EJB 컴포넌트 명세안의 다이어그램 들은 기존의 아이콘 파일과 같이 JAR파일 내의 JPEG이나 GIF형식으로 최소 320* 240 크기의 이미지 파일로 제공한다. 그래픽 유저 인터페이스를 제공하는 배치툴이나 컴포넌트 검색 툴에서는 이 이미지를 최소크기의 프레임으로 맞추어 보여주고, 확대 축소 기능을 부여한다.

[그림 5]는 Cabin 빈에 대한 확장된 EJB 컴포넌트 명세를 사용하여 배치 디스크립트를 작성한 예이다.

```
<?xml version="1.0"?>
<!DOCTYPE ejb-jar PUBLIC "./extend-ejb-jar_1_1.dtd">
<cluster>
 <cluster-code>00010001 (서비스/예약)
   </cluster-code>
<ejb-jar>
 <keyword>선박, 예약, 여행, 서비스 </keyword>
 <constraints> self.name != null implies self.bedcount > 0
</constraints>
 <enterprise-beans>
  <entity>
    <description>
        This Cabin enterprise bean entity represents a cabin on a cruise ship.
    </description>
    <ejb-name>CabinBea</ejb-name>
  중략
```

[그림 5] Cabin 빈의 확장된 EJB 컴포넌트 명세

5. 결론

컴포넌트 시장이 활성화됨에 따라 컴포넌트의 수는 기하급수적으로 늘어나고 있으며, 효과적인 컴포넌트의 재사용을 위해서는 공용 컴포넌트 개발과 사용자들이 해당 컴포넌트에 대한 올바른 이해를 통해 필요한 곳에 컴포넌트를 사용할 수 있도록 돕는 컴포넌트 명세 및 검색 시스템등이 반드시 필요하다. 현재의 대부분의 EJB 컴포넌트들은 EJB 배치 디스크립터와 개발자의 API문서나 정형화된 간략한 예제만으로 컴포넌트에 대한 명세를 대신하고 있다. 하지만 이것만으로는 사용자가 해당 컴포넌트에 대한 정확한 이해를 하기 어렵기 때문에 본 연구에서는 기존의 EJB 명세를 토대로 확장된 컴포넌트 명세를 제안하게 되었다. 확장된 EJB 컴포넌트 명세를 사용함으로써 얻는 이점은 기존의 단순한 텍스트 형식의 프로퍼티 변경형식 이외에 컴포넌트의 내.외형적인 환경을 도식화 함으로서 사용자가 쉽게 컴포넌트를 이해할 수 있도록 돕는다.는 것이다. 또한 EJB 컴포넌트 명세를 개발자만이 제공할 수 있는 것이 아니라 개발자와 사용자를 중계하는 역할을 하는 제 3자에 의해 제작되거나 첨부될 수 있다. 제 3자는 컴포넌트 뱅크를 구축하거나 컴포넌트 뱅크안의 컴포넌트들을 검색할 수 있는 어플리케이션 개발자도 포함할 수 있다. 각각의 명세들은 XML로 구현되어있기 때문에 구조적이며, 변경과 확장성이 용이하고, 특히 웹 환경에서 적합한 컴포넌트를 검색하기가 용이하다. 따라서 CORBA의 IIOP프로토콜을 지원하는 EJB 서버를 자동으로 찾아 다니며 컴포넌트 검색 서비스를 제공해주는 에이전트를 이용한 컴포넌트 검색 시스템을 구현하는데 용이하다. 각 명세의 요소별로 검색에 필요한 가중치를 부여함으로써 효율적인 검색 알고리즘을 산출할 수 있으며, 사용자의 요구사항에 적합한 컴포넌트를 검색하는데 도움을 준다.

향후 연구로는 앞서 설명한 이동 에이전트를 통한 검색서비스 구현과 사용자요구에 부합하는 컴포넌트 검색을 위한 가중치 알고리즘 및 EJB 컴포넌트 방식을 채택한 CORBA 컴포넌트 검색 및 시스템 구축에 대한 연구가 필요하다.

5. 참고 문헌

[1] 최은만,"컴포넌트 기반 소프트웨어 개발에서 프로그램 이해 문제", 소프트웨어 공학회지 9월호, 2000.

[2] 최은만, 김선희,"소프트웨어 컴포넌트의 이해를 위한 데이터 북 구성", KSEC2001.

[3] http://java.sun.com/j2ee, "EJB 1.1 specification"

[4] R. Monson, Enterprise Java Bean, O'Reilly, 2000

[5] Peter Herzum, Oliver Sims, "Business Component Factory: A Comprehensive Overview of ComponentBased Development for the Enterprise", OMG press, 2000.

[6] Digre, T., "*Business Object Component Architecture*" IEEE Computer, Sept/Oct 1998, p66-69

[7] Desmond F.D'Souza & Alan C.Will, "Object, Components and Frameworks with UML", Addison Wesley, 1997

[8] Object Management Group, "Object Constraint Language Spedification", Version 1.1, 1997

[9] Joao Pedro Sousa and David Garlan, "Formal Modeling of the Enterprise JavaBeansTM Component Integration Framework", FM' 99: World Congress on Formal Methods, 1999.