서버측 애플리케이션 개발을 위한 EJB 지원 엔터 프라이즈 빈즈 생성기/전개기의 설계 및 구현

(Design and Implementation of Enterprise Beans Generator/ Deployer supporting EJB for Server-Side Application Development)

노 혜 민 † 이 상 영 †† 김 송 주 †† 유 철 중 †††
(Hye Min Noh) (Sang Young Lee) (Song Ju Kim) (Cheol Jung Yoo)
장 옥 배 ††† 이 우 진 †††† 신 규 상 †††††
(Ok Bae Chang) (Woo Jin Lee) (Gue Sang Shin)

요 약 J2EE(Java™ 2 Platform, Enterprise Edition) 플랫폼의 핵심기술인 EJB(Enterprise JavaBeans)는 서버 측 컴포넌트 표준 모델이다. 최근 들어 점차 이를 지원하는 도구의 중요성이 대두되고 있고 EJB만을 전문적으로 개발하는 도구들도 선보이고 있는 추세에 있다. 본 논문에서는 EJB 서버 내에서 사용되는 엔터프라이즈 빈즈(Enterprise Beans)를 컴포넌트 모델로 설계하고 설계된 내용을 임포트(import)하여 골격 코드 수준으로 엔터프라이즈 빈즈 코드를 자동 생성하는 코드 생성기(code generator)와 이를 애플리케이션 서버에 전개하는 EJB 전개기(EJB deployer)를 설계하고 구현한다. 코드 생성기는 JAR 파일을 자동으로 생성해 주는 기능을 가지고 있다. 생성된 JAR 파일은 EJB 전개기를 통해 EJB 애플리케이션 서버에 전개되어 클라이언트가 사용할 수 있게 지원한다. 또한 본 도구는 EJB 스펙에 따라 개발된 EJB 서버와 컨테이너를 지원한다. 본 도구를 활용하면 보다 빠르고 정확하게 엔터프라이즈 빈즈를 생성하고 전개할 수 있다.

Abstract EJB(Enterprise JavaBeans), the cornerstone for J2EE(Java™ 2 Platform, Enterprise Edition), is a server-side component standard model. Recently, development of supporting tool for this component model becomes important and there are many tools for developing EJB. In this paper, we purpose to design Enterprise Beans by component model, and perform design and implementation of code generator which generates skeleton code for Enterprise Beans automatically. We also implement EJB deployer for deploying these on application server. The code generator can make a JAR file automatically. This JAR file is deployed on application server by EJB deployer, and used by a client. This tool supports EJB server and container conforming to EJB specification as well. Using this tool, developer can create and deploy of Enterprise Beans more fast and exactly.

† 학생회원 : 전북대학교 전산통계학과
hmino@cs.chonbuk.ac.kr
†† 비 회 원 전북대학교 전산통계학과
leesy@cs.chonbuk.ac.kr
songju@cs.chonbuk.ac.kr
††† 종신회원 : 전북대학교 컴퓨터과학과 교수
cjyoo@moak.chonbuk.ac.kr
okjang@moak.chonbuk.ac.kr
†††† 비 회 원 : 한국전자통신연구원 컴소연 S/W공학연구부 연구원
woojin@etri.re.kr
††††† 정 회 원 : 한국전자통신연구원 컴소연 S/W공학연구부 연구원
gsshin@etri.re.kr
논문접수 : 2001년 2월 2일
심사완료 : 2001년 7월 12일

1. 서 론

EJB는 급변하는 인터넷 환경 속에서 애플리케이션을 안정적으로 구축할 수 있도록 도와주는 서버 측 컴포넌트에 대한 표준 모델이다. 이러한 EJB를 기반으로 애플리케이션을 구축하면 기존에 개발자들이 부담해야 했던 트랜잭션, 동시성 제어, 지속성, 보안 등의 처리를 EJB 컨테이너(container)가 담당해준다. 그러므로 개발자들을 비즈니스 로직에만 전념할 수 있게 해주는 장점이 있다 [1, 2]. 이와 같은 이유로 기존의 시스템이나 EJB 기반

이 아닌 시스템들도 EJB를 구현하기 위한 시스템으로 전환을 시도하고 있는 추세에 있다. 또한 EJB를 개발하기 위한 언어로 Java를 사용하는데 이는 플랫폼에 독립적이고 분산 객체 지향 언어이기 때문이다[3].

이미 유수의 개발회사들이 자신들의 도구에 EJB를 구현하기 위한 모듈을 만들거나 아예 EJB만을 전문적으로 개발할 수 있는 도구를 내놓고 있다. 전자의 경우는 Symantec사의 Visual Café, IBM사의 Visual Age for Java, Inprise사의 JBuilder 등이 있으며 후자의 경우에는 Sterling사의 COOL:Joe 등이 있다. 이러한 전문업체들이 개발한 도구들은 EJB 구현을 쉽고 빠르게 구현 할 수 있도록 도와주고 있다. 특히 Sterling사 등에서는 컴포넌트 모델링에서부터 EJB 코드를 생성하는 과정까지를 지원하는 도구를 출시하는 등의 노력을 경주하고 있다. 즉 하나의 개발도구로 객체 모델링에서부터 이에 대한 Java 소스 코드 생성, EJB 서버에 설계하여 생성한 컴포넌트들을 전개해 곧바로 테스트와 디버깅까지 가능한 솔루션을 지향하고 있는 것이다[4, 5, 6]. 지금까지 수많은 EJB 관련 Java 개발도구가 쏟아져 나오고 있고 끊임없이 신기술이 등장하고 있다. 그렇지만 지금까지 출시된 Java 개발도구를 보면 기능이나 성능 면에서 계속 보완을 필요로 한다.

또한 EJB 환경을 뒷받침하는 솔루션들이 속속 개발되고 있는데 Sun, Bluestone, BEA, IBM, Oracle, Gemston, Progress Apptivity, Novera, Secant, CompuWare, Silverstream 그리고 그 외의 다양한 벤더들이 EJB 스펙에 따라 만드는 웹 애플리케이션 서버를 지원하고 있다.

국내 실정을 보면 EJB 서버 환경에서 동작하는 컴포넌트의 Java 코드를 자동으로 생성해주는 도구는 아직 완제품으로 개발되어 있지는 않다. 여러 벤처기업들이 특정 애플리케이션 도메인에 대한 컴포넌트 개발과 코드 생성기에 대한 연구를 하고 있지만 아직 코드 생성기의 설계조차 미흡한 실정이다. 여기서 코드 생성이란 명세를 만족하는 실행 가능한 프로그램을 명세로부터 도출하는 것을 말한다[7].

본 논문에서는 컴포넌트 모델로부터 엔터프라이즈 빈즈의 Java 소스 코드를 생성해내는 코드 생성기(code generator) 부분과 이를 애플리케이션 서버에 전개하는 EJB 전개기(EJB deployer) 부분에 대해 설계하고 구현한다. 코드 생성기에서는 모델링 파일(modeling file)에서 필요한 정보를 파싱(parsing)한 후 EJB 스펙에 맞게 EJB 컴포넌트를 생성한다. 이러한 코드 생성기를 통해 안정된 코드를 제공하여 주는 특징을 가진다. 또한 EJB 전개기에서는 생성된 컴포넌트에 대한 전개 디스크립터(descriptor) 부분을 두 가지 형태로 제공한다. 먼저 EJB 스펙에 맞게 생성되는 형태와 애플리케이션 서버에 특화되어 생성하는 형태가 있다. 이렇게 구분하여 제공함에 따라 차후에 애플리케이션 서버가 변경되었을 때 확장이 용이해지는 장점을 가진다. 본 논문의 구성을 보면 우선 2장에서는 관련 연구를 알아보고 3장에서는 코드 생성기와 EJB 전개기에 대한 설계를 한다. 그리고 4장에서는 코드 생성기 및 EJB 전개기에 대해 구현한 것을 제시하고 5장에서는 기존의 도구와의 비교를 통해 성능평가를 한다. 마지막 6장에서는 결론 및 향후 연구 방향을 제시한다.

2. 관련 연구

컴포넌트는 인터페이스를 통해 서비스를 제공해주는 독립적으로 배포가 가능한 단위 소프트웨어 조각이라고 정의할 수 있는데 여기에는 패키지, 래핑(wrapping)되어진 레거시(legacy) 코드와 데이타, 기존에 개발된 컴포넌트 등을 모두 포함한다[8, 9]. 그리고 소프트웨어 컴포넌트는 잘 정의된 인터페이스 집합을 구현한 코드이다. 이와 같은 컴포넌트들은 재사용이 요구되는데 그 이유로는 재사용이 가능한 컴포넌트는 애플리케이션 개발 속도를 증진시키고 전체 애플리케이션을 작성하기보다는 미리 작성된 컴포넌트를 가지고 조립하여 벤더가 간단한 로직만을 유지하면서 빠르게 작성할 수 있기 때문이다[10]. 이러한 컴포넌트 기반의 CBD(Component Based Development)는 Microsoft사의 DCOM, OMG의 CORBA 그리고 서버 측 컴포넌트 모델인 EJB와 같은 빌딩 블럭 접근 방법을 사용한다. CBD는 비즈니스 환경 및 기술 변화를 수용할 수 없는 현 기술의 한계와 업무 통합 또는 분화에 따르는 분산 환경, 이식성, 경량화 요구, 개발자 개인의 능력과 비 자동화된 개발 절차에 따르는 문제점, 그리고 기업 활동의 생명인 시장성의 한계에 따라 필요성이 제기되었다. 이러한 CBD는 설계, 구현, 재사용을 통하여 생산성을 향상시키고 효율적으로 테스트된 코드를 사용함으로써 신뢰성을 증가시킬 수 있다. 또한 보다 소규모로 이루어진 코드 기반이므로 유지보수 비용을 절감할 수 있으며 패키지 분산 및 재사용 소프트웨어를 위해 캡슐화 메커니즘을 제공하는 장점을 갖는다[11]. 이러한 객체 지향 패러다임을 지원하는 자동화 도구의 중요성이 날로 증대되고 있다[12]. 즉 컴포넌트 구축 과정을 간소화하고 컴포넌트 이면의 핵심 로직 작성을 가능하게 하는데 IBM사의 Visual Age for Java, Inprise사의 JBuilder, Sterling사의 COOL:

Joe 등의 EJB를 지원하는 통합개발환경(Integrated Development Environment) 도구들이 있다. 컴포넌트 전개와 유지보수를 위한 도구는 컴포넌트를 구입했을 때 컴포넌트의 전개와 유지보수를 위한 기능을 제공하는데 예를 들어 환경에 맞도록 컴포넌트들을 맞춤 활동을 하는 것이다. 이러한 것은 활성화되고 있는 컴포넌트 시장에서 필수적인 사항이고 잘 정의된 컴포넌트 구조는 서로 다른 벤더들에게 표준화를 제공해주는 장점을 가진다. 웹 기반 분산환경, 애플리케이션 개발환경, 다계층 환경으로의 변화 속에 컴포넌트 기술이 분산 시스템 구축을 위한 새로운 해결책으로 인정받게 되었다[13, 14]. 이러한 분산 네트워크 상의 컴포넌트에서 필수적인 위치 투명성, 보안, 동시성 제어, 동기화 및 트랜잭션 등을 EJB에서는 제공하기 때문에 기존 시스템이나 EJB 기반이 아닌 시스템들도 EJB를 구현하기 위한 시스템으로 전환을 시도하고 있는 추세에 있다[15]. 아울러 EJB 환경을 지원하는 도구도 속속 개발되고 있다.

3. 코드 생성기 및 EJB 전개기 설계

본 논문의 코드 생성기 및 EJB 전개기는 엔터프라이즈 빈즈 컴포넌트에 대한 모델링, 설계, 구현, 시험 및 서버로의 전개 기능 등을 수행한다. 그림 1에서 보는 바와 같이 코드 생성기는 설계된 컴포넌트 모델링 파일을 입력으로 하여 정보를 파싱한 후 EJB 1.1 스펙에 근거하여 EJB 컴포넌트의 골격 코드를 생성해낸다. 그리고 Java 코드 편집기에서 필요한 비즈니스 로직을 편집하면 EJB 전개기는 적절한 전개 디스크립터(descriptor)를 포함한 JAR 형식의 EJB 컴포넌트로 만든다. 그리고 이것을 애플리케이션 서버에 전개하는 기능을 수행한다.

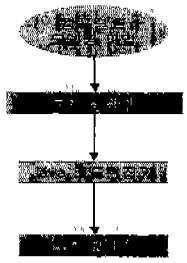

그림 1 코드 생성기 및 EJB 전개기의 구조

3.1 코드 생성기

코드 생성기는 컴포넌트 모델링 도구에서 생성된 정보를 바탕으로 각 컴포넌트 별로 수행 가능한 구현 코드를 자동 생성한다. 이러한 코드 생성기의 주요 기능을 보면 우선 구현 과정을 단순화하기 위해 컴포넌트 명세로부터 인터페이스 코드를 자동 생성하고 내부 클래스의 골격 코드를 생성하는 기능이 있다. 그리고 사용자가 비즈니스 로직을 작성할 수 있도록 Java 프로그래밍 환경을 제공하고 응용 서버에서 컴포넌트를 수행할 수 있는 EJB 코드 생성 및 수행 환경을 설정하는 기능이 있다. 코드 생성기에서 임포트하는 모델링 파일 내역과 전반적인 코드 생성기의 클래스 구성은 다음과 같다.

3.1.1 모델링 파일

모델링 파일은 컴포넌트 모델러로부터 생성된 파일로 코드 생성기에서 EJB 소스 코드를 생성하기 위한 정보를 담고 있다. 생성기에서 코드 생성을 위해 필요한 정보는 다음과 같이 <BEAN>과 <HELPER> 태그를 통해 모델링 파일에 표현된다.

먼저 <BEAN> 태그를 통해 표현되는 정보에는 빈 클래스(bean class), 홈 인터페이스(home interface), 원격 인터페이스(remote interface) 코드를 생성하기 위한 정보가 포함된다. 이러한 정보를 표현하기 위해 태그 안에 사용되는 필드(field)들에 대한 목록은 표 1과 같다.

표 1 <BEAN> 태그 필드 목록

필드 이름	필드 값의 의미
Name	빈의 이름
beanType	빈의 종류 : 엔터티(entity), 세션(session)
primaryKey	변수 중 프라이머리 키로 사용될 필드 설정
synchronize	동기화 여부(True, False)
attribute	멤버 변수
operation	멤버 메소드

또한 <HELPER> 태그를 통해 표현되는 정보를 보면 헬퍼(helper) 클래스 코드를 생성하기 위한 정보가 포함된다. 이러한 정보를 표현하기 위해 태그 안에 사용되는 필드 목록은 표 2와 같다.

표 2 <HELPER> 태그 필드 목록

필드 이름	필드 값의 의미
name	클래스의 이름
superClass	상위 클래스의 이름
attribute	멤버 변수
operation	멤버 메소드

표 3 모델링 파일 예

<BEAN>	<HELPER>	<HELPER>
Name:Cabin beanType:Entity primaryKey:int id synchronize:False attribute:int id attribute:String name attribute:int deckLevel attribute:int ship attribute:int bedCount operation:public void ejbCreate(int val1) operation:public String getName() operation:public void setName(Stringstr) operation:public int getShip() operation:public void setShip(intship)	name:Publish superclass:null attribute:String publishCode operation:public String getPublishCode()	name:Test2 superclass:null attribute:String checkBarCode attribute:String value1 attribute:String value2 attribute:String value3 attribute:String value4 attribute:String value5 attribute:String value6 operation:public String getTest1() operation:public void setTest1(Stringstr) operation:public String getTest2() operation:public void setTest2(Stringstr)

아울러 이와 같은 정보들을 담고 있는 모델링을 구체적인 예를 들어 설명하면 다음과 같다. 빈 이름이 'Cabin'이고 'Publish'와 'Test2'라는 헬퍼 클래스를 갖고 있는 엔티티 빈에 대한 모델링 파일을 보면 표 3과 같다.

모델링 파일은 위와 같은 형식으로 구성되며 이러한 모델링 파일의 내용을 기반으로 코드 생성기는 EJB 생성을 위한 골격 코드를 생성해 낸다. 즉 텍스트 형식의 모델링 파일에서 코드 생성에 필요한 정보를 추출하여 새로운 모델링 파일을 만들어낸다. 이와 같은 정보 파일을 만들어 내는 과정에 mdl 파일을 분석하는 코드만 추가하면 RationalRose 등과 같은 다른 모델링 도구와의 연계가 가능하다. 따라서 다른 도구를 이용하여 모델링 하였다 하더라도 본 논문의 코드 생성기 모듈을 이용하면 모델링 정보 파일을 분석하는 부분을 확장시켜 사용할 수 있다.

3.1.2 클래스 구성

코드 생성기는 총 9개의 클래스로 구성되어 있으며 각각의 기능은 표 4에서와 같다. 표에서 보는 바와 같이 모델링 정보를 추출하는 클래스들과 소스 코드 생성에 관련된 클래스들로 구성된다.

아울러 그림 2는 이러한 클래스간의 관계 및 상속성 등을 표현한 클래스 다이어그램이다. 본 코드 생성기는 소스 코드 생성을 위한 템플릿(template)을 가지고 있는 클래스들, 모델링 정보 추출 및 분석 클래스 및 코드를 생성해내는 메인 클래스로 구성되어 있다.

표 4 코드 생성기의 클래스

클래스 이름	기 능
SourceGen	템플릿(template)을 이용하여 코드를 생성해 내는 메인 클래스
SessionTemplet	세션 빈 클래스 코드의 템플릿을 가지고 있는 클래스
EntityTemplet	엔티티 빈 클래스 코드의 템플릿을 가지고 있는 클래스
RemoteTemplet	원격 인터페이스 코드의 템플릿을 가지고 있는 클래스
HomeTemplet	홈 인터페이스 코드의 템플릿을 가지고 있는 클래스
HelperTemplet	헬퍼 클래스 코드의 템플릿을 가지고 있는 클래스
EjbMethodTemplet	여러 EJB 메소드의 템플릿을 가지고 있는 클래스
EmiContentsLoader	모델링 파일로부터 정보를 추출하고 분석하는 클래스
JavaFilter	통합 도구에서 사용될 수 있는 프로젝트 정보를 생성하기 위해 .java 파일들을 필터링하는 클래스

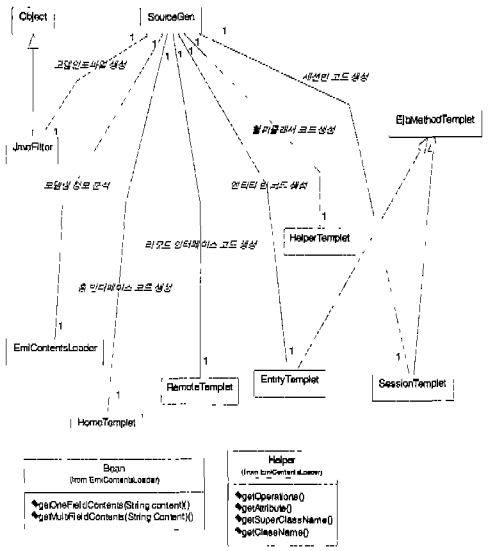

그림 2 코드 생성기의 클래스 다이어그램

3.2 EJB 전개기

EJB 전개기는 총 10개의 클래스로 구성되어 있다. 이 중 8개 클래스는 사용자 인터페이스를 구현한 클래스이다. 그리고 나머지 2개 클래스는 사용자 인터페이스를 구현한 클래스들을 이용하여 전체적인 EJB 전개기의 환경 설정 및 가동을 담당하는 메인 클래스이다. 표 5는 이러한 사용자 인터페이스를 구현한 클래스들과 각각의 기능들을 나타낸다.

표 5 EJB 전개기의 사용자 인터페이스 구현 클래스

클래스 이름	기 능
DeployToolActions	EJB 전개기 화면을 구성하는 컴포넌트들을 전개하고 각 컴포넌트에서 발생하는 이벤트를 인식하는 클래스
DeployToolWindow	EJB 전개기에서 일어나는 모든 이벤트를 처리하는 클래스
Wizard	여러 위저드(wizard)를 구성하기 위한 부모 클래스
NewEjbWizard	위저드 클래스로부터 상속을 받아 특정 위저드를 구현하는 클래스
EjbPanelGeneral	위저드에서 사용되는 패널을 구성한 클래스
EjbPanelEntitySetting	위저드에서 사용되는 패널을 구성한 클래스
EjbPanelWeblogicConfig	위저드에서 사용되는 패널을 구성한 클래스
EjbPanelNewEJB	위저드에서 사용되는 패널을 구성한 클래스

그리고 표 6는 전체적인 EJB 전개기의 환경 설정 및 가동을 담당하는 메인 클래스들의 목록을 나타내고 있다.

표 6 EJB 전개기의 메인 클래스

클래스 이름	기 능
Main	전개기를 가동하기 위한 메인 클래스
DeployTool	전개기의 전체 환경을 설정하는 클래스

아울러 그림 3은 이와 같은 EJB 전개기에서의 클래스간 관계 및 상속성 등을 표현한 클래스 다이어그램이다. 클래스 다이어그램에서 보는 바와 같이 사용자 인터페이스를 구현한 클래스들과 이들 클래스들을 이용하여 전체적인 EJB 전개기의 환경 설정 및 가동을 담당하는 메인 클래스들로 구성되어 있다.

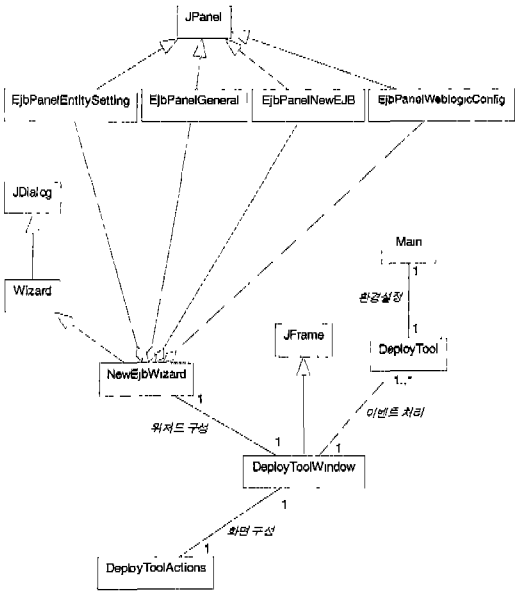

그림 3 EJB 전개기의 클래스 다이어그램

4. 코드 생성기 및 EJB 전개기 구현

4.1 코드 생성기

코드 생성기를 구현하기 위해서는 먼저 엔터프라이즈 빈즈의 클라이언트 관점 계약과 컴포넌트 관점 계약에 대해 인지해야 한다[16, 17]. 이러한 사항을 엔터프라이즈 빈즈의 아키텍처 계약이라고 한다. 본 코드 생성기에서는 이러한 계약들을 만족시키는 엔터프라이즈 빈즈의

소스 및 그 이외의 코드를 골격 수준으로 생성해내는 역할을 수행하였다. 먼저 엔터프라이즈 빈즈의 아키텍처 계약 중에 클라이언트 관점 계약(클라이언트와 컨테이너 사이의 계약)에는 클라이언트 관점에서의 홈 인터페이스, 원격 인터페이스, 메타 데이타(meta data) 인터페이스 그리고 핸들 등을 포함한다. 그리고 컴포넌트 관점 계약은 엔터프라이즈 빈즈와 컨테이너 사이의 계약으로 빈 제공자가 엔터프라이즈 빈 클래스에서 비즈니스 메소드를 구현하기 위한 요구조건 등을 포함한다.

코드 생성기에서의 주요한 클래스 소스 코드의 주요 부분을 보면 그림 4와 같다.

```
public class SourceGen {
    public SourceGen() {
    }
    public static void main(String[] args)  {
    // 환경 설정

    // 코드 생성 준비
        try {
            SessionTemplet sessiontemplet = new
SessionTemplet();
    // 엔티티 빈 코드 생성
            entitytemplet.buildEntityCode();
            fileEntityCode = new
File[entitytemplet.entityCodes.size()];
            for(int i =0; i<entitytemplet.entityCodes.size();i++) {
             bwEntityCode.close();
            beanNumCounter = beanNumCounter +1;
            }
    // 세션 빈 코드 생성
             sessiontemplet.buildSessionCode();
   // 홈 인터페이스 코드 생성
             hometemplet.buildHomeCode();
   // 원격 인터페이스 코드 생성
             remotetemplet.buildRemoteCode();
   // 헬퍼 클래스 코드 생성
             helpertemplet.buildhelperClassCode();
   // 모델링 정보 파일 생성
     String modelInfoText = modelInfoTxt.toString();
   }
}
```

그림 4 코드 생성기에서의 메인 클래스

위와 같은 코드는 모델링 파일 이름과 컴포넌트 이름을 받아들이는 인수와 생성되는 파일이 저장될 디렉토리를 설정하고 엔티티 빈 및 세션 빈 코드를 생성한다. 또한 홈 및 원격 인터페이스 코드, 기타 헬퍼 클래스 코드들을 생성시키는 구조로 구성되어 있다.

코드를 실행시키면 빈을 생성시키는 코드들이 생성된다. 즉 생성기를 통해 얻어진 소스 코드의 예를 보면 그림 5와 같다.

그리고 전체적인 도구 관리를 위해 생성해주는 프로젝트 파일의 예를 보면 그림 6과 같은데 여기에는 자신이 생성하는 파일의 이름을 나열하고 컴포넌트 이름과 소스 파일의 저장 위치에 대한 정보를 가지고 있다.

```
// 임포트 문장 삽입
import javax.ejb.*;
import java.rmi.*;
import javax.naming.*;
import java.util.*;

// public class [엔터프라이즈 빈즈 이름] implements
[SessionBean|EntityBean]
public class CabinBean implements SessionBean{

        // 엔터프라이즈 빈즈의 속성 변수 설정
        int id;
        String name;
        int deckLevel;
        int ship;
        int bedCount;
        private SessionContext sessioncontext;
        public void setSessionContext(SessionContext
        sessioncontext) {
             this.sessioncontext=sessioncontext;
        }

        // EJB-Rquired 메소드
        public void ejbActivate() {}
        public void ejbPassivate() {}
        public void ejbRemove() {}

        // EJB-Required 메소드중 ejbCreate()메소드는 입력
           파라미터에 따라 여러 개가 존재 가능
        public void ejbCreate(int val1) throws RemoteException
        {}
        public void ejbCreate(int val1, int val2) throws
        RemoteException {}

        // Buesiness 메소드
        public String getName() {
             return null;  // 리턴 타입의 반환형을 파싱하여 획득
        }
        public void setName(String str) { tihs.str = str; }
        public int getShip() {
             return 0;      // 리턴 타입의 반환형을 파싱하여 획득
        }
        public void setShip(int ship) { tihs.ship    ship;
        }
}
```

그림 5 코드 생성기를 통해 얻어진 소스 코드

```
<PATH>c:\modeler\demo\src\
                        // 생성된 소스 코드를 저장할 디렉토리
<COMPONENT_NAME>testCom
                        // 생성된 컴포넌트의 이름
<FILE>
                        // 생성된 소스 코드의 파일 이름
Cabin.java
CabinBean.java
CabinHome.java
Check.java
Publish.java
Test.java
Test2.java
TestBean.java
TestHome.java
</File>
```

그림 6 전체적인 도구 관리를 위한 프로젝트 파일의 예

이상의 코드를 실행하여 보여주는 사용자 인터페이스는 그림 7과 같다.

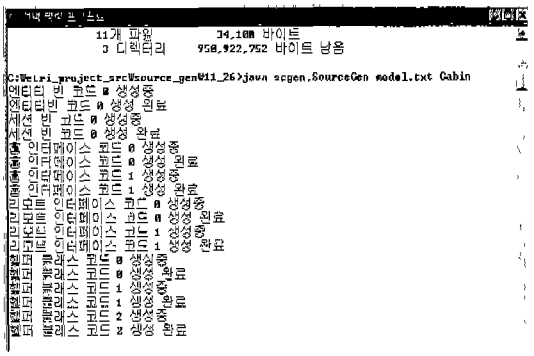

그림 7 코드 생성기 실행 화면(Dos 창의 메시지)

코드 생성기에서는 입력으로 모델링 파일을 받아 홈 및 원격 인터페이스, 엔터프라이즈 빈즈, 헬퍼 클래스 코드에 대한 골격 코드를 자동으로 생성한다. 그리고 통합 도구에서 사용될 프로젝트 파일을 생성한다.

본 논문의 코드 생성기에서 생성되는 EJB 코드는 EJB 특성이 반영되어진 모델링 도구로부터 생성된 모델링 정보 파일을 이용한다. 따라서 설계되어진 컴포넌트를 EJB 컴포넌트 스펙에 적절하게 생성해낸다. 이는 EJB 코드에 익숙하지 못한 사람들에게는 비즈니스 로직의 작성을 어렵게 할 수도 있지만 EJB 컴포넌트 스펙을 충실하게 지키고 있다는 점에서 EJB 컴포넌트 스펙을 충분히 숙지하고 있는 사람에게는 더욱 친숙하고 안정된 코드 스타일을 제공할 수 있는 장점을 가진다.

4.2 EJB 전개기

애플리케이션을 설치하고 맞춤(customization)하는 활동을 수행하는 과정을 전개라고 하는데 이 과정을 자동으로 해주는 프로그램을 EJB 전개기라 한다. J2EE 플랫폼은 이러한 전개를 위해 JAR 파일과 전개 디스크립터를 이용하고 있다. 여기서 전개 디스크립터란 특정 환경에 어떤 단위로 어떻게 묶고 전개하는가를 기술하고 있는 XML 기반의 텍스트 파일이다. 전개 디스크립터는 태그와 값을 <tag>값</tag>와 같이 표현한다. 본 논문의 EJB 전개기는 전개 디스크립터를 손쉽게 만들고 효율적으로 직접 서버에 전개할 수 있도록 하는 것을 목표로 설계되었으며 아울러 이에 대한 프로토타입 모듈을 구현하였다.

EJB 전개기에서의 주요 클래스의 소스를 보면 그림 8, 9와 같다.

```
package deployment.main;

public class Main
{
        public static void main(String args[])
        {
            // DeployTool 클래스를 인스턴스화
            new DeployTool();
        }
}
```

그림 8 EJB 전개기의 메인 클래스

```
package deployment.main;

public class DeployTool {
        public DeployTool()
        {
        // 전체적인 환경 설정
        File toolHomeDir = this.getToolHomeDirectory();

        // DeployTool의 전체 인터페이스를 인스턴스화
        DeployToolWindow deploytoolwindow = new
        DeployToolWindow();
        // 인터페이스에 프로그램 종료 명령 설정
        }

        // 환경 설정을 위한 메소드 정의
        public File getWorkingDirectory() { }
        private File getToolHomeDriectory() { }
}
```

그림 9 EJB 전개기의 소스(DeployTool.java)

EJB 전개기는 이러한 코드 외에 메인 인터페이스의 메뉴와 관련된 내용을 생성하는 메소드들, 위저드에 관련된 전개기에서 일어나는 모든 오퍼레이션을 설정하는 소스 및 위저드 상위 클래스들로 구성된다.

그림 10은 EJB 전개기를 실행시킨 전개기의 위저드를 보여준다.

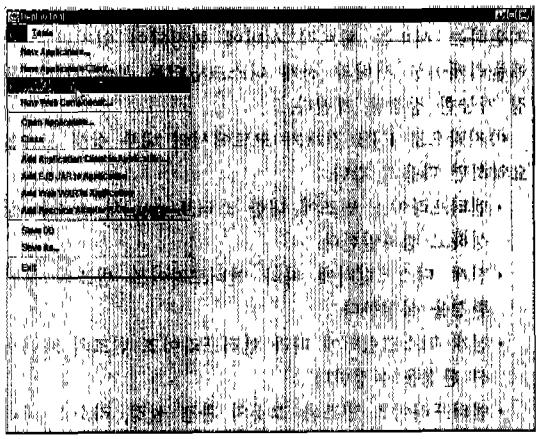

그림 10 EJB 전개기에서의 위저드 실행 화면

이와 같은 위저드는 상위 위저드 클래스를 설계하여 각 위저드가 동일한 모습을 보일 수 있도록 구현하였다. 그리고 각 위저드는 상위 위저드를 상속받아 원하는 패널(panel)로 변경할 수 있다. 이러한 위저드를 통하여 그림 11과 같은 전개 디스크립터를 생성시킨다. 생성된 전개 디스크립터 파일을 이용하면 웹 애플리케이션 서버에 컴포넌트, 모듈 및 애플리케이션 등을 설치하고 맞춤 활동을 할 수 있다.

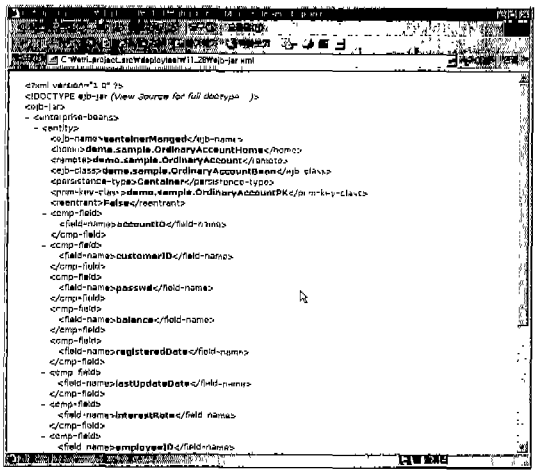

그림 11 생성된 전개 디스크립터(ejb-jar.xml)

본 논문에서의 EJB 전개기는 EJB 스펙 1.1과 웹로직(WebLogic) 서버 5.1을 대상으로 하고 있다. 그리고 EJB 스펙에 맞는 전개 디스크립터와 애플리케이션 서버에 특화된 전개 디스크립터를 생성하는 부분이 따로 동작하도록 설계되고 구현되었다. 그렇기 때문에 현재 지원하는 서버는 웹로직 서버에 한정되어 있지만 다른 애플리케이션 서버의 전개 디스크립터를 분석한다면 확장 가능한 장점을 가진다.

마지막으로 EJB 전개기 코드에서의 내부 실행 모습을 요약하면 다음과 같다.

- 엔터프라이즈 빈즈에 대한 스터브(stub)와 골격을 생성하고 컴파일한다.
- 전개 디스크립터에 따라 엔터프라이즈 빈즈의 보안 환경을 설정한다.
- 전개 디스크립터에 따라 엔터프라이즈 빈즈의 트랜잭션 환경을 설정한다.
- 엔터프라이즈 빈즈와 그것의 환경 설정, 리소스 레퍼런스(resource reference) 등을 JNDI 이름 공간(name space)내에 등록한다.
- 컨테이너 관리 영속성을 지니는 엔터프라이즈 빈즈에 대한 데이타베이스 테이블을 생성한다.

5. 비교 및 성능평가

본 논문에서의 코드 생성기 및 EJB 전개기와 가장 대표적인 관련 도구인 COOL:Joe를 비교하였다. 표 7에서와 같이 가장 대표적인 항목을 선정하여 비교 및 평가를 하였다.

표 7 COOL:Joe와의 비교

구분	항목	COOL:Joe 1.1	본 코드 생성기 및 EJB 전개기
코드 생성기	EJB 스펙 지원	1.1	1.1
	생성 소스 형태	특정 EJB 컴포넌트 형태(COOL:Joe에서만 사용)	표준 EJB 컴포넌트 형태(EJB 스펙을 따르는 표준 형태)
	데이타 표현 방법	영속성 클래스, 엔티티 빈	엔티티 빈
EJB 전개기	전개 형태	EJB 빈, 웹 컴포넌트	EJB 빈, 웹 컴포넌트, 애플리케이션

표에서 보는 바와 같이 지원가능한 EJB 스펙은 공통적으로 1.1 스펙을 지원하고 생성되는 소스 형태로는 COOL:Joe의 경우에 래핑된 특정 EJB 컴포넌트로 생성 소스를 제공하는데 반해 본 도구에서는 표준 EJB 컴포넌트 형태로 제공한다. 또한 데이타 표현 방법에 있어 COOL:Joe의 경우에 영속성 클래스 및 엔티티 빈 형태로 표현하는데 반해 본 도구에서는 엔티티 빈 형태로만 제공하여 모델링 도구로부터 생성된 모델링 정보 파일을 이용하여 EJB 스펙에 적절한 컴포넌트를 생성할 수 있게 지원한다. 이와 같이 본 도구에서 사용되는 코드 생성기와 EJB 전개기 모듈은 최신 EJB 스펙 뿐만 아니라 향후의 스펙 변화에도 대처할 수 있도록 하는데 중점을 두었다. 즉 컴포넌트 모델을 바탕으로 EJB 소스 생성시 EJB 스펙에 정의되어 있는 많은 목록을 내부에 저장하여 사용하였다. 따라서 변화되는 스펙의 변화에도 간단한 수정만으로도 대처 가능하도록 확장성을 높였다. 그리고 코드를 생성하는 단계를 자세히 알아보면 다음과 같다. 먼저 COOL:Joe의 경우 코드 생성시 컴포넌트 스펙을 변환시키는 것에서부터 시작한다. 이것은 컴포넌트 스펙 객체와 동일한 컴포넌트 구현 객체를 생성하는 단계인데 이때 COOL:Joe는 위저드를 사용한다. 이러한 위저드를 통해 비즈니스 로직을 작성하기 위해 사용되는 초기 컴포넌트 구현 클래스를 생성한다. 이와 같이

생성된 코드는 EJB에 적절한 코드는 아니며 COOL:Joe에서 사용되는 특정 컴포넌트 형식을 가진다. 구현 클래스가 생성되었다면 다음 단계부터는 비즈니스 로직을 정의 할 수 있다. COOL:Joe에서 비즈니스 로직 중에는 영속성에 관련된 클래스도 포함된다. COOL:Joe에서는 이러한 영속성 클래스를 생성하기 위해 COOL:Joe에서 제공하는 특정 팩토리(factory) 클래스를 사용하게 된다. 최종적으로 코드를 완성하게 되면 COOL:Joe에서의 컴포넌트 형식으로 하나의 컴포넌트가 완성된다. 이 컴포넌트는 컴파일된 Java 코드인 JAR 파일로 되어 있다. COOL:Joe에서는 이러한 컴포넌트를 기반으로 하여 EJB 형식의 코드를 만들어 낸다. 즉 COOL:Joe는 생성된 컴포넌트의 코드 변경 없이 EJB로 구현하기 위하여 필요한 모든 코드를 생성해낸다. 여기에 더불어 COOL:Joe는 EJB 프락시(proxy)도 생성한다. 이러한 프락시를 통해 로컬(local) 컴포넌트를 직접 호출하는 것과 같이 EJB를 호출할 수 있도록 해준다. 즉 COOL:Joe에서의 코드 생성 방법은 일반적인 Java 코드로 된 컴포넌트를 만들어 내고 여기에 COOL:Joe에서 제공하는 API 등을 이용하여 그 컴포넌트에 EJB 코드를 래핑하는 방식이다. 이에 반해 본 코드 생성기에서 생성되는 EJB 코드는 EJB 특성이 반영되어진 모델링 도구로부터 생성된 모델링 정보 파일을 이용한다. 이와 같은 COOL:Joe와 본 논문의 코드 생성기와의 차이점을 비교를 흐름도를 이용하여 요약하면 그림 12와 같다.

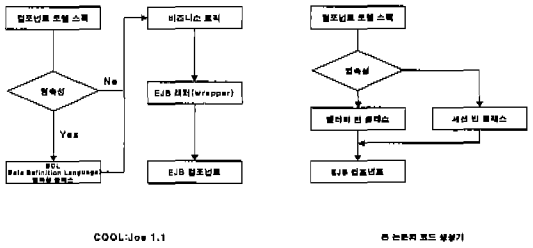

그림 12 COOL:Joe와 본 논문의 코드 생성기와의 차이를 보여주는 흐름도

위에서 제시한 것과 같이 COOL:Joe와 본 논문의 코드 생성기에서 생성되는 소스 코드의 구조를 보면 근본적인 차이가 있는데 더욱 자세한 구조 측면의 차이는 그림 13과 같다.

그림에서 보는 바와 같이 COOL:Joe의 경우에는 비즈니스 로직에 EJB 관련 클래스를 래핑하고 있지만 본 논문의 코드 생성기에서는 비즈니스 로직을 래핑하지

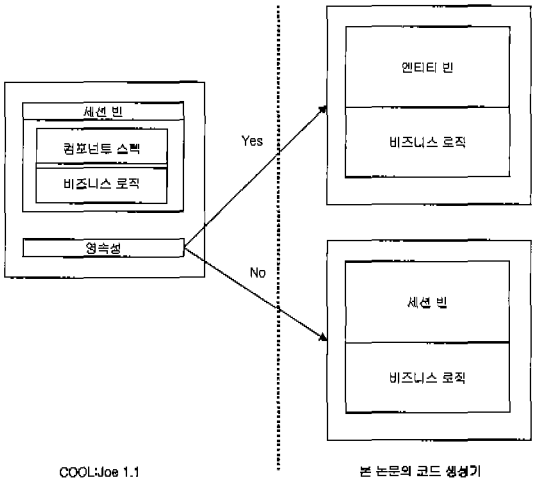

그림 13 COOL:Joe와 본 논문의 코드 생성기와의 구조상의 차이

않는다. 따라서 설계되어진 컴포넌트를 EJB 컴포넌트 스펙에 적절하게 바로 생성해낸다. 이는 EJB 코드에 익숙하지 못한 사람들에게는 비즈니스 로직의 작성을 어렵게 할 수도 있지만 EJB 컴포넌트 스펙을 충실하게 지키고 있다는 점에서 EJB 컴포넌트 스펙을 충분히 숙지하고 있는 사람에게는 더욱 친숙하고 안정된 코드 스타일을 제공할 수 있는 장점을 가진다.

6. 결론 및 향후 연구 과제

본 논문에서는 EJB 서버 내에서 사용되는 엔터프라이즈 빈즈를 컴포넌트 모델로 설계하여 설계된 내용을 골격 코드 수준으로 Java 소스 코드를 자동 생성하고 이를 애플리케이션 서버에 전개하는 코드 생성기와 EJB 전개기를 구현하였다. 본 논문의 코드 생성기는 적절하고 빠르게 EJB 스펙에 합당한 코드를 자동 생성해 주고 특히 EJB 컴포넌트 스펙을 숙지하고 있는 개발자에게는 더욱 친숙하고 안정된 코드 스타일을 제공할 수 있는 장점을 가진다. 또한 EJB 전개기는 전개 디스크립터와 애플리케이션 서버에 특화된 전개 디스크립터를 생성하는 부분이 따로 동작하도록 설계되고 구현되었기 때문에 확장 가능한 장점을 가진다.

본 도구를 활용하면 보다 빠르고 정확하게 엔터프라이즈 빈즈를 생성 및 전개할 수 있어 대학과 연구소 등의 EJB 기반 컴포넌트 제작에 이용 가능하다. 아울러 코드 생성기 및 EJB 전개기의 설계 및 구현에 있어 다양하고 세부적인 제반 기술들의 뒷받침만 된다면 EJB

기반의 엔터프라이즈 빈즈 생성기의 국산화를 이룩하는 데 있어 초석이 될수 있을 것이다.

아울러 향후 연구 방향으로는 먼저 코드 생성기 부분에 있어 모델링 파일의 내역을 최적화할 필요가 있고 EJB 전개기 부분에 있어 범 애플리케이션 서버에 전개할 수 있도록 보다 심층적인 연구가 필요하다. 아울러 본 연구를 더욱 발전시켜 개발 시간 단축, 노력 경감, 오류 방지 등의 효과를 얻을 수 있는 매크로 기능이 가미된 코드 생성기 및 EJB 전개기를 고려할 수 있겠다. 즉 개발자의 개입이 많은 빈 관리 지속성 엔티티 빈 개발 시 자동화할 수 있는 매크로 기능의 유형을 추출하여 이를 설계하고 구현할 수 있겠다. 아울러 생성한 엔터프라이즈 빈즈를 애플리케이션 서버에까지 전개하는 과정을 보다 자동화시키는 코드 생성기 및 EJB 전개기의 개발이 필요하다.

참 고 문 헌

[1] Ed Roman, "Mastering Enterprise JavaBeans & the Java 2 Platform, Enterprise Ed." Wiley & Sons, 1999.

[2] Jung Pil Choi, "Aspect-Oriented Programming with Enterprise JavaBeans," Enterprise Distributed Object Computing Conference, EDOC 2000 Proceedings Fourth International, 2000.

[3] S. Sengul, J. W. Gish , and J. F. Tremlett, "Building a Service Provisioning System using the Enterprise Java Bean Framework," Network Operations and Management Symposium, 2000 IEEE/IFIP, 2000.

[4] Sun MicroSystems Document, URL: http://industry.java.sun.com/solutions /products

[5] IBM, "Visual Age for Java 3.0 Tutorial," 1999.

[6] Sterling Software, "COOL:Joe Tutorial," V1.0, 1999.

[7] 김철홍, 천윤식, 김강호, "LOTUS 명세로부터 C++ 소스코드의 자동 생성," 한국정보처리학회논문지, 제4권, 제12호, pp. 3138-3150, 1998.

[8] Sun MicroSystems Inc., "Java™ Remote Method Invocation Specification," 1998.

[9] www.selectst.com/component

[10] Kyo C. Kang, "Issues in Component-Based Software Engineering," The Proceedings of International Workshop on Component-Based Software Engineering, LA, USA, 1999.

[11] http://www.rational.com/products/rup

[12] 선수균, 송영재, "통합 객체 관리 모델을 위한 F77/J++ 생성기에 관한 연구," 한국정보처리학회논문지, 제7권, 제10호, pp. 3064-3074, 2000.

[13] C. McClure, "Software Reuse Techniques: A Guide to Adding Reuse to the Software Process," Extended Intelligence Inc., 1996.

[14] Sherif Yacoub, Hany Ammar, and Voglor Alimili, "Characterizing a Software Component," Proceedings International Conference on Software Engineering, 1999.

[15] Richard Monson-Haefel, "Enterprise JavaBeans," O Reilly, 1999.

[16] Sun MicroSystems Inc., "Enterprise JavaBeansTM Specification," V1.1, 1999.

[17] Sun MicroSystems Inc., "Enterprise JavaBeansTM Specification." V1.2, 2000.

노 혜 민

2000년 전북대학교 컴퓨터과학과 졸업(이학사). 2000년 ~ 현재 전북대학교 대학원 전산통계학과 석사 과정. 관심 분야는 컴포넌트 기반 소프트웨어 개발, 정형 명세 기법, 소프트웨어공학 등

이 상 영

1994년 숭실대학교 산업공학과 졸업(공학사). 1994년 ~ 1995년 대한전선㈜ 근무. 1998년 전북대학교 대학원 산업공학과 졸업(공학석사). 1999년 ~ 현재 전북대학교 대학원 전산통계학과 박사과정. 관심분야는 에이전트, 멀티미디어, 소프트웨어공학, 인터넷 비즈니스 등

김 송 주

1998년 전북대학교 화학공업공학과 졸업(공학사). 2000년 ~ 현재 전북대학교 대학원 전산통계학과 석사과정. 관심분야는 소프트웨어공학, 에이전트, J2EE, 통합메시징 시스템 등

유 철 중

정보과학회논문지 : 컴퓨팅의 실제 제 7 권 제 1 호 참조

장 옥 배

정보과학회논문지 : 컴퓨팅의 실제 제 7 권 제 1 호 참조

이 우 진

1992년 경북대학교 컴퓨터과학과 졸업(학사). 1994년 한국과학기술원 전산학과 졸업(이학석사). 1999년 한국과학기술원 전산학과 졸업(이학박사). 1999년 7월 ~ 현재 한국전자통신연구원 컴퓨터.소프트웨어기술연구소 소프트웨어공학연구부 컴포넌트공학연구팀 선임연구원. 관심분야는 S/W 컴포넌트 기술, 요구 공학(requirements engineering), 정형적 명세 기법, Petri nets 등

신 규 상

1981년 성균관대학교 통계학과 졸업(학사). 1983년 서울대학교 대학원 계산통계학과 졸업(이학석사). 1983년 시스템공학연구소 연구원. 1987년 시스템공학연구소 선임연구원. 1997년 한국전자통신연구원(ETRI) 책임연구원. 2001년 충남대학교 대학원 컴퓨터과학과 졸업(이학박사). 현재 ETRI 컴퓨터.소프트웨어기술연구소 S/W공학연구부 컴포넌트공학연구팀 팀장. 관심분야는 S/W 컴포넌트 기술, S/W 재공학, CASE, 객체지향 기법 등