PSM으로부터 EJB 코드 생성을 자동화하는 방안에 관한 연구*

최연준[O] 권오천 신규상
한국전자통신연구원 컴퓨터소프트웨어연구소
{june[O], ockwon, gsshin}@etri.re.kr

A Study on Automatically Generating EJB Code from PSM

Yeon-Jun Choi[O], Oh-Cheon Kwon, Gyu-Sang Shin
ETRI-Computer & Software Technology Laboratory

요 약

급격히 발전해 온 엔터프라이즈 환경에 대하여 체계적으로 소프트웨어 개발 패러다임이 변하고 있다. 이종 플랫폼, 나아가 이종 개발 플랫폼을 자유로이 연동시킬 수 있는 개발 방법에 대한 필요성이 대두되면서 MDA 개념이 등장하였다. MDA는 프로그래밍 언어 혹은 런타임 플랫폼에 독립적인 모델에 의해 개발하는 방식으로, 개발된 모델을 특정 플랫폼에 알맞은 형태로 변환함으로써 개발 모델 및 코드의 재사용성을 극대화한다. 본 논문에서는 MDA 개발에 있어서 특정 플랫폼에 알맞은 코드를 생성하기 위한 기능과 그 설계 방안을 논한다.

1. 서 론

소프트웨어 개발에 있어서 하드웨어의 플러그인 스타일을 블랙박스와 그에 대한 인터페이스로 제공하는 컴포넌트 개발 방법론이 대두됨에 따라 개발 용이성과 재사용성이 중요시되어 왔다.

웹이 엔터프라이즈 영역의 비즈니스에 적용되면서 이러한 컴포넌트 개발 방법론과 각종 런타임 플랫폼이 개발되었다. 그러나 다양한 프레임워크의 난립으로 같은 영역의 유사한 비즈니스라고 해도 이중 개발이 불가피하게 되면서 구현 모델을 플랫폼 독립적으로 만들고자 하는 노력이 시작되었고 이것이 국제 기구인 OMG(Object Management Group)에서 만든 MDA(Model Driven Architecture)이다.

MDA는 크게 비즈니스 모델, 플랫폼 독립 모델(Platform Independent Model, 이하 PIM), 플랫폼 종속 모델(Platform Specific Model, 이하 PSM), 모델로부터 생성되는 플랫폼 종속적 구현 언어 코드로 이루어진다.

본 논문에서는 PSM 단계에서 만들어진 모델로부터 코드를 자동으로 생성하기 위한 방안을 알아본다.

2. 연구 배경

엔터프라이즈 영역에 웹 어플리케이션이 적용되기 시작함에 따라, 특정 영역에 대한 솔루션과 방법론도 개발되었다.

이에 따라 각 벤더 역시 자신의 플랫폼에 알맞은 컴포넌트 개념을 정립하고 제품화하였다. 마이크로소프트사의 윈도우 플랫폼 하에서 동작하는 COM+나 유닉스 환경에서 가장 대표적으로 사용되는 자바 진영의 J2EE 플랫폼에 포함된 EJB 등이 주요 비즈니스 컴포넌트이다.

그러나, 특정 영역에 대한 솔루션을 가지고 있음에도 불구하고 플랫폼이 다른 경우 설계한 문서만을 재사용하고 내부 로직을 재사용할 수 없어, 플랫폼에 그다지 크게 영향을 받지 않는 비즈니스 로직도 모두 새로 개발해야 하는 등, 지원해야 할 플랫폼의 종류가 많아짐에 따라 어플리케이션 구현에 드는 시간 및 비용이 크게 늘어난다. 이는, 비슷한 개념의 컴포넌트 개발 방법론이라 할 지라도 구현 자체는 플랫폼에 의존적인 특성[1]을 가지고 있어, 이렇게 개발된 컴포넌트는 동일한 플랫폼 아래에서만 재사용할 수 있다는 단점을 가진다.

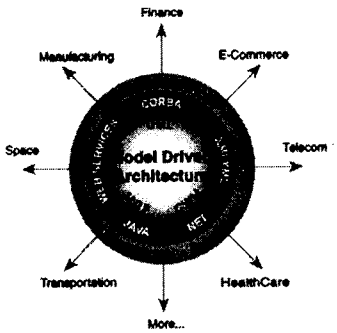
시장 적시성(time-to-market)과 안정성, 논리적인 비즈니스 어플리케이션 개발이 절대적인 영향을 미치는 엔터프라이즈 영역에 있어서 플랫폼에 독립적인 모델에 대한 엔터프라이즈 비즈니스에 대한 표준적인 개발 방법을 정의하면서, 개발 방식보다는 개발하고자 하는 목적 시스템에 초점을 맞추어 MDA를 제창하였다[1].

MDA는 비즈니스의 핵심에 해당하는 부분을 플랫폼 독립

* 본 연구는 2002년 과학기술부 국가지정연구실 사업으로 수행되었음

[1] COM+ 컴포넌트는 윈도우 플랫폼에서만 사용 가능하며, J2EE 프레임워크를 따르는 컴포넌트의 경우 운영체제에는 무관하나 각 어플리케이션 서버에 따라 컴포넌트 간의 호출에 대한 프로토콜이 표준화되어 있지 않다.

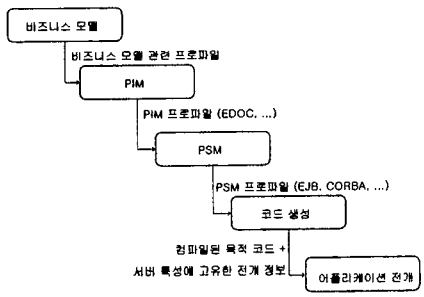
적인 모델로 작성하여, 원하는 플랫폼에 어플리케이션을 전개(deploy)시키는 경우에 플랫폼 의존적인 부분을 독립적인 모델로부터 자동화하여 생산성을 극대화시키고자 하는 접근 방법이다.



[그림 1] MDA의 기본 개념

MDA의 기본 개념은 [그림 1]과 같다. MDA 기반에 해당하는 UML, MOF, CWM이 플랫폼에 독립적인 모델의 하부 구조 및 비즈니스 표현 방식을 담당하고, 플랫폼이 아닌 표준 스펙에 의해 개발된 모델 독립적 어플리케이션이 그보다 상위 레벨의 플랫폼인 Java/J2EE, .NET, CORBA, 웹서비스에 매핑된다. 마지막으로, 각 플랫폼에서 하부 프레임워크로 제공하는 트랜잭션, 디렉토리/네이밍 서비스, 보안 기능 등이 합쳐져 통신, 제조, 금융 등의 각 엔터프라이즈 영역에 적용된다.

이러한 MDA를 지원하는 도구는 [그림 2]와 같은 주요한 동작 흐름을 가진다.



[그림 2] MDA의 동작 흐름도

PIM에 대한 UML 프로파일(profile)로 OMG에서는 UML profile for EDOC(Enterprise Distributed Object Computing)을 표준으로 권유하고 있으며[4], PSM 단계의 각 컴포넌트 플랫폼에 대한 프로파일의 표준화 작업은 각 벤더 및 관련 단체에서 추진하고 있다.

각 단계에의 전이는 모두 모델 변환에 의해 이루어지는데, 이 중 코드 생성에 관련하여, MOF 기반의 모델을 프로그래밍 언어로 변환하는 여러 연구가 진행되고 있다.

iQgen은 PIM을 표현하는 XMI로부터 JSP 템플릿과 제어 언어를 이용하여 Java 프로그래밍 코드를 생성하는 기능을 제공한다[5]. 그러나, 기본 메소드 생성 및 모델링되어 있는 메소드의 껍데기만 생성하는 등, 골격 코드에 대한 코드 생성만 지원하고 있으며 모델의 행위로부터 논리적인 흐름을 가지는 어플리케이션 코드를 생성하지는 않는다.

본 연구에서는 PSM 및 행위 흐름 등의 관련 모델 정보로부터 코드를 생성하는 과정에 대하여 알아 보고, 그 중 EJB를 위한 UML 프로파일로부터 프로그래밍 언어 코드를 생성하는 방안에 대해 논한다.

3. 본 론

PIM→PSM 변환이 일어나면서 플랫폼 독립적인 모델이 특정 플랫폼에 의존적인 모델로 구체적인 형태 변환을 한다. PSM 단계에서 만들어지고 정제된 모델은 다음의 정보를 가지고 있어야 한다.

- 플랫폼에 알맞게 변환되어 정제된 모델
- PIM 및 PSM에서 만들어지고 정제된 논리적 행위 흐름 정보

플랫폼에 알맞은 모델은 각 컴포넌트 플랫폼을 위해 정의된 UML 프로파일[2]을 그 내용으로 가지며, 논리적 행위 흐름 정보는 PIM 단계에서 어플리케이션 설계자가 ASL(Action Semantic Language), OCL(Object Constraint Language) 등의 비즈니스 로직 정보나 조건을 기술할 수 있는 UML의 일부 혹은 다른 방식으로 표현할 수 있다.

EJB에 대한 프로파일은 기본적으로 표준 EJB의 전개 디스크립터(Deployement Descriptor, 이하 DD)를 따라 모델링되어 있다[3]. 다음은 코드 생성과 관련하여 외부에 드러나는 부분과 내부에서 사용되는 부분으로 나누어 본 EJB 프로파일의 일부이다.

1) 외부 뷰
- 외부에 드러나는 오퍼레이션
 :EJBRemoteMethod,EJBCreateMethod,EJBFinderMethod
- 외부에서 사용되는 인터페이스
 :EJBRemoteInterface,EJBHomeInterface,EJBSessionHomeInterface,EJBEntityHomeInterface
- 사용하는 클래스
 : EJBPrimaryKey
- 태그 값(tagged value)
 : EJBSessionType

2) 내부 뷰
- 속성 필드
 : EJBCmpField,EJBPrimaryKeyField
- 구현 클래스
 : EJBImplementation
- 서브시스템 관계
 :EJBEnterpriseBean,EJBSessionBean,EJBEntityBean

[2] 현재 EJB 1.1 프로파일, CORBA 2.0 프로파일에 대한 초안이 나와 있다.

- 다른 부분과 연관되는 관계
:EJBReference, EJBAccess
- 태그 값(tagged value)
:EJBTransAttribute, EJBEnvEntries 등

이 중 코드 생성에 직접적으로 관계하는 프로파일 요소는 외부 뷰의 오퍼레이션, 인터페이스, 사용 클래스, 그리고 내부 뷰의 속성 필드와 구현 클래스이다.

행위 정보는 다음 [그림 3]과 같이 표현될 수 있다.

```
// Send a 'time for bed' event to all children 5 and
under.
select many children from instances of C;
for each child in children
   if(child.age<5)
      while(child.awake)
         generate C1:'time for bed' () to child;
         if(not lights.out)
            generate C2:'turn off lights' () to child;
         end if;
      end while;
   end if;
end for;
```

[그림 3] 행위 정보 기술의 예

이들에 대한 코드 변환 방법은 [그림 4]과 같이 일관성 있는 규칙을 가지게 되고, 이 규칙을 통하여 모델에 대한 코드를 생성한다.

1. EJBHomeInterface를 검사하여 Session인 경우는 세션빈으로, Entity인 경우는 엔티티빈으로 EJB 종류를 결정하고 홈 인터페이스 정보를 얻는다.
2. EJBRemoteInterface로부터 리모트 인터페이스를, EJBImplementation으로부터 빈(Bean) 클래스 정보를 얻는다.
3. 엔티티빈인 경우 엔티티 정보를 구성한다.
 가. EJBPrimaryKey로부터 프라이머리 키 클래스 정보를 얻는다.
 나. EJBCmpField로부터 엔티티빈 내에서 관리될 속성 필드 목록을 얻는다.
 다. EJBPrimaryKeyField로부터 프라이머리 키 클래스 내에서 프라이머리 키 필드로 사용될 속성 필드 목록을 얻는다.
4. 메소드 정보를 얻는다.
 가. EJBRemoteMethod로부터 리모트 인터페이스의 메소드 정보를 얻는다.
 나. EJBCreateMethod로부터 홈 인터페이스 및 빈 클래스 내에서 사용될 create() 메소드 정보를 얻는다.
 다. EJBFinderMethod로부터 홈 인터페이스에 들어갈 finder 메소드 정보를 얻어 메소드 인터페이스를 구성한다.
5. 1-4에서 결정된 EJB 정보를 이용하여 EJB 구조를 만든다.
 가. 각 인터페이스 및 클래스 이름에 대한 기본 골격 코드와 기본적으로 필요한 기본 클래스 임포트 코드를 생성한다.
 나. 각 인터페이스 및 클래스에 4에서 추출된 메소드 코드를 생성한다.
 다. 엔티티빈인 경우 3에서 추출된 정보를 이용하여 프라이머리 키 클래스를 구성하고 홈 인터페이스 및 빈 클래스 내의 관련 메소드를 수정한다.
6. 각 EJB의 행위에 해당하는 ASL을 분석하여 해당 메소드의 몸체(body)에 추가한다.
 가. 행위 기술 언어로 기술된 비즈니스 로직을 자바 요소에 맞게 변환한다.
 나. 데이터 관련 요소를 엔티티빈 사용 규칙에 맞게 변환한다.
7. 프로파일 내의 요소에 1-6의 과정을 반복하여 모든 EJB 코드를 추출한다.

[그림 4] PSM 프로파일로부터 코드 생성 방법

[그림 4] Cont.

위와 같은 방법으로, EJB 프로파일 및 행위 기술 언어로부터 코드가 완성된 자바 코드를 얻을 수 있다.

4. 결 론

본론에서 살펴 보았듯이, EJB 프로파일로부터 모델링된 EJB의 기본 형태 및 내부 논리 흐름을 포함한 코드 생성에 대한 일련의 과정에 대한 규칙을 알아 보았다.

현재, EDOC으로부터 EJB를 추출해 내는 과정이 완전하게 자동화되어 있지 않으며 가능한 자동화 규칙을 도출해 내는 작업이 OMG를 비롯한 각 단체 및 업체에서 진행중이다.

본 논문에서는 현재 EJB에 대한 PSM 프로파일을 코드로 변환하는 과정에 대한 자동화 단계까지 고려되어 있으나, 모델 내에 함께 포함되는 ASL 에 대한 코드 생성 자동화를 설계할 예정이다.

[참고 문헌]

[1] http://www.omg.org/mda/.
[2] 이우진, 최연준 등, " J2EE 플랫폼에서의 개념적 컴포넌트 모델링 및 컴포넌트 생성 지원 도구 개발" , 제8-D권 제6호 별책, 한국정보처리학회논문지 D, 2001.
[3] OMG, UML Profile for Enterprise Distributed Object Computing Specification Part II Supporting Annexes, OMG, 2001.
[4] OMG, UML Profile for Enterprise Distributed Object Computing Specification, OMG, 2002.
[5] Innoq, " iQgen : Getting Started" , http://iqgen.innoq.org/tutorial_0.html, 2003.