UML Profile 작성시 코드 생성 정보 기술을 위한 메타모델

[1]

김우식[O] 정양재 신규상
한국전자통신연구원 컴퓨터소프트웨어연구소
{wsk[O], yjjung, gsshin}@etri.re.kr

A Metamodel For Code Information Of UML Profile

Woosik Kim[0] Yangjea Jung Gyusang Shin
ETRI - Computer & Software Technology Laboratory

요 약

2001년 OMG는 그 동안의 모델링 관련 표준화 작업의 성과를 바탕으로 모델 중심의 개발 방법인 MDA를 표준으로 정하였다. MDA의 핵심은 잘 정의된 비즈니스 독립적인 모델을 플랫폼 종속적인 모델로 자동 변환하고 그 변환된 모델을 통해서 코드를 자동 생성함으로써 소프트웨어의 생산성을 높이고 플랫폼 변화에 능동적으로 대처 할 수 있다는 것이다. 본 논문에서는 코드 생성을 위해서 플랫폼 종속적 모델의 기술 방법인 UML Profile에 코드 관련 정보를 UML을 통해서 정의할 수 있도록 하는 코드 생성 정보 메타 모델을 제시한다.

1. 서 론

2001년 OMG(Object Management Group)는 그 동안의 모델 관련 표준화 작업의 성과를 바탕으로 모델 중심의 시스템 통합과 개발을 위한 기술 표준인 MDA(Model Driven Architecture)를 정의하였다[1][2].

MDA의 모델 중심의 개발 프로세스는 다음과 같다. 우선 비즈니스 요구만을 플랫폼에 독립적으로 모델링 한 Platform Independent Model(PIM)을 작성하고 이를 변환함으로 실제 플랫폼에 배치 가능한 수준의 정보를 담고 있는 Platform Specific Model(PSM)을 얻어낸다. 마지막으로 PSM을 통해서 코드를 자동으로 생성한다. MDA의 장점은 이러한 개발 과정을 따르고 정확한 PIM만 가지고 있다면 기술 여건이 변화하더라도 PIM으로부터 PSM과 코드를 자동으로 변환하여 얻을 수 있기 때문에 높은 생산성과 재사용성을 유지할 수 있다는 것이다.

이러한 MDA의 핵심적인 요소는 각 단계별 모델 변환의 자동화에 있다. 자동화된 모델 변환과 시스템을 모델링을 하기 위해서 쓰여지는 모델링언어는 내적인 정합성을 유지해야 하며 모델링 대상 도메인의 시맨틱을 올바르게 표현 할 수 있어야 한다. 전자를 위해서 OMG는 UML, CWM, MOF의 코어(Core)를 재정의하는 작업을 수행하고 있으며[3], 후자를 위해서는 UML Profile 기법을 활용하고 있다. UML Profile은 도메인 또는 플랫폼 정보를 올바르게 UML로 표현하기 위해서 UML 모델을 확장하여 표준으로 정한 것이다. 때문에 MDA에서는 해당 플랫폼과 도메인마다 UML Profile을 정의하고 있다.

본 논문에서는 MDA의 모델 변환의 각 단계 중에서 PSM 수준에서 코드의 자동 생성을 위한 UML Profile을 지원하는 메타모델을 정의한다.

본문의 구성은 다음과 같다. 2장에서는 연구배경으로 UML을 위한 코드 생성 메타 모델에 대해서 기술되고, 4장에서는 결론 및 향후 과제에 대해서 논의 된다

2. 연구 배경

UML profile 이란 UML의 기본 building block을 UML 자체의 확장 메커니즘인 Stereotype, Tagged Value와 UML의 Constraints 기술 방법을 이용하여 UML을 특별한 목적에 맞게 tailoring한 것이다. 즉, UML Profile은 특정 도메인의 노테이션에 대해서 UML 표준을 정하여 패키지화 한 것이다.
MDA에서 UML profile이 쓰이는 경우는 다음과 같다.

- PIM 수준에서 특정 도메인에 해당하는 노테이션들을 표현하기 위함 (예, UML profile for EDOC)[4]
- PSM수준에서 해당 기술 플랫폼을 UML로 올바르게 표현하기 위함(예, UML profile for CORBA, UML profile for EJB)[5]

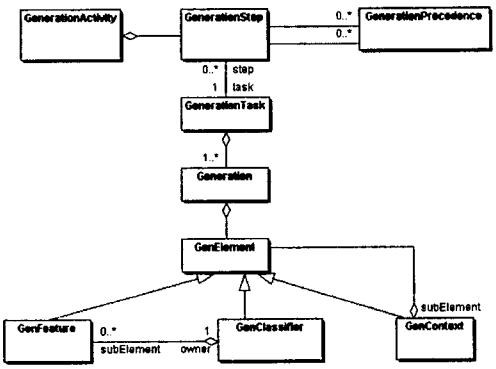
PSM 수준의 UML Profile은 해당 플랫폼과 1:1관계를 가지고 있으며 실제로 프로그래밍 언어인 경우 해당 언어의 BNF를 분석하여 UML Profile을 정의한다. 따라서 UML Profile을 가지고 모델링 된 정보는 UML Profile에 정의된 매핑 정보를 따라서 코드로 생성 될 수 있다.
하지만 UML Profile에 정의된 실제 플랫폼과 UML과의 매핑은 텍스트로 기술 되어 있기 때문에 프로그래머가 스펙을 분석하여 개별 UML Profile마다 코드 생성을 위한 모듈을 작성해야 한다. MDA에 따르면 UML Profile은 플랫폼의 생성에 따라서 지속적으로 생성되게 되는데 그 때마다 해당 UML Profile에 모델 변환이나 코드 생성을 위한 모듈을 개발자는 만들어야 한다. 따라서 UML Profile은 그 내부 정의에 있어서도 표준화 될 필요가 있다. 이를 위해서 본 논문은 PSM 수준의 UML Profile에 코드 매핑 및 생성과 관련 된 정보를 모델링 통해서 표준화 된 방법으로 표시하는 코드 메타 모델을 정의한다.

[1] 본 연구는 2002년 과학기술부 국가지정연구실 사업으로 수행되었음

3. 코드 생성 정보를 위한 메타모델

코드 생성을 위한 메타 모델의 코드 생성 정보는 크게 3가지 정보를 담고 있다.

- 코드 생성 단위: 코드가 어떠한 단위로 생성되어야 하는 지를 나타낸다.
- 코드 생성 위치: 생성될 코드 정보가 어느 위치에 생성 되는지를 나타낸다.
- 코드 생성 순서: 메타 모델의 어떠한 요소들부터 생성이 되어야 하는지를 나타낸다.



(그림 1) 코드 생성 정보 메타모델

위의 세가지 정보를 표현하기 위한 메타 모델은 (그림1)과 같다. 위의 메타모델은 CWM의 변환 메타 모델을 바탕으로 하여 일부 시맨틱을 변경한 것이다[6].

제시된 메타 모델에서 GenerationActivity, GenerationStep, GenerationPrecedence, GenerationTask 등은 CMW 변환 메타 모델의 TransformationActivity, TransformationStep, TransformationTask에 대응되는 개념이며 제시된 메타모델의 개념들은 그 대상만을 변환 관계에서 생성으로 변경하고 코드 생성 과정의 순서 및 테스크 정의는 CWM 것을 그대로 따른다.

노테이션	설명
Generation	코드 생성의 행위단위를 나타낸다.
GenElement	코드로 표현되어야 하는 정보임을 나타낸다.
GenFeature	생성 된 다른 코드 정보와 같이 표현되어야 하는 정보를 나타낸다.
GenClassifier	파일, 패키지 등 코드 생성시 단일하게 하나의 단위로 여겨 질 수 있는 정보를 말한다.
GenContext	코드 생성 정보의 패키징을 위한 노테이션이다.

<표 1> 코드 생성 정보 메타모델 노테이션 설명

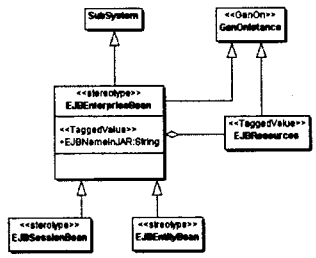
CWM과 다르게 제시된 메타모델은 CWM에 기반한 메타모델간의 변환을 목적으로 하는 것이 아니다. 제시된 메타모델은 UML Profile 내부의 공통으로 사용되어 Profile 요소들 간에 돌어난 코드 생성 정보와 Profile에 텍스트로 기술된 코드 매핑 정보를 UML로 Profile에 추가로 표현함을 목표로 한다.

4. 코드 생성 정보의 기술

4.1 코드 생성 단위 정보의 기술

코드 생성 단위 정보는 UML Profile 작성시에 확장되는 개념들이 GenElement 또는 GenElement의 하위 클래스들을 상속 받음으로 표시된다.

(그림2)는 EJB에 대한 UML Profile 일부에 코드 생성 메타 모델을 이용하여 코드 생성 단위에 대한 정보를 포함 시킨 것이다. " UML Profile For EJB" 에 정의된 EnterpriseBean 클래스는 실제 구현 시에 전개 디스크립터에 포함되는 내용을 UML 클래스로 표현한 클래스이다. 따라서 EnterpriseBean과 그 하위 클래스의 내용들은 코드 생성시에 전개 디스크립터에 쓰여져야 하기 때문에 GenFeature 클래스의 인스턴스를 상속 받는다

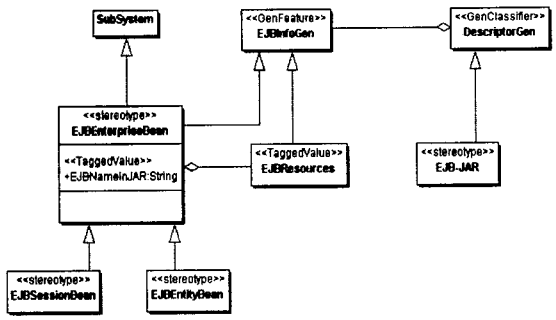


(그림 2) 코드 생성 단위 정보의 기술

UML Profile 작성시에 코드 상의 Attribute 정보와 Configuration 정보는 TaggedValue 형태로 UML 모델 상에 표기 된다. 따라서 TaggedValue로 선언된 값들 또한 코드 생성의 단위가 되어야 한다. 이 정보를 올바르게 표기하기 위해서는 기존의 Profile에 클래스의 Attribute 형태로 선언 되어 있는 TaggedValue를 Profile 메타모델 상에 클래스로 표기하고 해당 TaggedValue를 포함하는 클래스에 Aggregation으로 정의한다. 이러한 작업이 필요한 경우는 하나의 클래스에 표기된 TaggedValue 값들이 다른 코드 생성 목표를 가지는 경우이며 그 외의 경우에는 클래스가 상속 받은 코드 생성 단위 정보를 그대로 이어 받는다.

4.2 코드 생성 관계 정보의 기술

코드 생성 관계 정보는 GenElement와 GenElement 간의 Aggregation 관계로 표현된다.



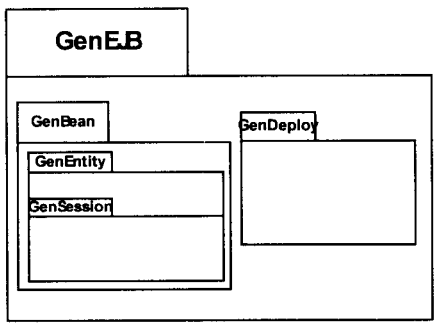
(그림 3) 코드 생성 관계 정보의 기술

(그림3)의 EJB-JAR는 EJB가 전개 될 때 필요한 정보를 담고

있는 XML 파일이다. 따라서 코드 생성 정보 메타모델의 GenClassifier를 상속 받게 된다. 이 XML 파일의 정보는 " UML Profile For EJB" 의 Design View를 통해서 정의된 EJBEnterpriseBean와 그 하위 클래스인 EJBSessionBean, EJBEntityBean에 존재한다. 따라서 이 부분에 대한 코드 단위 정보의 표시인 EJBInfoGen 클래스와 DescriptionGen클래스 간에는 Aggregation 관계가 존재하게 된다.

4.3 코드 생성 정보의 패키징

일반적으로 UML 기반의 모델들은 관련 모델 요소들을 모아서 패키지 단위로 정의한다. 따라서 코드 생성 정보 메타 모델에서도 관련 생성 단위에 따라서 패키지를 정의한다.

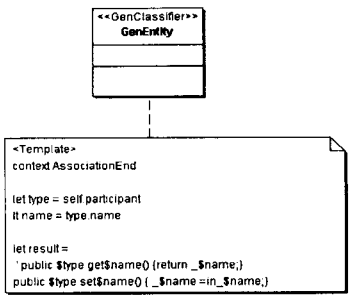


(그림 4) 코드 생성 메타 모델의 패키지 예

코드 생성 패키지의 정의는 모델 추적성과 모델의 부분 생성을 위해서 중요하다.
(그림4)는 EJB의 코드 생성을 위한 메타 모델들을 그 기능 단위별로 나누어서 패키징한 것이다.

4.4 코드 템플릿 정보 추가

제안된 코드 생성 메타 모델에서는 GenClassifier 노테이션이 생성 프로그램 단계에서의 생성 단위가 됨으로 GenClassifier에 코드 생성을 위한 코드 템플릿 정보를 UML Note 노테이션을 이용하여 표시한다.



(그림 5) OCL를 이용한 코드 템플릿 정보 표시의 예

(그림5)는 OCL(Object Constraint Language)를 이용하여 EntityBean 관련 코드 생성을 위한 템플릿을 정의한 예이다. 템플릿에 반드시 OCL을 이용할 필요는 없다. 중요한 사항은 GenClassifier 단위의 템플릿 정의라는 점이다.

5. 결론 및 향후 과제

지금 까지 UML Profile에 모델링 기법을 활용하여 코드 생성과 관련 정보를 정의할 수 있는 코드 생성 메타 모델에 대해서 살펴 보았다.

코드 생성 메타 모델의 장점은 UML Profile 정의 시에 코드 생성과 관련 된 코드 생성 단위, UML Profile에 정의 된 내용의 코드 생성 정보 관계 그리고 관련 코드 생성 요소 정보를 UML Profile의 메타 모델에 적용할 수 있다는 것이다. 이를 통해서 개발자가 새로운 UML Profile을 정의 할 때도 제시된 코드 생성 메타 모델을 이용해서 코드 생성 구조를 정의 할 수 있으며, 이를 해석하여 코드를 생성하는 모듈도 동일한 코드 생성 메타 모델을 통해서 그 재사용성을 높일 수 있다.
향후 과제로는 OCL과 ASL를 통해서 좀더 상세한 코드 생성 정보를 표준화 된 방식으로 정의 하는 것이며 최종적으로는 프라그마틱한 방식으로 UML Profile을 정의할 수 있는 구조와 방법을 모색하는 것이다.

6. 참고 문헌

[1] Object Management Group, " Model Driven Architecture" , OMG document ormsc/01-07-01.
[2] Jon Siegel and the OMG Staff Strategy Group, " Developing in OMG's Model Driven Architecture" , ftp://ftp.omg.org/pub/docs/omg/01-12-01.pdf, November 2001
[3] David S. Frankel, " Model Driven Architecture Applying MDA to Enterprise Computing" , Wiley Publishing Inc, 2003
[4]JSR-000026 UML/EJB™ Mapping Specification 1.0-draft, http://www.jcp.org/aboutJava/community/process/review/jsr026
[5]Object Management Group, UML Profile for Enterprise Distributed Object Computing Specification, OMG Adopted Specification ptc/02-02-05
[6] Object Management Group, " Common Warehouse Metamodel (CWM) Specification, Version 1.0" , OMG document ad/01-02-01