MDA기반 S/W 컴포넌트

최성운 • TTA, IT S/W 기술위원회 S/W응용기술 연구반 위원
명지대학교 컴퓨터공학과 교수
홍선주 • 명지대학교 컴퓨터공학과 박사과정
장진호 • TTA, IT S/W 기술위원회 간사 및 S/W응용기술 연구반 의장
한국전자통신연구원 소프트웨어공학연구부 책임연구원
전인걸 • TTA, IT S/W 기술위원회 S/W개발기술 연구반 간사
한국전자통신연구원 소프트웨어공학연구부 연구원

1. 서론

소프트웨어 컴포넌트(Component : 부품) 기술의 등장은 소프트웨어 산업의 생산성 향상에 새로운 가능성을 제시하고 있다. 이미 제작된 컴포넌트를 조립하여 소프트웨어 시스템을 개발할 수 있다면, 획기적인 생산성 및 품질의 향상을 기대할 수 있기 때문이다. 그러나 실제로 컴포넌트 기술을 소프트웨어 개발에 적용하는데 교육, 경험 등 많은 문제점이 발생하고 있으며, 컴포넌트 기술의 이해 부족으로 생산성의 향상을 가져오지 못하는 경우도 종종 발생하고 있다. 이러한 현상은 새로운 기술도입 시 항상 발생하고 있으며, 컴포넌트 기술도입 또한, 산업계에서 의문 시 하고 있는 실정이다.

하지만 컴포넌트 기술이 소프트웨어 생산성 및 품질향상을 가져 올 수 있다는 데에는 이론의 여지가 있을 수 없다. 왜냐하면 소프트웨어 시스템과 하드웨어 시스템의 개발 및 생산공정은 유사한 구조를 가지고 있으며, 하드웨어산업의 오랜 역사를 통해, 컴포넌트 기술은 이미 검증되었기 때문이다. 다만 소프트웨어 산업에 컴포넌트 기술을 도입하기 위해서는 소프트웨어의 특성을 고려한 보다 체계적이고 조직적인 방법이 요구되고 있다.

2. S/W 컴포넌트 기술동향

2.1 소프트웨어 시스템의 특성

컴포넌트를 조립함으로써 소프트웨어 생산성의 향상을 기대할 수 있다는 가정은 소프트웨어 및 하드웨어 시스템 개발 시 동일하게 적용될 수 있다. 하지만 소프트웨어 시스템의 경우 하드웨어 시스템의 조립생산과는 차이가 있다. 하드웨어 시스템의 조립생산의 문제는 동일한 시스템을 대량생산하는 단계에 집중되어 있는 반면, 복제가 용이한 소프트웨어 시스템의 경우에는 그 초점이 새로운 시스템을 개발하는 단계에 집중되어 있기 때문이다. 즉, 소프트웨어 복제를 통하여 개발된 소프트웨어 시스템의 대량생산을 용이하게 할 수 있는 것이다. 다시 말하면 소프트웨어 조립생산의 문제는 대량생산의 문제가 아니라, 특화된 요구에 적합한 다양한 시스템들의 개발문제인 것이다.

2.2 컴포넌트 기술의 본질

소프트웨어 컴포넌트 기술은 다양한 시스템에 개별적으로 적용되어야 하는 기술이다. 이러한 이유로 컴포넌트 기술은 서로 다른 기종의 컴퓨터 및 운영체제에 공통적으로 적용될 수 있어야 한다. 또한 컴포넌트 자체의 개발보다는 컴포넌트를 조립하기 위한 시스템의 구조(Architecture)개발에 초점을 맞추어야 한다. 컴포넌트 구조는 컴포넌트가 조립되어야 할 틀이며, 이 구조 속에서 컴포넌트의 역할 및 인터페이스가 정의되기 때문이다. 컴포넌트 기술은 컴포넌트를 제작하는 기술이 아니라 시스템의 구조에 맞추어 관련 도구들을 이용하여 컴포넌트를 조립하는 기술이다.

2.3 미들웨어 프레임워크(Middleware Framework)

미들웨어 프레임워크는 이기종의 컴퓨터 환경 즉 분산 환경하의 컴포넌트 간의 연동(Inter-operability) 및 조립 구조를 제공한다. 즉, 분산된 컴포넌트들이 미들웨어 프레임워크를 통해 서로 통신하고 작동되는 것이다. 컴포넌트 기술은 이러한 미들웨어 프레임워크가 개발되면서 본격적으로 소프트웨어 시스템 개발에 적용되기 시작하였다. 미들웨어 프레임워크의 등장으로 인해 동종의 시스템에서 뿐 아니라 이기종의 시스템에서도 컴포넌트의 재사용이 가능해졌기 때문이다. 현재 개발된 미들웨어 프레임워크 제품으로는 OMG(Object Management Group)의 CORBA(Common Object Request Broker Architecture) 제품들, Sun사의 J2EE(Jave 2 Platform, Enterprise Edition) 제품들 그리고 Microsoft사의 .NET 등이 있다.

2.4 미들웨어 프레임워크의 한계

미들웨어 프레임워크의 등장과 함께 컴포넌트 기술은 구조적 체계를 갖추게 되었으며, 재사용성을 향상시킬 수 있게 되었다. 하지만 미들웨어 프레임워크 또한 한계에 부딪치게 되었다. 전체 개발 노력의 30% 미만을 차지하는 구현 단계만의 생산성 향상으로는 전체 소프트웨어 생산성 향상의 문제를 해

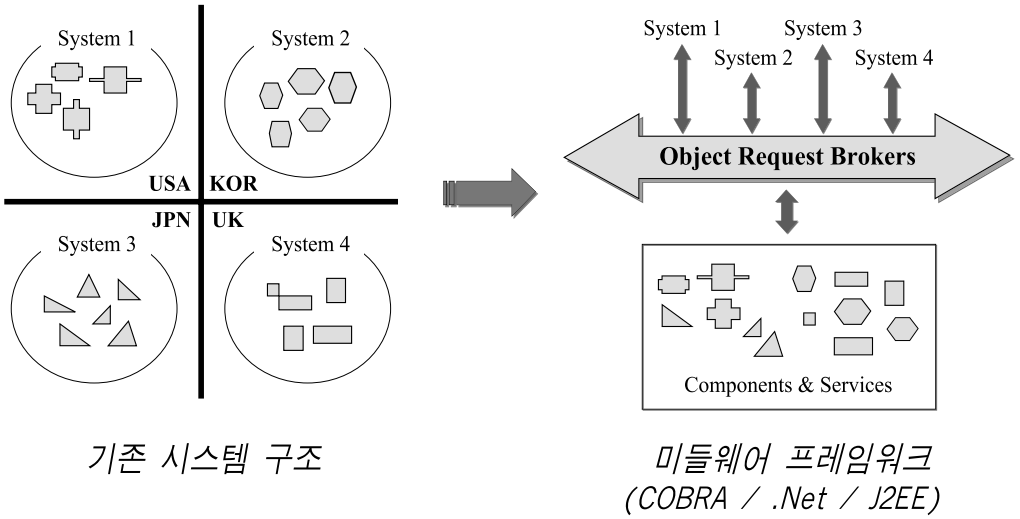

그림 1. 미들웨어 프레임워크

결할 수 없기 때문이다. 또한 여러 종류의 미들웨어 프레임워크가 등장함에 따라 또다시 미들웨어 프레임워크간의 연동 문제에 봉착하게 되었다. 객체 표준화 기구인 OMG에서는 CORBA 사양(Specification)을 중심으로 미들웨어 프레임워크간의 연동 구조를 정의하고 있으나 아직 .NET과의 호환은 이루어지지 않고 있다. 컴포넌트 기술의 성공적인 정착을 위해서는 소프트웨어 라이프사이클(Lifecycle) 전체 및 여러 종류의 미들웨어 프레임워크를 지원하는 표준 모델의 정의가 시급한 실정이다.

2.5 컴포넌트 기술의 방향

컴포넌트 기술은 미시적으로 컴포넌트를 조립할 수 있는 구조, 관련 컴포넌트 및 이러한 구조를 분산 환경 하에서 유연하게 구현할 수 있는 미들웨어 프레임워크 등을 포함한다. 거시적으로는 컴포넌트 기반 시스템의 개발, 관리, 구현, 유지보수, 배포 등 소프트웨어 라이프사이클 전반에 걸친 기술을 포함한다. 현재 컴포넌트 기술은 구현뿐 아니라 관련 기술 전반을 포함하는 포괄적인 표준 구조를 정의하는데 그 초점을 맞추고 있다. 미들웨어 프레임워크의 한계에서 보았듯이 구현을 중심으로 컴포넌트 기술의 표준 구조를 정의하는 것은 시장적 측면에서 불가능할 뿐 아니라, 호환성 측면에서도 문제가 있기 때문이다. 컴포넌트 기술의 핵심인 기술요소들 간의 포괄적 호환성을 유지하기 위해서는 이에 관련된 모든 요소들을 포함하는 표준 구조의 정의가 필수적이라 할 수 있다.

3. OMG의 MDA 표준화 동향

3.1 MDA

현재 컴포넌트 기술의 표준 구조는 OMG의 MDA(Model Driven Architecture)에 의해 제시되고 있다. MDA는 모든 컴포넌트 기술요소의 표준 메타 모델(Meta Model)을 정의하고 이를 기반으로 각 구성요소를 정의함으로써 모든 컴포넌트 기술요소들의 호환성 및 시스템 간 동작성을 보장하고 있다.

MDA의 핵심은 메타 모델을 기반으로 구현 환경에 독립적 모델(Platform Independent Model : PIM)을 자동으로 각 구현 환경에 적합한 구현 종속 모델(Platform Specific Model : PSM)로 변환할 수 있는 구조를 정의하는 것이다. 즉 표준 메타 모델을 기반으로 구현 환경에 독립적인 시스템을 개발하고 이를 자동으로 구현 환경에 배치함으로서, 구현 단계에서의 생산성 향상뿐 아니라, 표준 메타 모델에 기반을 둔 시스템들의 일반적인 호환성을 기대할 수 있게 되는 것이다.

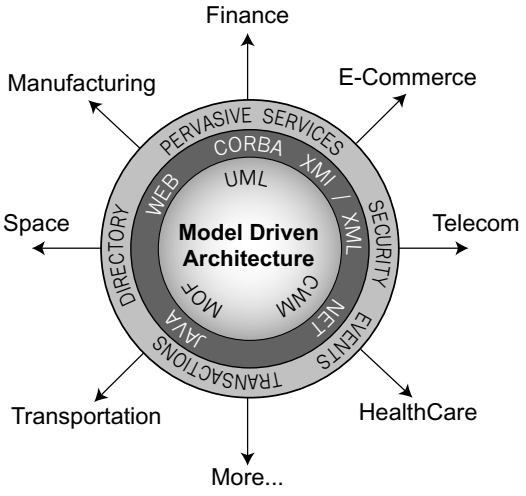

그림 2. Model Driven Architecture

3.2 MDA 구조

3.2.1 핵심기술

MDA는 다음과 같은 네가지 핵심기술을 근간으

로 하여, 기본 서비스(Pervasive Services) 및 도메인 특성(Domain Facility)들로 구성된다.

1) MOF(Meta Object Facility)

MOF는 객체 및 컴포넌트 기술의 핵심을 정형화한 모델로서, 객체지향 본질 그 자체이다. MOF는 객체지향 모델을 작성하기 위해 사용되는 메타모델의 필수요소와 문법, 구조를 정의하는 메타메타 모델이다. MDA에서 MOF는 CWM(Common Warehouse Metamodel)이나 UML(Unified Modeling Language) 메타모델에 대한 공통 모델로서 제공된다. OMG에서 제공하는 MOF 사양은 다음과 같은 내용을 포함한다.

- 일반적인 MOF 객체 및 관계에 대한 추상 모델
- MOF 기반의 메타모델과 언어 독립적인 인터페이스(CORBA IDL로 정의됨) 간의 매핑 규칙
- MOF 기반의 메타모델로부터 생성된 모델을 다루기 위한 일반적인 오퍼레이션들

MOF의 시멘틱(Semantic)은 일반적으로 메타모델로부터 모델의 구축, 전개(Discovery), 탐색(Traversal) 및 갱신을 지원하는 메타데이터 리파지토리 서비스를 정의한다. 특히 MOF의 구현은 효과적인 메타데이터 작성 및 자동화 도구 개발 시 이용될 수 있다는 점에서 의미를 갖는다. MOF의 정형화된 내용은 MDA 모델들을 위한 표준 저장소를 위해 사용된다. UML의 클래스 다이어그램으로 표현한 MOF 모델은 다음 그림 3과 같다.

2) UML(Unified Modeling Language)

UML은 객체 및 컴포넌트 시스템을 표현하기 위한 표준 언어로서 시스템의 아키텍쳐 및 구성 객체, 객체간의 상호작용 등을 모델링하기 위해 사용된다.

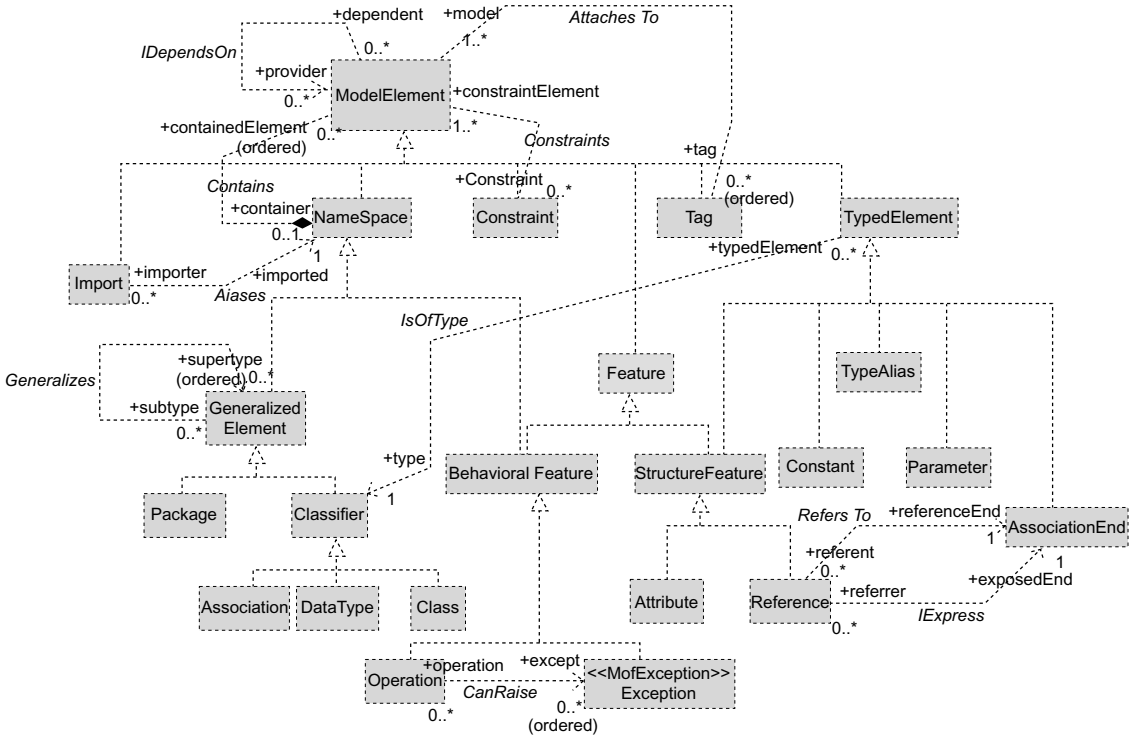

그림 3. MOF 모델 클래스 다이어그램

UML은 시스템의 분석 및 설계 시에 구현환경에 무관하게 표준화된 방법으로 시스템을 모델링할 수 있게 한다. 시스템의 정적인 구조를 나타내기 위해 UML에서는 네가지 형태의 다이어그램-클래스 다이어그램, 객체 다이어그램, 컴포넌트 다이어그램, 배치 다이어그램-을 제공한다. 또한 시스템의 동적인 행위를 모델링하기 위해서는 다섯가지 형태의 다이어그램-유스 케이스 다이어그램, 상태챠트 다이어그램, 순차 다이어그램, 활동 다이어그램, 협동 다이어그램-을 제공한다. 각종 UML 다이어그램들은 UML 메타모델을 통해서 정교하게 정의가 되어 있으며, 가시적인 UML 모델들은 다른 정형적인 언어로 자동변경이 가능하다.

그림 4는 MOF와 UML 모델들간의 관계를 나타낸다. 이러한 4계층 메타모델링 구조는 MDA기반 시스템 모델링의 기본 구조이다. MDA에서 UML은 구현 독립적인 모델이나 구현 종속적인 모델 작성 시에 사용되기도 하며, MOF나 CWM을 표현하기 위한 기호로서의 기반을 제공하기도 한다. MDA기반 시스템 개발의 첫번째 단계가 UML을 이용하여 구현 독립적 모델을 작성하는 것이다.

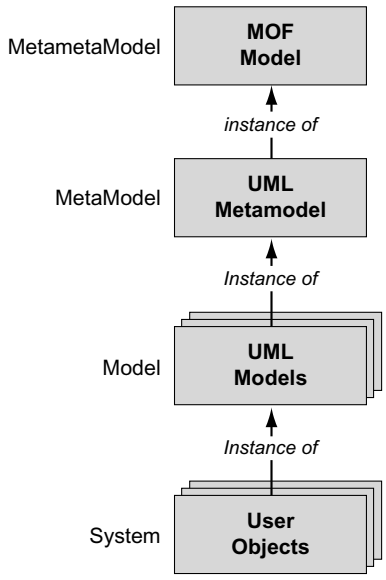

그림 4. 4-계층 메타모델링 구조

UML은 모델링 영역의 경계를 확립하는 프로파일(Profile)에 의해 다양한 플랫폼에 대하여 구체적으로 특화될 수 있다. 프로파일이란 UML의 기본 모델을 특화하거나 확장하기위해 제공되는 메커니즘(태그 값, 스테레오 타입)을 말한다. 이러한 프로파일은 누구나 정의할 수 있으나, CORBA나 EDOC (Enterprise Distributed Object Computing), EAI (Enterprise Application Integration)을 위한 OMG의 표준 프로파일과 같이 다수의 사람들이나 기업에 의해 정의된 프로파일이 가장 유용하게 사용된다. 이렇게 표준화된 프로파일은 구현 독립모델로부터 구현 종속 모델을 생성하거나, 구현 종속 모델로부터 코드를 생성하기 위한 규칙들을 제공하게 된다.

3) XMI(XML Metadata Interchange)

XMI는 XML(eXtensible Markup Language) 기반 데이터 관리를 위한 표준으로 MOF 기반 모델을 XML로 매핑하기 위한 표준 사양이다. 구축된 모델이나 메타 모델들을 유용하게 사용하기 위해서는 이러한 모델들이 다양한 도구들 간에 혹은 도구와 리파지토리 사이에서 호환이 가능해야 한다. 이러한 것을 가능하게 하기 위해 OMG에서 표준화 한 것이 XMI이다. UML과 MOF, XML에 기반하는 XMI는 DTD(Document Type Definition)와 스키마를 통해 XML로 메타모델과 모델의 표현(Representation)을 표준화 한다. XMI DTD와 스키마는 모델들에 대한 메타데이터가 된다.

XMI는 객체와 객체간의 관계를 나타내기 위해 태그 기반의 언어를 사용하며, XML의 태그가 MOF 기반의 모델을 XML로 표현하는데 어떻게 사용되는지를 정의한다. MOF 기반의 메타모델은 XML DTD로 번역되며, 모델은 XML 문서로 변환된다.

4) CWM(Common Warehouse Metamodel)

CWM은 데이터 웨어하우징이나 비즈니스 분석 영역에서 주로 나타나는 비즈니스 메타데이터와 기술적 메타데이터를 표현하는 표준 메타모델이다. CWM은 이질적인 멀티 벤더 소프트웨어 시스템 간에 메타데이터의 인스턴스를 교환하기 위한 기반을 제공한다. CWM은 자료 저장소(Data Repository) 통합을 위한 산업표준이며, 데이터베이스 모델(스키마), 스키마 변형 모델, OLAP(On-Line Analytical Processing), 데이터 마이닝 모델들의 표현방법을 표준화한다.

시스템 간에 공유되는 메타데이터는 일관된 하나 이상의 CWM 메타모델의 관점에서 정형화된다. CWM을 지원하는 도구를 사용하면 데이터 웨어하우스 모델로부터 구체적인 데이터 웨어하우스의 인스턴스가 생성될 수 있을 것이다.

MOF와 CWM은 애플리케이션과 데이터 모델링, 결과를 저장하기 위한 리포지토리를 위한 기반을 형성하며, UML은 애플리케이션 모델을 구축한다(CWM은 그중에서도 데이터 모델을 구축하는 방법을 정의한다). XMI는 툴과 리포지토리 사이에서 모델의 호환을 담당한다.

MDA는 이러한 핵심 모델들을 기반으로 Web, Java, CORBA 등 각 구현 플랫폼으로의 변환을 위한 구조 및 분산 시스템에 필요한 트랜잭션(Transaction), 이벤트(Event) 등의 기본 서비스(Pervasive Services)를 정의하고 있다.

3.2.2 기본 서비스(Pervasive Service)

분산 애플리케이션 구현을 위한 모든 플랫폼들은 공통적인 몇몇 필수 서비스를 제공하고 있다. 이러한 서비스들의 종류는 다양하지만 일반적으로 디렉토리 서비스(Directory Service), 이벤트 핸들링(Event Handling), 지속성(Persistence), 트랜잭션(Transaction), 보안(Security) 등을 포함한다. 이러한 서비스들이 특정 플랫폼에 맞춰 정의되거나 구축될 경우 해당 플랫폼에 한정되는 특성을 갖게 된다. 이러한 것을 피하기 위해 OMG는 UML의 구현 독립 모델(PIM) 레벨에 해당하는 기본서비스를 정의하였다. 구현 독립모델 작성 시 기본서비스의 내용을 모델에 반영하며, 구현 종속모델 작성 시 MDA에 의해 지원되는 모든 미들웨어 플랫폼에 대한 내용이 반영된다.

플랫폼 독립적인 비즈니스 컴포넌트 모델에서 기본서비스들은 매우 개략적으로 나타난다. 특정 플랫폼에 대하여 구현 종속모델이 작성되면 해당 플랫폼

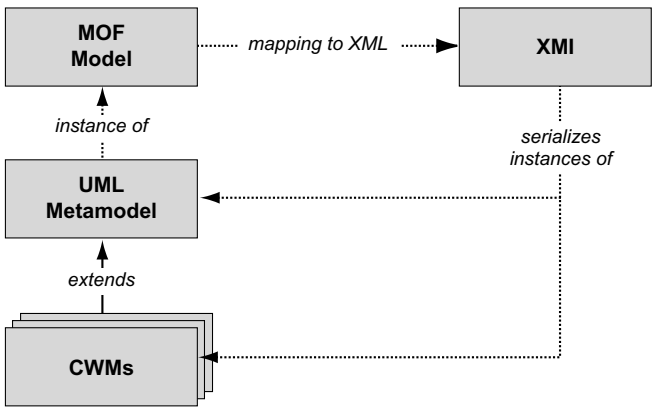

그림 5. MDA 핵심기술간의 관계

고유의 서비스를 호출하도록 코드가 생성될 것이다. 이러한 기본서비스들은 OMG를 구성하는 PTC (Platform Technical Committee)나 DTC(Domain Technical Committee)에 의해서 정의 · 표준화된다.

3.2.3 도메인 사양(Domain Specification)

OMG 활동의 많은 부분이 DTC의 DTF(Domain Task Force)를 통해 특정 도메인 시장을 위한 서비스나 특성(facility)을 표준화 하는데 초점을 두고 있다. 초기에는 이러한 명세들이 OMG의 IDL (Interface Definition Language)로 쓰여진 인터페이스들로 구성되었다.

MDA에서 OMG의 도메인 특성 명세는 유용성과 영향력을 최대화하기 위해 UML을 사용하여 표현된 구현 독립모델(PIM)의 형태이며, 적어도 하나의 특정 플랫폼에 대한 인터페이스 정의와 UML을 사용하여 표현된 구현 종속 모델(PSM)에 의해 보강될 것이다. MDA의 공통적인 원칙들은 구현 코드의 일부 생성을 가능하게 할 것이다. DTC의 DTF는 해당 애플리케이션 분야의 표준 기능을 위한 표준 프레임워크를 생성한다. 예를 들어, 수납 계정(Account Receivable)을 위한 Finance DTF 표준은 UML을 사용하여 표현된 구현 독립모델과 CORBA 플랫폼 기반 UML모델, IDL 인터페이스, Java 플랫폼 기반 UML 모델과 Java 인터페이스를 포함한다.

3.3 MDA기반 개발공정

MDA기반 개발공정은 자동화 도구를 기반으로 다음과 같은 단계로 이루어진다.

- 1단계 : 구현 독립적 모델 작성
- 2단계 : 구현 종속적 모델 작성
- 3단계 : 애플리케이션 생성

1) 구현 독립모델(Platform Independent Model : PIM)

모든 MDA 기반 프로젝트는 그림 6의 상단에서

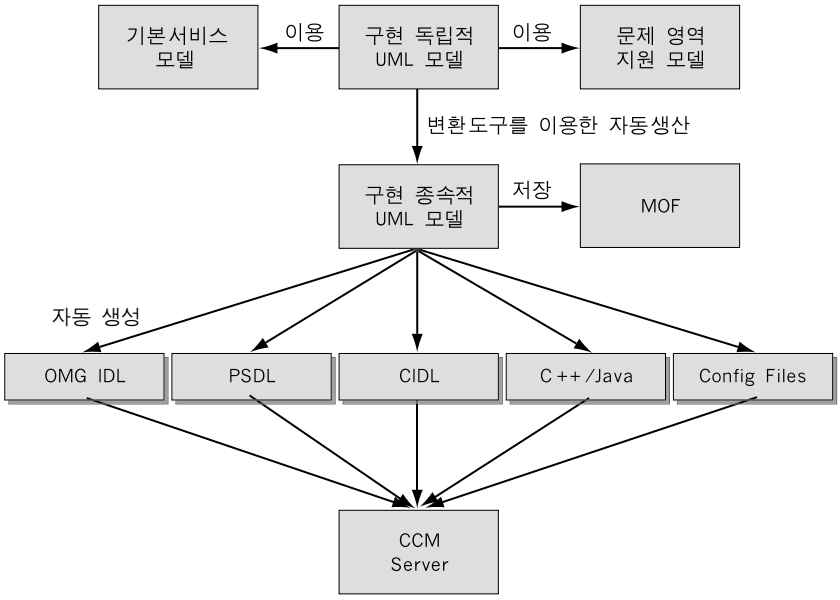

그림 6. MDA 기반 CCM 서버 생성단계

보는 바와 같이 UML을 이용하여 구현 독립적 모델(PIM)을 작성하는 것으로부터 시작된다. 구현기술과는 무관한 비즈니스 기능과 행위를 반영하는 MDA의 구현 독립적 모델은 시스템 프로그래머보다는 비즈니스 영역 전문가에 의해 구축되는 것이 바람직하다. UML의 클래스 및 객체 다이어그램은 시스템의 정적 구조를 구체화시키며, 순차 및 활동 다이어그램은 시스템의 행위를 구체화시키는데 사용된다. 구현 독립적 모델에도 그 정련(Refine) 정도에 따라 몇 레벨이 존재하며, 보다 정련된 구현 독립적 모델은 일반적인 플랫폼의 내용을 반영하는 몇몇 행위를 포함한다.

구현 독립적 모델작성 시에 기본서비스(Pervasive Servie) 및 도메인 특성(Domain Facility)에 대한 내용을 포함하게 되는데, 일반적으로 MDA 애플리케이션 모델링 자동화 도구들이 이러한 기본모델을 제공하게 된다. MDA에서는 서버의 내용뿐 아니라 클라이언트 환경의 모델링도 가능하다. 어떤 플랫폼이나 아키텍쳐에 제한적이지 않게 MDA는 웹 브라우저, JVM(Java Virtual Machine), CORBA 등의 클라이언트를 실질적으로 모델링 할 수 있다.

2) 구현 종속적 모델(Platform Specific Model : PSM)

구현 독립적 모델작성이 완료되면 MOF로 저장되고 이 내용은 구현 종속적 모델을 생성하기 위한 매핑 과정의 입력물로 사용된다. 자동화 도구(Mapping Tool)는 구현 독립적인 모델을 구현 종속적(Platform Specific) UML 모델로 변환한다. UML에서 제공하는 제한 및 확장 메커니즘은 MDA에 의해 요구되는 구현 독립적 모델 및 구현 종속적 모델의 세부적인 표현을 가능하게 한다. UML 프로파일(Profile)이라 일컬어지는 표준화된 확장 메커니즘의 집합(스테레오 타입과 태그 값)은 특정 환경이나 특정 플랫폼 등의 한정된 사용을 위해 UML 모델을 특화할 수 있도록 한다. CORBA기반의 UML 프로파일은 2000년에 OMG에 의해 채택되었다. 현재 기타 플랫폼들에 대한 프로파일의 채택작업이 진행 중이다.

구현 종속적 모델을 생성하기 위해서는 먼저 목표 플랫폼을 선택하여야 한다. 선택하는 플랫폼은 단일 플랫폼일 수도 있고 다중 플랫폼일 수 있다. 매핑단계가 수행되는 동안 일반적인 애플리케이션 모델로

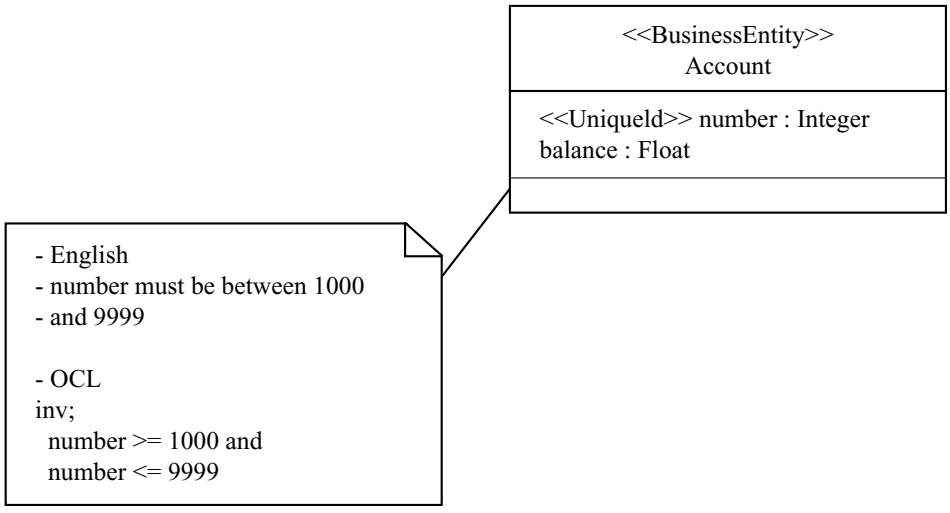

그림 7. 구현 독립적 모델의 예

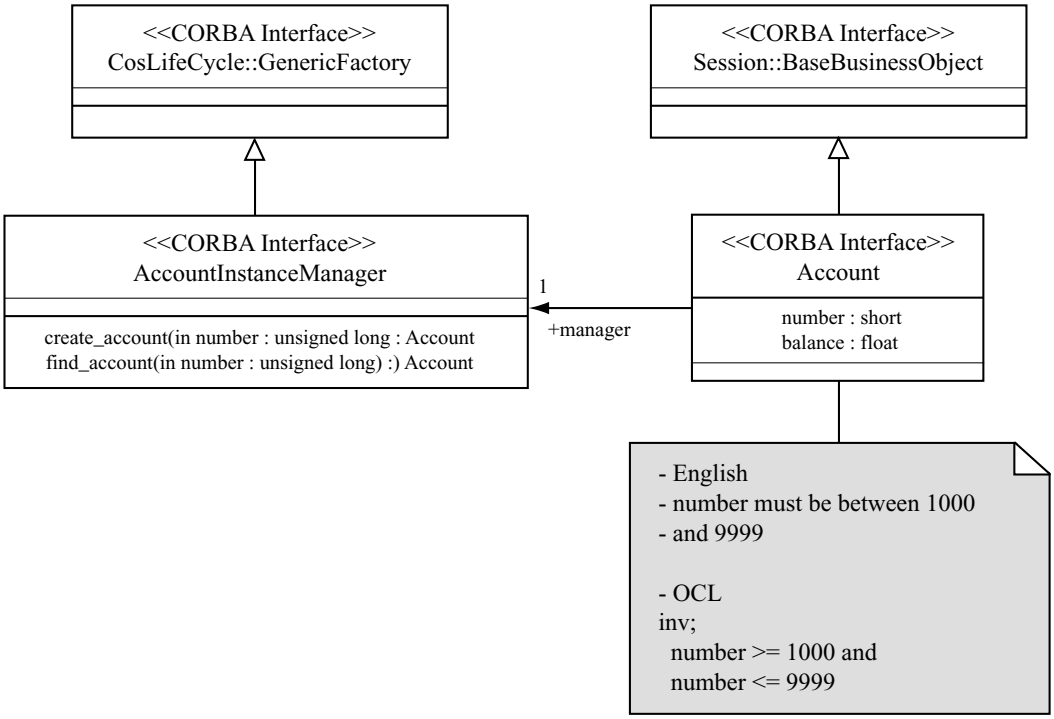

그림 8. 구현 종속적 모델의 예

설계된 시스템의 실행 특성과 구성(Configuration) 정보들은 목표 미들웨어 플랫폼에 의해 요구되는 특정한 형태로 변환된다.

OMG의 표준 매핑규칙을 기반으로 하는 자동화 도구들은 가능한 많은 부분의 변환을 자동으로 수행하지만, 프로그래머가 모호하게 생각할 수 있는 부분은 분석가가 직접 수정해야 한다. MDA의 초기 버전에서는 상당한 양의 코딩이 요구되었으나 UML 프로파일 및 매핑규칙이 점차 성숙해짐에 따라 그 양이 점차 줄어들고 있다. MDA 알고리즘과 이를 지원하는 자동화 도구들의 기능이 향상되는 만큼 애플리케이션 설계자들의 능력도 향상될 것이며, 모델의 변환은 점차 완전 자동화될 것이다.

3) 애플리케이션 생성

자동화 도구(Code Generation Tool)를 이용하여 구현환경에 적합한 프로그램을 만들어 낸다. 구체적인 기능 등은 직접 프로그래밍 한다. 뿐만 아니라 MDA기반 자동화 도구들은 특정 플랫폼에서 요구하는 모든 파일을 생성해 준다.

3.4 MDA 장점

MDA기반 시스템 개발공정에서 알 수 있듯이, MDA는 메타모델을 이용하여 대부분의 구현공정을 자동화할 수 있는 구조를 제공한다. 이러한 MDA기반 개발방법은 시스템의 구현결과 뿐 아니라, 분석 설계 관리 등 프로젝트 진행 전체결과를 재사용 가능하게 한다. 또한 시스템이 구현환경과 독립적으로 정의되므로, 이식성(Portability)을 증가시킨다. 시스템 구조가 구현환경으로부터 독립됨으로써 개발된 시스템의 라이프사이클이 구현 종속적인 시스템에 비해 증가한다.

4. 결론

컴포넌트 기술의 도입은 소프트웨어 생산성을 획기적으로 향상시킬 수 있는 유일한 방안으로 대두되고 있다. 하지만 컴포넌트 기술의 도입이 컴포넌트를 몇 개 만든다고 해결될 일이 아니다. 앞서 언급한 바와 같이 컴포넌트 기술은 구조적 통합적 기술이므로, MDA 등과 같은 표준 구조를 기반으로 장기적 안목을 가지고 점진적으로 추진되어야 한다. 즉, 컴포넌트 기술의 도입은 CMM(Capability Maturity Model) 혹은 SPICE(Software Process Improvement and Capability dEtermination) 등의 소프트웨어 프로세스 평가모델이 제시하는 바와 같은 점진적 단계를 거쳐 도입되어야 한다. 컴포넌트 기술의 도입은 조직의 기술적, 관리적, 재정적, 문화적 문제를 수반하기 때문이다. 참고적으로 Butler Group이 제안한 컴포넌트 기술도입의 단계는 다음과 같다.

- 1단계 : 사용자 인터페이스 컴포넌트 정도의 이용단계
- 2단계 : 구현환경에서의 컴포넌트 이용단계
- 3단계 : 분석 및 업무에서의 컴포넌트 이용단계
- 4단계 : 컴포넌트 재사용의 단계
- 5단계 : 기업업무 전반에 걸친 컴포넌트 이용단계
- 6단계 : 성숙단계

현재 국내에서는 KCSC(Korea Consortium for Software Component Promotion)를 중심으로 컴포넌트 기술을 적극적으로 도입하려는 시도가 이루어져, 관련 기업들이 2단계로 진입하려 하고 있고, 미국의 경우는 OMG를 중심으로 많은 기업들이 3-4단계로의 진입을 시도하고 있다. 이러한 상황을 고려해 볼 때 국내 컴포넌트 기술의 조속한 정착을 위해서는 MDA 등과 같은 컴포넌트 기술의 표준 구조를 기반으로 미국 등과 같은 선진국들이 겪은 시행착오를 최소화하여야 할 것이다. 따라서, 컴포넌트 자체의 개발기술보다는 컴포넌트 구조를 중심으로 점진적, 장기적 기술도입을 추진하여야 할 것이다.

참고문헌

[1] OMG Model-Driven Architecture Home Page: http://www.omg.org/mda/index.htm

[2] OMG Architecture Board MDA Drafting Team, "Model-Driven Architecture: A Technical Perspective", ftp://ftp.omg.org/pub/docs/ab/01-02-01.pdf

[3] Desmond Dsouza, Enterprise Integration , Software Development, 1999

[4] OMG Meta Object Facility Specification, Version 1.3, September, 1999.
http://www.dstc.edu.au/Research/Projects/MOF/rtf/.
http://www.omg.org/.

[5] Tolbert, D., "CWM: A Model-Based Architecture for Data Warehouse Interchange", Workshop on Evaluating Software Architectural Solutions 2000, University of California at Irvine, May, 2000.
http://www.cwmforum.org/uciwesas2000.htm

[6] Object Management Group, XML Metadata Interchange Specification, Version 1.1, http://www.omg.org/.

[7] D'Souza, D., "Model-Driven Architecture and Integration: Opportunities and Challenges", Version 1.1.
http://www.catalysis.org/publications/

papers/2001-mda-reqs-desmond-6.pdf
[8] OMG, UML Profile for EDOC RFP Home Page: http://cgi.omg.org/techprocess/meetings/schedule/UML_Profile_for_EDOC_RFP.html

전화기 자판배열 표준화 문자입력방식 통일 추진

무선인터넷 활성화와 이용자들의 불편을 해소하기 위해 휴대전화 문자입력방식이 표준화된다. 한국정보통신기술협회(TTA 사무총장 임주환)는 정통부가 그동안 중단된 '전화기자판 한글배열 표준화' 작업을 재추진하기로 하고 검토작업에 들어갔다고 3월 12일 밝혔다. TTA는 무선 데이터 활성화를 위해 이용자 불편사항을 취합, 이를 해소하기 위해 최근 휴대전화 충전기 표준화를 추진한데 이어 업체간 상이한 방식으로 이용자들이 불편을 겪고 있는 휴대폰 문자입력방식을 통일하기로 했다. 이를 위해 TTA는 그동안 추진해 왔던 '전화기자판 한글배열 표준화 추진 현황자료'를 정보통신부에 제출했다. 문자입력 방식 통일작업은 이용자 불편 해소외에 전화기 등 각종 단말기를 정보단말기로 활용할 수 있는 기반을 마련하고 한글배열의 독자적 개발로 인한 업체간 중복투자를 개선하는 효과가 있을 것으로 예상된다. 현재 휴대전화의 문자입력 방식은 삼성전자가 '천지인', LG전자가 '이지한글 '을 개발해 휴대전화에 적용하고 있다. 한편 TTA가 수행한 '전화기자판 한글배열 표준화' 작업은 재일교포가 제안한 한글대응 전화번호 연상암기용 방식과 KT에서 제안한 전화기이용 문자입력용방식에 대해 검토작업을 진행해왔다.

새 XML 디지털서명 표준 승인

인터넷 표준화기구인 월드와이드웹컨소시엄(W3C)이 보안성 및 확장성을 극대화한 새로운 XML기반 디지털 서명 표준을 승인했다고 로이터통신이 최근 보도했다. 'XML 시그너처 신택스 앤드 프로세싱(XML Signature Syntax and Processing)'이라 불리는 이 표준은 보안성은 물론 기존 XML기술을 활용하기 때문에 사용 편의성이 높다. W3C는 이 표준을 적용할 경우 문서의 필요한 부분에만 서명하는 것이 가능하기 때문에 웹기반 애플리케이션에 유연성을 확보할 수 있다고 설명했다. 이와 함께 이 표준은 단순한 텍스트파일 뿐 아니라 'bmp'이나 'jpg' 같은 기술로 압축된 그래픽 및 이미지 파일에도 적용된다. 특히 새 표준은 문서전송에 있어 정보가 실린 파일의 타당성 확인과정에서 정보를 개방해 읽고 재전송하는 것이 가능하다. 이번 새표준 승인으로 향후 이 표준을 이용한 디지털 서명이 급속히 확산될 것으로 전망된다. 마이크로소프트(MS)의 XML웹서비스 부문 책임자인 로버트 워비는 "MS는 이 표준을 강력하게 지지한다"면서 "비주얼 스튜디오 닷넷은 이 기능을 지원할 것"이라고 밝혔다. MS는 이미 이 표준을 최근 선보인 비주얼 스튜디오 프로그래밍 툴에 내장한 것으로 알려졌다. 한편, 이번 표준 승인작업에는 W3C와 인터넷엔지니어링태스크포스(IETF), MS · IBM · 베리사인을 비롯해 볼티모어 테크놀로지스와 시티그룹 · 모토로라 · 선마이크로시스템스 등이 참여했다.