제 18 권 제 1 호
Vol. 18 No. 1

2016

제 18회 한국 소프트웨어공학 학술대회 논문집

Proceedings of the 18th Korea Conference on Software Engineering (KCSE 2016)

- **일시: 2016년 1월 27일(수) ~ 1월 29일(금)**
- **장소: 강원도 평창 한화리조트(휘닉스파크점)**

주최: 한국정보과학회, 한국정보처리학회
주관: 한국정보과학회 소프트웨어공학 소사이어티
한국정보처리학회 소프트웨어공학 연구회
한국전자통신연구원
후원: ㈜솔루션링크, ㈜코스콤, ㈜모아소프트, ㈜비트컴퓨터, 소프트웨어공학엑스퍼트그룹㈜, ㈜엔에스이, 한국산업기술시험원, 무인자율 및 적응형 SW 연구단, ㈜다한테크, SW 상시모니터링기술연구단, 시스트란 인터내셔널, ㈜티큐엠에스, ITRC 고품질융합소프트웨어연구센터, 슈어소프트테크㈜, ITRC 소프트웨어안전성보증연구센터, STA 테스팅컨설팅㈜, ㈜이에스지, 정보통신산업진흥원 소프트웨어공학센터, ㈜테스트마이다스, TTA 소프트웨어시험인증연구소

	논문 발표 E			
	E1: SW 테스팅 3 좌장 이정원 교수(아주대) 장소: 그랜드홀 2	**E2: 정적 분석** 좌장 유준범 교수(건국대) 장소: 세미나실 1	**E3: SW 융합** 좌장 이찬근 교수(중앙대) 장소: 세미나실 2	**E4: SW 품질 3** 좌장 이병정 교수(서울시립대) 장소: 세미나실 3
10:50-12:10 (80 분)	DO-178C 기반 소프트웨어 동적 테스트 방안 [산업체논문] 윤상은, 이종민, 정영은(TTA) 안전 무결성 기준을 활용한 테스트 케이스 우선순위 [단편논문] 송옥수, 김보운, 최병주(이화여대) 확장된 라운드 트립 기법을 이용한 발사통제장치 테스트 기법 [단편논문] 배정호, 안세준, 장부철, 구봉주 (국방과학연구소) 안드로이드 어플리케이션 개발 생산성을 위한 권한 자동 검사 기법 [단편논문] 노승학, 인호(고려대)	안드로이드 이미지 로딩 라이브러리 비기능 품질 속성 비교 조성래, 정연철, 민상윤(KAIST) 코드마인드: 온더플라이 소스코드 정적분석 도구 [산업체논문] 신승철, 이욱세, 노상훈, 김제민(코드마인드) 유사도 메트릭을 이용한 컴포넌트 상호연관성 추출 [우수단편논문] 이재권, 김기섭, 정우성(충북대)	다양한 이해관계자들 간 정보공유를 위한 산업용 로봇 소프트웨어 설계 기법 [단편논문] 박진용, 유철중(전북대) 스마트 환경에서 디지털 사이니지와 이기종 디바이스간의 상호운용성을 지원하기 위한 메시지 플로팅 시스템 차민재, 이동우, 남태우, 염근혁(부산대) Automotive 소프트웨어 Security 향상을 위한 Coding Guideline 에 대한 연구 우중기, 정지현, 민상윤(KAIST)	소프트웨어 품질관리를 위한 이종 코드 변환 프레임워크 연구 [단편논문] 손현승, 김영철(홍익대) 역공학을 이용한 오픈소스의 소스코드 요구사항 도출 및 Best Practices 와의 비교 분석을 통한 소스코드의 품질평가 [단편논문] 위대한, 이석원(아주대) 소프트웨어 프로세스 인증제도의 국방 적용방향 [단편논문] 김영봉 유천수(한국국방연구원) 원자력 계측제어 소프트웨어의 안전성 분석을 위한 Safety Case 의 Arguments 개발 절차 [우수단편논문] 이동아, 유준범(건국대학교), 이장수 (한국원자력연구원)
12:10-12:30	**폐회식** 장소: 그랜드홀 2			사회: 이병정 학술위원장(서울시립대)
13:00-18:00		**SW 상시모니터링기술연구단** 장소: 세미나실 1 (13:00-18:00)		

* 위 일정은 사정에 따라 변경될 수가 있습니다.

소프트웨어 품질관리를 위한 이종 코드 변환 프레임워크 연구

손현승*, 김영철*

* 홍익대학교 소프트웨어공학연구실
세종특별자치시 조치원읍 세종로 2639
{son, bob}@selab.hongik.ac.kr

요약: MDA(Model Driven Architecture)는 플랫폼 독립적인 모델을 종속적인 모델로 변환해 코드를 생성하는 순공학 기법이고 ADM(Architecture Driven Modernization)은 코드로부터 모델로 만드는 역공학 기법이다. MDA 와 ADM 의 병합은 이종 플랫폼에서의 모델과 코드 동기화 가능해 소프트웨어 변경에 의한 모델과 코드 불일치 문제를 해결할 수 있다. 그러나 현재 MDA 와 ADM 은 호환되지 않아 모델변환에 많은 규칙과 절차가 요구된다. 본 논문에서는 이종 플랫폼에서 양방향으로 모델과 코드를 변환 가능한 이종 코드 변환 프레임워크를 제안한다. 제안한 프레임워크는 양방향 변형이 가능하도록 MDA 와 ADM 병합한 개발 프로세스, 모델 변환 언어, 엔진으로 구성되어 있다. 이 프레임워크는 모델에서 코드생성까지 서로 다른 언어와 방법으로 적용된 기존 모델변환의 개선으로 한 번의 모델 변환으로 모델에서 코드, 코드에서 모델로 변환 가능하다. 또한 이종 모델과 코드를 양방향으로 변환이 가능하다.

핵심어: MDA(Model Driven Architecture), 메타모델, ADM(Architecture Driven Modernization), 모델변환, 역공학

1. 서 론

소프트웨어의 중요성이 높아지면서, 소프트웨어는 광범위하게 사용되었고 크기가 켜졌고 복잡해졌다. 이러한 상황에서 소프트웨어의 개발비 절감과 신뢰성 확보 등의 품질관리가 중요한 현안으로 대두되고 있다.

고품질의 소프트웨어를 만들기 위해서는 SP(Software Process) [1], CMMi(Capability Maturity Model Integration) [2], TMMi(Test Maturity Model integration) [3] 등과 같은 인증 프로세스를 도입하여 개발과정의 성숙도를 높이거나 테스팅이나 GS(Good Software) [4] 같은 제품의 품질을 높이는 방법이 있다. 그러나 인증 프로세스는 평가 기준, 절차, 방법, 도구 활용, 문서 등과 같은 다양한 것들이 필요하여 도입 비용이 높다 [5]. 또한 테스팅은 전문 인력과 기술이 추가적으로 필요하다.

이러한 소프트웨어 개발 실정을 볼 때, 고품질 소프트웨어를 위한 다른 방안으로 기존 소프트웨어 자산 축적과 재사용 활용한 기술도 있다 [6]. 소프트웨어는 본질적인 면에서 축적된 지식과 경험을 반영한 상품으로 특이한 소유, 거래의 개념을 갖고 이를 재사용과 공유로 사회적 가치 증대가 가능하다.

이러한 소프트웨어의 특성 때문에 많은 회사들은 기존 소프트웨어로부터 자산을 구축하거나 재사용할 수 있는 플랫폼을 선호한다. 이렇다 보니 소프트웨어 기술이 발전할 수록 많은 수의 플랫폼이 생겨나고 있다. 이런 다양한 소프트웨어 플랫폼 등장으로 다중 플랫폼에서의 소프트웨어의 품질을 높일 수 있는 방법도 필요해졌다.

MDA(Model Driven Architecture) [7-8]는 닷넷(.Net), 코바(CORBA), 자바(Java)와 같은 플랫폼 기반 소프트웨어의 플랫폼간 종속 문제를 모델을 활용해 없애고 모델을 코드로 생성해 이종 플랫폼의 코드까지 생성할 수 있다. MDA 는 플랫폼의 독립과 종속 모델로 분리하고 두 모델간의 차이를 모델 변환 언어와 엔진을 사용해 자동화 한다. 두 모델간의 추상화 수준 차이로 생기는 정보는 프로파일이나 모델 변환 규칙을 통해서 극복한다.

ADM(Architecture Driven Modernization) [9]은 역공학 기법으로 코드로부터 모델을 모델변환으로 만드는 자동화 기법이다. 코드를 모델로 만들기 위해서 텍스트로 된 코드를 파싱한 ASTM(Abstract Syntax Tree Metamodel) [10]와 코드와 모델의 정보를 모두 저장하는 KDM (Knowledge Discovery Metamodel) [11]을 사용한다.

* 본 논문은 "손현승, MDA/ADM 통합 모델 기반의 이종 코드 변환 프레임워크, 홍익대학교 대학원, 2015" 박사학위 논문을 재편집하였음.

† 이 논문은 2015 년 교육부와 한국연구재단의 지역혁신창의인력양성사업(NRF-2015H1C1A1035548)과 2015 년도 정부(교육부)의 재원으로 한국연구재단의 지원을 받아 수행된 기초연구사업임(NRF-2013R1A1A2011601).

MDA 와 ADM 의 병합은 모델과 코드의 동기화가 가능하여 소프트웨어 변경에 따른 모델과 코드의 불일치 문제를 해결할 수 있다. 그러나 현재 MDA 와 ADM 기법의 수행은 두 기법 사이에 호환되지 않아 모델 변환시 많은 규칙과 절차가 요구된다.

본 논문에서는 이종 플랫폼 상에서 양방향으로 모델과 코드 변환을 위한 이종 코드 변환 프레임워크를 제안한다. 제안한 프레임워크는 양방향으로 모델과 코드를 변환하기 위해서 개발 프로세스, 모델 변환 언어, 엔진으로 구성되어 있다. 개발 프로세스에서는 MDA 와 ADM 를 병합하기 위해서 두 프로세스는 같은 모델을 사용하도록 하고 양방향 모델 변환 언어를 통해서 양방향 변형을 가능하게 한다. 또한 코드와 모델은 ASTM 을 공유하여 별도의 언어와 도구 없이 양방향으로 모델과 코드를 변환 할 수 있도록 하였다. 이를 위해 양방향 모델 변환 언어와 엔진은 여러 종류의 모델 변환 방법(Model to Model, Model to Text, Text to Model)을 하나의 언어로 처리할 수 있다.

제안한 프레임워크는 기존 소프트웨어 코드를 모델을 경유하여 이종 코드로 변환하거나, 코드를 모델로, 모델을 코드로 생성 가능하다. 특히, MDA 와 ADM 의 접목을 통해서 순공학 뿐만 아니라 역공학 기법 포함으로 코드를 이용한 개발에서 그래프를 통한 시각화나 문서화가 용이하여 소프트웨어 품질관리에 도움을 준다.

2. 이종 코드 변환 프레임워크

이종 코드 변환 프레임워크는 이종 플랫폼 개발을 위해 필요한 모델과 코드를 양방향으로 모델 변환할 수 있는 프로세스, 방법, 도구를 포함하는 프레임워크이다. 이 코드 변환 프로세스는 그림 1 과 같이 MDA 와 ADM 을 병합한 프로세스로 플랫폼 독립 모델로부터 양방향 모델 변환 언어로 종속 모델로 만들고 이를 ASTM(Abstract Syntax Tree Metamodel)으로 변환하여 코드를 생성하는 순방향 프로세스와 생성된 코드를 역으로 ASTM 으로 만들고 ASTM 을 종속 모델로, 종속 모델을 독립 모델로 변환하는 역방향 프로세스로 구성되어 있다. 그러므로 이 프로세스는 순방향과 역방향에 대한 변환을 모두 수행할 수 있는 순환구조이다.

MDA 과 ADM 을 병합만 한 경우 MDA 와 ADM 의 모든 과정을 수행해야 하므로 변환 절차가 복잡해지고 비효율적인 문제가 있다. 이러한 문제를 해결하기 위해서 제안한 이종 코드 변환 프로세스는 기존의 MDA 와 ADM 에서 꼭 필요한 단계만 수행하도록 최적화한다. 모델에서 모델의 변환은 양방향의 모델 변환 언어를 개발하여 변환의 수행을 간단하게 하였고 코드생성과 파싱은 ASTM 을 사용하여 표준화된 하나의 AST(Abstract Syntax Tree)에서 처리 할 수 있도록 하였다.

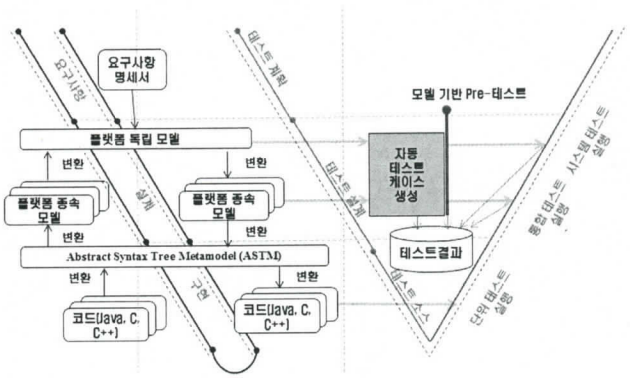

그림 1 이종 플랫폼을 위한 코드 변환 프로세스

제안한 프로세스의 실현을 위한 이종 코드 변환 구조는 그림 2 와 같다. 이 코드 변환 구조는 모델에서 모델로 변환하기 위한 변환 언어와 엔진과 ASTM 을 기반으로 코드 생성 및 파싱할 수 있는 파서로 구성한다. 모델 변환 언어는 모델에서 모델로 변환을 수행하기 위해서 변환되는 규칙을 메타모델을 사용하여 작성하고 양방향 모델 변환을 보다 효과적으로 수행하기 위해서 제안한 모델 변환 언어는 트리를 기반으로 한다. 파서는 ASTM 을 코드로 코드를 ASTM 으로 변환해 주는 역할을 수행한다. 이러한 두 가지 부분이 병합되어 모델에서 코드, 코드에서 모델로 자유롭게 자동변환 할 수 있다.

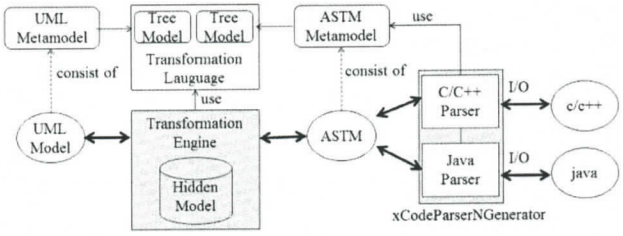

그림 2 이종 코드 변환의 구조

양방향 모델 변환에서는 모델간의 추상화 수준의 차이가 있거나 모델과 모델이 1:1 로 완전 매치되지 않는 경우 정보가 손실될 수 있다. 이 문제를 해결하기 위해서는 모델 변환 수행시 모델 변환 규칙이 1:1 로 대응되지 않을 때, 두 모델 사이의 관계를 1:1 대응 관계가 되도록 해줌으로 정보 손실의 문제점을 해결 할 수 있다.

3. 결 론

단일 플랫폼 기반 개발은 소프트웨어의 재사용으로 빠르게 개발이 가능하지만 이종 시스템 개발에

적합하지 않다. 또한 고품질 소프트웨어를 위해서는 코드의 분석을 통해 모델로 변환하거나 결함을 찾을 수 있는 방법이 필요한데, 기존 방법은 한 번에 수행할 수 없었다. 본 논문은 이러한 이종 시스템의 개발 환경과 소프트웨어 품질의 문제를 해결하고자 이종 코드 변환 프레임워크를 제안하였다.

제안한 이종 코드 변환 프레임워크는 양방향 모델과 코드를 변환 가능하여 모델에서 코드, 코드에서 모델의 과정을 일괄적으로 통합한 자동화 방법이다. 이를 위해서 MDA 와 ADM 을 병합한 프로세스를 제안하고, 또한 모델에서 모델 변환을 위한 양방향 모델과 코드 변환 언어와 엔진을 개발했다.

이종 코드 변환 프레임워크는 이종의 플랫폼을 사용하는 모바일, 스마트폰 환경, 임베디드 소프트웨어 등에 활용과 코드 가시화가 가능하여 소프트웨어 품질관리에 도움이 될 것으로 기대한다.

향후 연구로 양방향 모델 변환의 효율성을 높이기 위한 알고리즘과 언어 체계 개선 연구이다. 또한 제안한 프레임워크를 테스트 프로세스를 접목한다면 테스트 프로세스에 설계 모델을 그대로 사용하면 자동 테스트에 적용하여 더 높은 품질향상이 이루어질 수 있다.

참고문헌

[1] 임수빈, 이승주, 남성욱, 소프트웨어프로세스 품질인증 적용 가이드북 2 등급, 긍정 미디어, 2011.

[2] M.B. Chrissis, M. Konrad, S. Shrum, CMMI Guidlines for Process Integration and Product Improvement, Addison-Wesley, 2003.

[3] E. Van Veenendaal, J.J. Cannegieter, Test Maturity Model integration (TMMi), TMMi Foundation, 2010.

[4] 김현정, 장형진, 김장경, 신석규, "TTA 소프트웨어 시험 · 인증 서비스 소개", 한국통신학회지(정보와 통신), Vol. 31, No. 7, pp. 24-31, 2014.

[5] 양해술, 배두환, "소프트웨어 품질표준화와 시험 · 인증기술의 동향", 정보과학회지, Vol. 23, No. 3, pp. 45-55, 2005.

[6] I. Jacobson, M. Griss, P. Jonsson, Software reuse: architecture, process and organization for business success, ACM Press/Addison-Wesley, 1997.

[7] OMG, "Technical Guide to Model Driven Architecture: The MDA Guide v1.0.1", Document Number: omg/2003-06-01

[8] S.J. Mellor, A.N. Clark, T. Futagami, "Model-Driven Development", IEEE Software, no. 5, pp. 14-18, 2003.

[9] V. Khusidman, "Architecture-Driven Modernization and Transformation", ADM Transformation White Paper, pp. 1-5, 2008.

[10] OMG, "Architecture-driven Modernization: Abstract Syntax Tree Metamodel (ASTM) Version 1.0", Document Number: formal/2011-01-05.

[11] OMG, "Architecture-Driven Modernization: Knowledge Discovery Meta-Model (KDM) Version 1.3", Document Number: formal/2011-08-04.