프트웨어공학
사 이 어 티

18권 제1호
ol. 8 No. 1

KIISE · KIPS
Joint Workshop on Software Engineering Technology 2010
(KSEJW-2010)

2010
한국 소프트웨어공학기술 합동 워크샵 논문집

- 일시 : 2010년 8월 19일(목) ~ 20일(금)
- 장소 : 서울 상암동 누리꿈 스퀘어

주최 : 한국정보과학회, 한국정보처리학회
주관 : 한국정보과학회 소프트웨어공학 소사이어티
한국정보처리학회 소프트웨어공학 연구회
후원 : 포항공과대학교 융합소프트웨어개발 연구센터(COSDEC)
고려대학교 고신뢰 융합소프트웨어공학센터(CEEDS)
단국대학교 금융IT를 위한 소프트웨어공학연구센터(SERC-FIT)
숭실대학교 모바일 서비스 소프트웨어공학센터(MSSEC)
서강대학교 소프트웨어 요구 및 검증공학 센터(ReVeT)
KAIST 소프트웨어프로세스 개선센터(SPIC)
정보통신산업진흥원 소프트웨어공학센터, (주)다한테크

목 차

특별 강연

초청강연

패널 토의

Software Architecture

모델 변환을 이용한 안드로이드 어플리케이션 개발

손현승*, 김우열*, 장우성*, 김영철*

홍익대학교 일반대학원 소프트웨어공학 연구실*
{ son, john, bob, jang}@selab.hongik.ac.kr*

요약: 최근 스마트폰 경쟁이 치열해지면 다양한 플랫폼들이 출시되고 있다. 기존의 스마트폰용 어플리케이션 개발은 플랫폼에 종속적으로 만들어져 이종 시스템 개발 시 플랫폼 별로 각 각 개발해야 하는 문제가 있다. 애플은 코코아, 구글은 안드로이드, 마이크로소프트는 윈도우 모바일등 각 벤더마다 고유의 플랫폼으로만 개발한다. 본 논문의 목적은 이종의 스마트폰 어플리케이션 개발을 위해서 모델변환기법을 적용하는 것이다. 이 방법은 플랫폼의 독립적인 것과 종속적인 모델로 분리하고 모델 변환에 대한 규칙을 통해 독립 모델에서 종속모델 변환을 자동화하는 기법이다. 모델 변환 기법은 기본적으로 모델, 메타모델, 모델변환언어가 기본적으로 필요하다. 본 논문에서 모델은 UML, 메타모델은 UML 메타모델, 모델변환언어는 ATL 을 사용하였다.

핵심어: 모델 변환, 안드로이드, 스마트폰, ATL, 메타모델

1. 서론

전 세계적으로 스마트폰의 판매량은 계속 증가되고 있는 추세이다. 최근(2009 년 후반기) 국내에 아이폰이 출시 되면서 국내에도 스마트폰에 대한 관심과 시장이 활성화 되었다. 기존의 피처폰과(Feature Phone)는 다르게 스마트폰(Smart Phone)은 다양한 컨텐츠를 활용할 수 있는 점이 매력적이다. 메일을 주고받을 수 있으며 게임, 워드프로세스, 네비게이션 등 다양한 어플리케이션을 활용할 수 있다. 결국 스마트폰 시장에서 다양한 컨텐츠를 얼마나 빠르고 안정적으로 개발해 줄 수 있는 환경이 매우 중요해 졌다.

소프트웨어 개발을 빠르게 하기 위해서는 기존에 만들었던 자원들을 최대한 활용 가능한 재사용 체계가 갖추어져 있어야 한다. 그러나 모바일 임베디드 소프트웨어는 시스템에 종속적인 특성을 지니고 소스 코드를 중심으로 개발이 진행되기 때문에 소프트웨어의 재사용이 어렵다[1]. 더욱이 각 휴대폰 제조 및 공급 회사마다 다양한 플랫폼을 제공하기 때문에 이를 해결할 수 있는 방법이 필요하다[2].

본 논문에서는 이종의 스마트폰 개발환경을 위해 안드로이드 응용 프로그램 개발에 모델 변환 기법을 적용한다. 안드로이드 플랫폼에 적용하기 위해서 안드로이드 어플리케이션의 구조를 분석하고 독립적인 특성과 종속적인 특성들로 분류화한다. 안드로이드 플랫폼으로 모델 변환을 위해 필요한 메타모델을 추출해내고 마지막으로 모델 변환을 위한 규칙들을 제안한다. 모델 변환의 규칙은 모델 변환 언어인 ATL(ATLAS Transformation Language)으로 기술한다. 적용사례로 안드로이드 어플리케이션의 독립모델을 만들고 이것을 이클립스 환경에서 모델변환을 수행한다. 생성된 안드로이드 플랫폼의 종속 모델을 모델 변환 규칙에 의해 정확하게 생성되었는지 확인한다.

적용사례를 통해서 모델변환 기법을 스마트폰 환경에 적용할 수 있음을 확인할 수 있었다. 또한 안드로이드 플랫폼 뿐만 아니라 아이폰, 윈도우 모바일 등 다양한 플랫폼에 적용 가능성을 보았다. 향후 연구로 모델 변환 방법을 아이폰, 윈도우 모바일용에 적용할 것이다.

본 논문의 구성은 다음과 같다. 2 장에서는 관련된 연구로서 기존의 MDD 의 기본 개념과 모델변환 방법 및 ATL 대해서 기술한다. 3 장에서는 모델 변환을 수행하기 위해서 안드로이드 어플리케이션 분석과 모델 변환을 수행하기 위한 메타모델 정의와 규칙 생성에 대해서 언급한다. 4 장에서는 적용사례로 안드로이드 어플리케이션을 개발을 위해서 타겟 독립 모델을 만들고 이를 모델변환 수행과정에 대해서 다룬다. 마지막으로 5 장에서는 결론 및 향후 연구를 언급한다.

2. 관련연구

2.1 모델 변환 방법

모델 변환은 입력되는 소스 모델을 대상 모델로

† 본 연구는 지식경제부 및 정보통신산업진흥원의 대학 IT 연구센터 지원사업(NIPA-2010-(C1090-0903-0004))과 교육과학기술부와 한국연구재단의 지역혁신인력양성사업으로 수행된 연구결과임.

변환하는 방법이다. 모델 변환의 종류에는 모델에서 모델, 모델에서 코드, 코드에서 모델 등이 있다. 또한 모델 변환에서 사용되는 모델은 UML 뿐만 아니라 컨트롤 플로우 다이어그램, 데이터 플로우 다이어그램 등 다양한 모델로도 변환이 가능하다. UML 은 UML 메타모델이 지원이 되기 때문에 메타모델을 만들지 않아도 되지만 메타모델이 없는 모델일 경우 MOF(Meta Object Facility)[3]를 이용하여 메타모델로 설계가 요구된다.

모델 변환을 수행하기 위해서는 소스 모델, 소스 메타모델, 타겟 메타모델, 변형 정의가 필요하다. 모델 변환 엔진은 이러한 요소들을 사용해서 타겟 모델로 변환한다. 모델 변환에서 중요한 요소로는 변환 룰을 정의하는 언어에 있다. UMT(UML Model Transformation)[4], MTL(Model Transformation Language)[5], QVT(Query / View / Transformation)[6], ATL(ATLAS Transformation Language)[7,8] 등이 있다. 기존 기법들 중에 ATL 은 무엇보다도 변환규칙 정의가 보다 명확하고 이해하기 쉬우며 복잡한 변환규칙에 대한 정의가 가능하다. 또한 변환모델에 대해서 재사용 및 합성을 지원하는 장점을 가지고 있다. 본 논문에서는 여러 가지 변환 언어들 중에서 ATL 의 이러한 장점 때문에 모델 변환 언어로 선택하였다.

2.2 ATL(ATLAS Transformation Language)

ATL 은 ATLAS INRIA & LINA 연구 단체가 개발한 모델 변환 언어이다[9]. 이것은 메타모델 및 모델로 지정된 모델 변환 언어이다. ATL 은 개발자들에게 원본 모델에서 다수 대상 모델로 일으키는 방법을 지정하는 방법 제공한다.

ATL 언어는 서술적이고 일종의 프로그램 언어의 일부이다. 변환 언어를 표현하는데 선호하는 방법은 서술적인 것이다. 그것은 단순히 원본과 대상 모델 성분 사이에 관계를 표현하는 것을 가능하게 한다. 또한, ATL 은 간단하게 서술적으로 표현될 수 없는 복잡한 명세를 위하여 OCL(Object Constraint Language)[10]을 사용한다.

ATL 이 정의하는 규칙은 원본 모델의 성분이 어떻게 일치하고 대상 모델의 성분을 창조하고 초기화하기 위한 방법으로 구성된다. 기본적인 모델 변환 외에, ATL 는 모델에 요구를 지정하는 것을 가능하게 하는 질의 문을 정의한다. 또한 ATL 은 ATL 라이브러리의 정의를 통해 모듈화를 허용한다.

ATL 통합 개발 환경(IDE)은 다수 표준 ATL 변환의 디자인을 쉽게 하기 위하여 개발 도구 를 제공한다. ATL 개발 환경은 또한 모델과 메타모델의 취급에 다수 추가 기능을 제안한다. 이 특징은 메타모델의 명세에 전념한 간단한 본문 표기법, 또한 일반적인 본문 통어론과 그들의 대응 모형 대표 사이 다수 기준 교량을 포함한다.

3. 모델 변환을 이용한 안드로이드 어플리케이션 개발

모델 변환을 수행하기 위해서 기존의 안드로이드 플랫폼을 분석하고 이를 통해서 타겟 독립 모델과 모델변환 규칙을 작성한다.

3.1 플랫폼 분석

안드로이드 아키텍처는 Applications, Application Framework, Libraries, Android Runtime, Linux Kernel 레이어로 분리되어있다. 이들 중에서 본 논문에 분석 대상은 Application Framework 이다. 기본적으로 응용프로그램을 개발하기 위해서는 Application Framework 만으로 충분하기 때문이다. 또한 안드로이드는 잘 정의된 Application Framework 를 제공하기 때문에 하위 레벨에 있는 Libraries 를 직접 제어할 필요는 없다.

기본적인 Application Framework 의 요소를 살펴보면 Activities, Service, Broadcast receivers, Content providers 이 있다. Activities 는 사용자 인터페이스를 보여주는 부분으로 화면의 레이아웃을 결정하고 UI 컴포넌트를 배치시키고 뷰를 붙이는 등의 작업을 할 수 있다. 기본적인 프로그램이라면 Activities 만 이용하여도 충분하다. Service 는 화면에 보여주는 UI 는 아니고 백그라운드로 실행되는 일종에 데몬 같은 것이다. 안드로이드에서는 프로그램이 반복적으로 수행되는 것들을 Service 로 정의할 수 있다. Broadcast receivers 는 다른 응용프로그램들과의 통신을 제공해준다. 예를 들어 어떤 데이터를 다운로드하고 그 장치가 사용 가능하게 되면 다른 어플리케이션이 이를 알 수 있도록 메시지를 전달 해준다.

Content providers 는 다른 응용프로그램이 제공하는 기능을 사용할 수 있도록 한다. 예를 들어 파일처리를 할 때, SQLite 를 사용하여 데이터베이스를 사용할 수 있는 것을 말한다.

3.2 모델 변환 규칙 작성

3.2.1 규칙 1 : 기본 구조 생성

기본 구조의 생성은 안드로이드 어플리케이션을 구동하기에 필요한 기본적인 요소를 말한다. 이와 더불어 UML 클래스 다이어그램에 기본적으로 필요한 데이터 타입 등을 생성한다. 기본 구조를 생성하기 위한 순서는 먼저 dataType 클래스를 복사한다. 안드로이드 기본적으로 생성되는 Activity 클래스가 컨트롤을 수행하는데 이것은 컨트롤 클래스 생성시에 만들어지기 때문에 기본 구조 생성과정에서 필요하지 않다. 그림 1 에 기본 구조 생성 방법을 자세히 설명하였다. 규칙 1 의 ATL 은 수행되는 조건은 없고 한번만 실행된다.

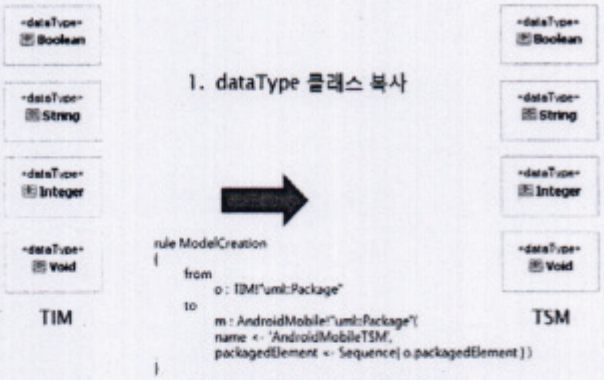

그림 1. 기본 구조 생성

3.2.2 규칙 2 : view 생성

안드로이드에서 사용되는 뷰에는 View 클래스가 있다. 그러므로 화면에 그림을 그리기 위해서 View 클래스가 요구된다. view 생성을 위한 순서는 먼저 View 클래스를 생성하고 그 다음 View 클래스와 LinkView 클래스 간에 일반화관계를 생성한다. 안드로이드는 View 를 서브 클래싱하는 클래스는 생성자를 반듯이 포함해야 한다. 그러므로 LinkView 클래스에 생성자와 그에 해당하는 파라메터를 추가해준다. 규칙 2 를 ATL 로 표현하면 그림 2 와 같다. 수행되는 조건은 스테레오 타입 <<view>>가 있는 클래스만 변환한다.

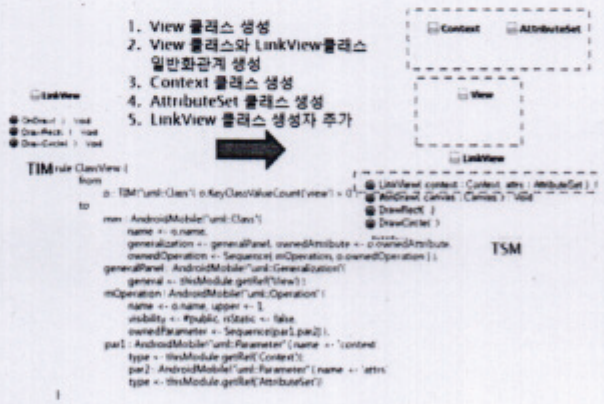

그림 2. view 생성

플랫폼 분석과정을 통해서 추출한 독립/종속 모델을 통해서 모델 변환 규칙을 작성할 수 있다. 총 5 가지 규칙을 통해서 생성할 수 있다. 사용한 스테레오 타입은 클래스에서는 <<view>>, <<controller>> 메서드에는<<view_ondDraw>>, <<button_onClick>> 이다. 모델 변환을 규칙은 기본 구조 생성, view 생성, view_onDraw 생성, controller 생성, button_onClick 생성 총 5 가지 형태로 view_onDraw 생성, controller 생성, button_onClick 생성 3 가지는 view 생성과 같은 방법으로 규칙을 생성한다.

4. 적용사례

모델 변환을 수행하기 위한 적용사례는 버튼을 누르면 웹브라우져를 열고 등록되어 있는 사이트로 이동하는 간단한 주소록 관리 프로그램이다. 이러한 프로그램을 만들기 위해서는 그림 3 과 같이 윈도우 화면에 버튼이 필요하다.

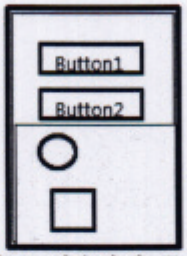

그림 3. 적용사례 구조도

4.1 타겟 독립 모델

타겟 독립 모델을 정의 하면 그림 4 와 같이 할 수 있다. LinkView 는 화면에 그림을 표현하는 클래스이다. LinkViewController 는 화면에서 어떠한 이벤트가 발생되었을 때 처리해주는 핸들러를 관리해주는 클래스 이다. 본 논문의 사례에서는 버튼 2 개가 사용되었으므로 Btn1_onClick()과 Btn2_onClick() 메서드가 컨트롤 클래스에 추가하였다.

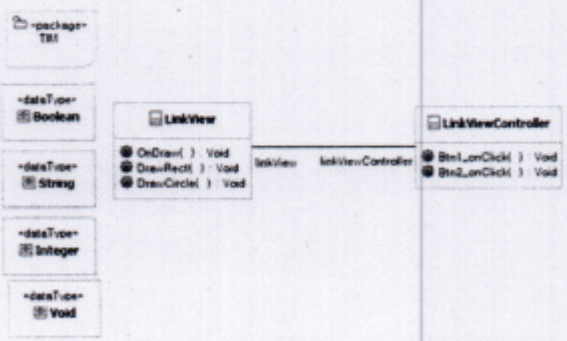

그림 4. 타겟 독립 모델의 클래스 다이어그램

4.2 타겟 종속 모델

타겟 독립 모델을 3 장에서 제안한 규칙 5 개를 수행하면 그림 5 와 같은 클래스 다이어그램을 생성할 수 있다. 그림의 숫자는 적용된 규칙 번호 이다.

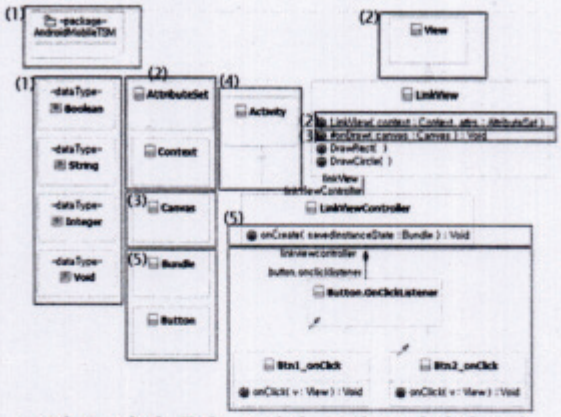

그림 5. 타겟 종속 모델의 클래스 다이어그램

4.3 구현

타겟 종속 모델을 코드 생성한 후 버튼을 눌렀을 때 브라우저를 여는 코드를 작성하여 주면 프로그램을 완성할 수 있다. 이 프로그램을 컴파일 하여 실행하면 그림 6 과 같이 수행되는 결과를 확인 할 수 있다.

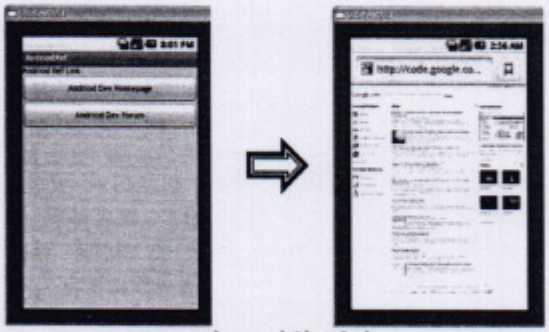

그림 6. 시현 결과

5. 결론

본 논문에서는 이종의 스마트폰 개발을 위해서 안드로이드 플랫폼 개발에 모델 변환 기법을 적용하였다. 모델 변환 기법을 적용하기 위해서 안드로이드 플랫폼의 어플리케이션의 구조를 분석하였다. 분석한 결과를 통해서 독립적인 특성과 종속적인 특성들로 구분하고 독립/종속 모델 API 매트릭스를 만들고 이를 기반으로 모델 변환 규칙을 만들었다. 만들어진 규칙은 5 개의 규칙이며 크게 구분하면 기본구조, 클래스 단위, 메서드 단위 3 가지로 분류화 할 수 있다. 또한 모델 변환 규칙을 ATL 을 사용하여 기술하였고 이것을 이클립스 환경에서 모델 변환을 수행하였다. 그 결과 모델 변환이 변환 규칙대로 클래스 다이어그램을 생성한 것을 확인 할 수 있었다.

향후연구로 아이폰, 윈도우 모바일 플랫폼을 분석하고 본 논문에 적용한 방법을 각 플랫폼에 맞게 확장할 것이다.

참고문헌

[1] Axel Jantsch, (2004), Modeling Embedded System and SOCs, Mogan Kaufmann

[2] Woo Yeol Kim, Hyun Seung Son, R. Young Chul Kim, C. R. Carlson,"MDD based CASE Tool for Modeling Heterogeneous Multi-Jointed Robots", Proceedings of the 2009 WRI World Congress on Computer Science and Information Engineering, Vol. 7, 775-779, 2009.04.01.

[3] OMG, Meta Object Facility Specification, In OMG Unified Modeling Language Specification, Version 2.0, January 2006.

[4] Roy Grønmo and Jon Oldevik, "An Empirical Study of the UML Model Transformation Tool(UMT)", In The First International Conference on Interoperability of Enterprise Software and Applications (INTEROP-ESA), Geneva, Switzerland, February 2005.

[5] D. Vojtisek and J.-M. Je´ze´quel, "MTL and Umlaut NG:Engine and Framework for Model Transformation",http://www.ercim.org/publication/Ercim_News/enw58/vojtisek.html

[6] OMG, Documents associated with Meta Object Facility (MOF) 2.0 Query/View/Transformation, Version 1.0, April 2008.

[7] J. Be´zivin, G. Dupe´, F. Jouault, G. Pitette, and J. E. Rougui, "First Experiments with the ATL Model Transformation Language: Transforming XSLT into XQuery", Proceedings of the Workshop on Generative Techniques in the Context of Model Driven Architecture, Anaheim,CA, 2003.

[8] F. Jouault and I. Kurtev, "Transforming Models with ATL", Proceedings of Model Transformations in Practice Workshop (MTIP), MoDELS Conference, Montego Bay, Jamaica, 2005.

[9] Wikipedia, "ATLAS Transformation Language" http://en.wikipedia.org/wiki/ATLAS_Transformation_Language

[10] OMG, Object Constraint Language Specification, In OMG Unified Modeling Language Specification, Version 2.0, May 2006.