제26회 한국정보처리학회 추계학술발표대회 논문집 제13권 제2호 (2006. 11)

임베디드 소프트웨어 개발을 위한 하이브리드 방법론 지원 도구

김종필*, 오기영*, 홍장의**
충북대학교 전자계산학과
*{kimjp, ohgy}@selab.chungbuk.ac.kr, **jehong@chungbuk.ac.kr

A Tool for Hybrid Modeling Approach of Embedded Software Development

Jong-Phil Kim, Gy-Young Oh, Jang-Eui Hong
Dept of Computer Science, Chungbuk National University

요 약

임베디드 소프트웨어를 개발하기 위한 많은 노력이 요구되고 있다. 최근 UML 2.0이 임베디드 소프트웨어를 모델링하기 위한 다양한 특성을 포함하면서, UML을 이용하여 임베디드 소프트웨어를 모델링하기 위한 다양한 시도가 이루어지고 있다. 그러나 기존에 임베디드 소프트웨어는 대체적으로 구조적 설계 방법론에 근간하여 개발되어 왔다. 본 연구에서는 다양한 모델링 방법론을 지원하기 위한 임베디드 소프트웨어 개발용 지원 도구에 대하여 설명한다. 본 연구에서 개발한 도구는 다양한 모델링 방법의 지원뿐 아니라, 코드의 생성에서도 다양성을 지원하도록 하였다.

1. 서론

임베디드 소프트웨어는 최근 10여년동안 매우 많은 응용 영역에서 사용되어 왔다. 특히 응용 시스템의 규모가 대형화되고, 사용자 서비스가 다양화되면서 임베디드 소프트웨어에 대한 요구는 더욱 증대되고 있다. 따라서 임베디드 소프트웨어를 개발하기 위한 다양한 공학적 기술 또한 발전하고 있는 추세이다[10].

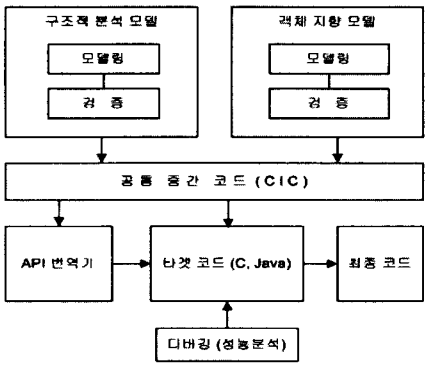
이러한 기술 중에서 임베디드 소프트웨어를 모델링하기 위한 방법론에 대한 연구들도 여러 가지 제시되었는데, CODART(Concurrent Design Approach for Real-Time Systems)[9]와 같은 구조적 방법론과 OCTOPUS[3]와 같은 객체지향 방법론이 대표적이라 할 수 있다. 그런데 최근 객체지향 개념을 지원하는 모델링 언어인 UML 2.0[14]이 발표되면서 임베디드 소프트웨어 모델링에 UML을 이용하는 시도가 증가하고 있다.

본 연구에서는 기존에 자료 흐름도(DFD), 상태차트(Statechart), 구조 차트(Structure Chart) 등을 사용하는 구조적 방법론과 클래스 다이어그램, 시퀀스 다이어그램 등을 사용하는 객체지향 방법론을 함께 사용하여 소프트웨어를 모델링할 수 있는 도구를 개발하는 데 그 목적이 있다. 특히 임베디드 소프트웨어 개발 프로젝트에서 서로 다른 팀이 각각 다른 모델링 표현법을 사용하거나 기존의 모델을 통합하여 새로운 소프트웨어를 개발하고자 하는 경우 매우 유용하게 지원할 수 있을 것이다. 또한 코드의 생성에 있어서도 C언어나 Java와 같은 언어를 생성하도록 하고 있다.

2. 개발 도구의 구성

개발하고자 하는 도구의 구성은 (그림 1)과 같다. (그림 1)에서 보는 바와 같이 개발도구는 임베디드 소프트웨어의 모델링 및 코드 생성을 지원하는 도구로써, HOPES(Hopes of Parallel Embedded Software)라고 불린다[2]. HOPES 도구의 기본 개념은 다양한 분석 및 설계 모델을 수용하기 위해 모델로부터 공통의 중간 코드인 CIC(Common Intermediate Code)를 생성하는 것이다. 생성된 중간 코드는 임베디드 소프트웨어의 아키텍처 측면과 실행 측면이 내

본 과제는 정보통신부 선도기반기술개발사업의 지원으로 수행되었습니다.



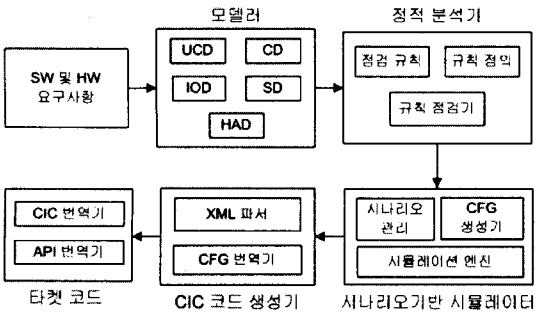
(그림 1) HOPES 도구의 구성 및 개념도

용을 모두 포함하도록 생성되며, 이로부터 타겟 환경에 적합한 소스 코드가 생성되도록 하고 있다. 구조적 모델이나 객체지향 모델은 CIC 코드의 생성에 앞서 각각 요구사항이 만족 여부 및 모델의 일관성을 검사하게 되며, 생성된 CIC 코드는 CIC Translator를 통해 C나 Java 언어로 번역된다. 이러한 번역과정에서 CIC 코드에 포함된 다양한 API들이 타겟 언어에 맞도록 확장되며, 가상 프로토타입 환경에서 디버깅과 성능 분석 과정을 거쳐 최종의 코드를 생성하게 된다.

3. ESUML 도구 구성 요소

HOPES 도구를 구성하는 여러 가지 컴포넌트 중에서 본 논문에서는 객체지향 기반 모델링을 지원하는 ESUML(Embedded Software development using UML 2.0) 도구에 대하여 중점적으로 설명한다.

ESUML 도구 부분에 대한 보다 상세화된 운영 개념은 (그림 2)와 같다.

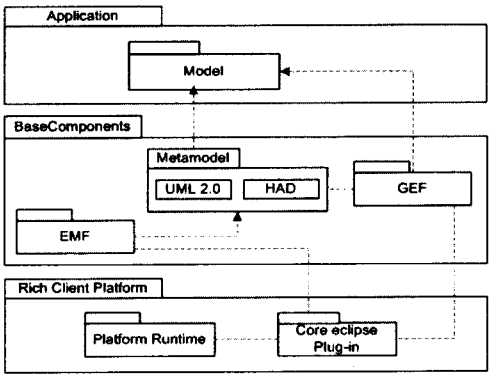


(그림 2) ESUML 도구 구성 및 개념도

임베디드 소프트웨어 개발의 상위 단에는 소프트웨어 및 하드웨어 요구사항을 기반으로 하는 모델링 부분이 위치하고, 모델의 일관성 및 동적 행위 분석 부분, 그리고 모델로부터 CIC 코드를 생성하는 부분과 하위 단의 타켓 코드 생성 부분이 존재한다. 각 구성요소에 대하여 설명하면 다음과 같다.

3.1 ESUML 모델러

ESUML 모델러는 UML 다이어그램의 편집기 이다. 특히 본 연구에서는 상호작용 기반의 임베디드 소프트웨어 모델링 방법을 채택하고 있으며, Use Case, Class, Interaction Overview, 그리고 Sequence 다이어그램을 사용한다[6]. 또한 타겟의 하드웨어 아키텍처를 모델링하기 위한 다이어그램이 생성된다. 이들은 UML 2.0 메타모델과 새롭게 정의된 하드웨어 아키텍처 다이어그램의 메타모델을 이용하여 모델을 생성하고 저장, 관리한다. (그림 3)은 ESUML 모델러의 아키텍처를 나타낸 것이다.



(그림 3) ESUML 모델러의 소프트웨어 아키텍처

3.2 ESUML 정적분석기

ESUML 정적분석기는 생성된 모델의 일관성을 분석하기 위한 기능을 제공한다. 일관성 분석은 특정 다이어그램(예를 들면, 시퀀스 다이어그램)내에서의 일관성, 서로 다른 다이어그램(예를 들면, 클래스 다이어그램과 시퀀스 다이어그램)간의 일관성에 대하여 분석한다. 일관성 분석을 위한 규칙들은 OCL 언어[13]를 이용하여 표현되며, 소프트웨어 개발자가 직접 분석을 위한 규칙을 정의할 수 있다. 다음은 OCL로 작성된 모바일 폰 메시징 서비스 시스템에 대한 일관성 분석 규칙의 예제이다.

3.3 ESUML 시뮬레이터

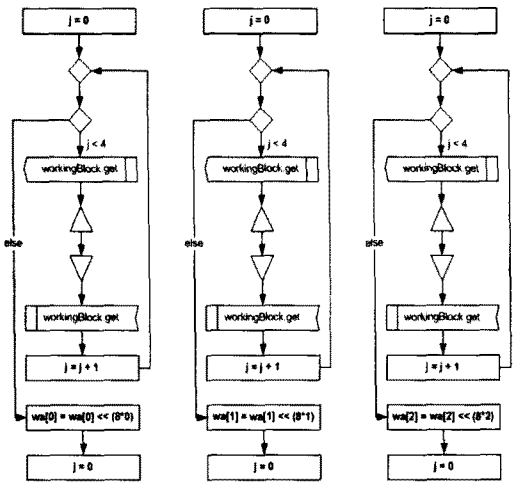
시뮬레이터는 생성된 모델이 동적으로 올바르게 수행되는지를 판단하기 위해 제공된다. 시뮬레이션

∀ e ∈ Element | *extended* (e, BinaryAssociation) :
if (e.connection.size ≠ 2) *then let* (Model.default = inconsistency)

∀ m ∈ Message, ∀ o∈ Operation, ∀ c ∈ Classifier |
(c = m.receiver) *and* (o = m.action) *and* (o.type = 'CallAction') :
if not (*isOperationOf* (o, c)) *then let* (Model.default = inconsistency)

을 수행하기 위하여 시나리오를 정의하게 되고, 정의된 시나리오에 따라 모델에 표현된 행위가 Interaction Overview 다이어그램을 따라 실행된다.

모델의 시뮬레이션을 위해서는 클래스 다이어그램에 나타난 액티브 오브젝트 - 클래스 다이어그램에서는 <<control>>, <<periodic>>, <<asynchronous>>의 스테레오타입 클래스들 -를 중심으로 CFG (Control Flow Graph)[7]를 생성하게 되며, 이 그래프를 따라가며, 행위 시뮬레이션이 진행된다. (그림 4)는 도구로부터 생성되는 AES 암호화 시스템[8]에 대한 CFG의 예를 보여준다.



(그림 5) CFG 생성 예제

3.4 CIC 코드 생성기

CIC 코드를 생성하기 위해서는 하드웨어 아키텍처 다이어그램의 메타 모델 정보와 UML 다이어그램 정보가 필요하다. 하드웨어 아키텍처에 대한 메타모델 정보로부터 CIC 코드내의 실행 구조에 대한 정보를 생성하고, UML 모델로부터 기능 실행을 위한 처리 로직이 생성된다.

CIC 코드는 기본적으로 두가지 형식으로 생성된다. C 언어와 유사한 구조를 갖는 CIC-C 타입 코드와 CIC-J 타입 코드이다. 전자는 타겟 코드를 C 언어로 생성하고자 하는 경우이며, 후자는 Java와 같은 객체지향 언어로 타겟 코드를 생성하기 위함이다. 이러한 타입의 코드가 갖는 공통점은 공유 메모리 접근이나, 프로세서 간 통신을 지원하기 위한 API를 갖는다는 점이다. 다음은 Microwave Oven Controller 시스템에 대한 CIC-C 타입의 코드 예제를 보여 준다.

```
typedef struct {
    BOOL isCooking;
    BOOL (*isCooking)(OvenCtrl*);  /* attributes and methods */
    :
} OvenCtrl;

OvenCtrl ovenCtrl = { FALSE, OvenCtrl_method_isCooking, ...};

void OvenCtrl_Init(void) {
   /* Task OvenCtrl Initialize Code */
}

void OvenCtrl_go(void) {
  Event event;
  while( TRUE ) {
    QUEUE_RECEIVE("channel1", (char *)&event, sizeof(event));
    switch( event.eventCode ) {
      case EVENT_CODE_cancelCook:
           ovenCtrl.isCooking = *(ovenCtrl.isOvenCtrl) (&ovenCtrl);
           if ( ovenCtrl.isCooking ) {
              event.eventCode = EVENT_CODE_beep;
              event.parameter = NULL;
              QUEUE_SEND("channel2", (char*)&event, sizeof(event));
              event.eventCode = EVENT_CODE_turnOff;
              event.parameter = NULL;
              QUEUE_SEND("channel3", (char*)&event, sizeof(event));
              event.eventCode = EVENT_CODE_turnOff;
              event.parameter = NULL;
              QUEUE_SEND("channel4", (char*)&event, sizeof(event));
              event.eventCode = EVENT_CODE_clearTime;
              event.parameter = NULL;
              QUEUE_SEND("channel5", (char*)&event, sizeof(event));
           } else {
              event.eventCode = EVENT_CODE_beep;
              event.parameter = NULL;
              QUEUE_SEND("channel2", (char*)&event, sizeof(event));
           }
        }
    }
}
```

3.5 타켓 코드 생성기

타겟 코드 생성기는 CIC 코드로부터 API Translator에 의해 통신 API 및 병렬 API 등이 타겟 코드로 변역되고, 생성된 코드가 탑재될 운영 환경, 즉 운영체제나 디바이스 인터페이스 등에 맞도록 번역된다.

4. 도구 개발 환경

ESUML 도구는 윈도우 환경의 이클립스(버전 3.2)[2, 5] 플랫폼에서 개발되었다. 도구 개발에 대한 세부 환경을 살펴보면 다음과 같다.

4.1. Rich Client Platform

Rich Client Platform(RCP)[5, 11]은 이클립스 3.0부터 도입된 기술로, 이클립스가 단순히 개발툴이 아닌 어플리케이션 플랫폼으로서 활용할 수 있도록 정의된 것이다. 이클립스는 모든 GUI 플랫폼에서 사용할 수 있도록 추상화된 GUI 플랫폼인 SWT[5]를 기반으로 하고 있다. SWT를 바탕으로 GUI 플랫폼의 독립성을 확보하고, 그 위에 수많은 플러그인이 기여된 결과가 이클립스이다. 이 중에서 독립적

인 어플리케이션(Stand-alone Application)에 필수적인 요소들을 뽑아 구성한 것이 RCP이다.

4.2 EMF, GEF Plug-In

EMF(Eclipse Modeling Framework)[4, 12]는 구조화된 모델을 기반으로 어플리케이션을 만들어 내기 위한 소스 생성 도구이며 자바 프레임워크이다. EMF는 소스 생성이외에 다른 어플리케이션과 상호 교환을 위해 XML[1] 문서로 모델을 저장하는 기능을 제공한다. 생성된 소스는 확장 가능한 자바 클래스들의 집합으로 생성된다. 이러한 EMF는 모델 기반을 통해 빠른 개발을 할 수 있도록 지원한다.

GEF(Graphical Editing Framework)[12]는 Graphical Object의 편집을 위한 프레임워크이다. 즉, 페인트 계열의 그래픽을 위한 것이 아니라 Drawing 계열의 그래픽을 위한 것이다. GEF는 Object를 그래피컬하게 표현하는 것과 마우스, 키보드 또는 워크벤치로부터 상호작용을 지원하기 위해 필요한 기본적인 컴포넌트를 제공함으로써 UML 다이어그램 에디터 같은 복잡한 어플리케이션을 보다 적은 노력으로 개발할 수 있도록 지원한다.

4.3 이클립스 UML2 및 HAD 메타모델

이클립스 UML2[5]는 이클립스 플랫폼을 위해 EMF기반으로 만들어진 UML 2.0 메타모델이다. 이클립스 UML2는 모델링 도구 개발을 지원하기 위한 구현된 메타모델을 제공하고 모델의 상호교환을 위한 공통 XMI 스키마를 제공한다.

HAD(Hardware Architecture Diagram) 메타모델은 하드웨어 아키텍처를 모델링하기위한 메타모델을 정의한 것이다. 하드웨어 아키텍처를 구성하는 요소들에 대한 메타정보들을 제공하고 임베디드 소프트웨어의 PSM(Platform Specific Mode)을 모델링할 수 있도록 지원한다.

5. 결론 및 향후 연구내용

임베디드 소프트웨어의 응용 분야가 확대되면서 소프트웨어 개발을 위한 모델링 기술이 발전하고 있다. 구조적 개발 방법과 객체지향 개발 방법은 임베디드 소프트웨어 개발의 가장 일반적인 방법론으로 사용되어 왔다. 본 연구에서는 구조적 방법론과 객체지향 개발 방법의 동시 지원이 가능한 모델링 프레임워크와 지원 도구에 대하여 소개하였다. 이중에서도 특히 UML2.0을 기반으로 하는 모델링 지원 도구, ESUML에 대하여 상세하게 소개하였다.

제안하는 지원도구는 기존에 구조적 방법론을 사용하던 임베디드 소프트웨어 개발 조직이나 UML을 이용하려는 조직에서 모두 유용하게 활용할 수 있는 지원도구가 될 것으로 보인다. 특히 CIC 라는 공통의 코드를 통해 다양한 모델을 수용할 수 있도록 함으로써, 하이브리드 개념의 모델링이 가능하다. 향후의 연구내용은 현재 개발 중인 도구를 완성하고 프로토타입 시스템 개발에 적용하는 것이다.

참고문헌

[1] 김윤명, "XML을 위한 Java Programming", 가남사, 2002

[2] 하순회, "MPSoC용 임베디드 소프트웨어 설계 및 검증을 위한 모델기반 프레임워크", 정보과학회지, 24권8호, 2006, pp.12-18

[3] Maher Awad, et al., "Object-Oriented Technology for real Time Systems: A Practical Approach Using OMT and Fusion", Prentice Hall

[4] Frank Budinsky, et al., "eclipse Modeling Framework", Addison Wesley, 2003

[5] Eclipse.org, Home, www.eclipse.org

[6] H.E. Eriksson, et al., "UML2 Toolkit", Wiley Publishing, 2004

[7] Douglas G. Fritz, Robert G. Sargent, "An overview of hierarchical control flow graph models", 27th conference on Winter simulation, 1995, pp.1347-1355

[8] FIPS-197, "Advanced Encryption Standard, NIST CSRC, 2001

[9] Hassan Gomaa, "Software Design Methods for Concurrent and Real-Time Systems", Addison Wesley Professional

[10] Edward A. Lee, "Embedded Software", Advance in Computer, Vol.56, 2002

[11] Jeff McAffer, Jean-Michel Lemieux, "eclipse Rich Client Platform", Addison Wesley, 2003

[12] Bill Moore, et al., "Eclipse Development using the Graphical Editing Framework and the Eclipse Modeling Framework", ibm.com/redbooks

[13] OMG.org, "UML2.0 OCL Specification", ptc / 03-10-14

[14] OMG.org, "Unified Modeling Language: Superstructure", version 2.0