

제25회 춘계학술발표논문집(상)

Proceedings of the 25th KIPS Spring Conference

제13권 제1호



Database · Artificial Intelligence · Multimedia · Software Engineering
Teleeducation · Information processing · Computer System
Image & Speech Processing · Information & Telecommunications

- 일시 : 2006년 5월 12일(금) ~ 13일(토)
- 장소 : 숙명여자대학교
- 주최 : 사단법인 한국정보처리학회



사단
법인

한국정보처리학회

Korea Information Processing Society

4. PS-Block 구조 기반의 반복횟수 분석 구조 설계(kips_0189)	김윤관*, 신 원, 김태완, 장천현(건국대학교) · 195
5. TMO기반 프로그램을 위한 소스코드 분석 및 분해(kips_0191)	이재석*, 신 원, 김태완, 장천현(건국대학교) · 199
6. 재사용 소프트웨어 컴포넌트의 Statemate 모듈화(kips_0219)	김창진*, 최진영(고려대학교) · 203
7. 일관성을 보장하는 UML 특성모델 편집기(kips_0227)	임용섭*, 김지홍(경원대학교) · 207
8. CMMI 기반 프로젝트 관리시스템(ML2-PMS) 구축(kips_0230)	박상필*, 정일재, 엄희균, 황선명(대전대학교) · 211
9. MDA 기반 모바일 컨버전스 임베디드 소프트웨어 개발 도구 및 방법론에 관한 연구(kips_0261)	박은주*, 김행곤(대구가톨릭대학교) · 215
10. 임베디드 소프트웨어 테스트 도구를 위한 아키텍처에 대한 연구(kips_0266)	장신재*, 김지영, 손이경, 김행곤(대구가톨릭대학교) · 219
11. 산업용 내장형 소프트웨어 품질평가 모델의 개발(kips_0289)	이하용*(서울벤처정보대학원대학교), 황석형(선문대학교), 양해술(호서대학교) · 223
12. .Net Framework 환경에서 한국형CMS(Contents Management System)설계 및 구현(kips_0290)	이정태*(고려대학교) · 227
13. 실시간 임베디드 소프트웨어 모델링을 위한 xUML 확장에 관한 연구(kips_0300)	김우열*, 김영철(홍익대학교) · 231
14. IT 조직의 효과적인 프로세스 개선을 위한 사례연구(kips_0324)	송기원*(중앙대학교), 김진수(건양대학교) · 235

[

포 스타

]

1. 소프트웨어 신뢰성 품질 평가에 대한 표준화 연구(kips_0017)	정혜정*(평택대학교), 정원태(경문대학), 정영은, 신석규(TTA) · 241
2. 위험기반 테스트에서 제 3자 시험기관의 위험요소 분석 연구(kips_0040)	이상복*, 김기두, 박정환, 신석규(한국정보통신기술협회) · 245
3. 임베디드 S/W의 체계적 재사용을 위한 재사용 체계(kips_0043)	유미선*, 차정은, 양영종(한국전자통신연구원) · 249
4. 필드 위치결정을 위한 리팩토링 요인(kips_0134)	정영애*, 박용범(단국대학교) · 253
5. AOP 코드 이해를 지원하는 애스펙트 클래스 참조 테이블(ACRT)(kips_0183)	박옥자*, 박종각, 유철중, 장옥배(전북대학교), 신현철(백석문화대학) · 257
6. S/W 로지스틱 테스트 노력함수의 적정성에 관한 연구(kips_0056)	최규식*(건양대학교) · 261
7. X-Internet 기반 웹 UI 시스템 개발에 관한 연구(kips_0112)	김귀정*(건양대학교) · 265
8. 온톨로지를 이용한 S/W Product line 도메인의 명시적 feature 분석 모델(kips_0250)	이순복*(고려대학교), 이태웅(한국과학기술정보연구원) 김진우, 백두권(고려대학교) · 269

9. PSP/TSP-6시그마 도구 적용 방법론에 관한 연구(kips_0274) 박영규*, 최호진, 백종문(한국정보통신대학교) · 273
10. PSP 지원을 위한 개인 메트릭 자동 수집 및 분석 도구 개발(kips_0283) 신현일*, 최호진, 백종문(한국정보통신대학교) · 277
11. 내장형 소프트웨어 품질평가를 위한 툴킷의 개발(kips_0287) 이충환*(호서대학교), 이하용(서울벤처정보대학원대학교), 양혜술(호서대학교) · 281
12. 임베디드 시스템 소프트웨어 개발을 위한 정형적 접근(kips_0299) 이수영*, 김진현, 최진영(고려대학교) · 285
13. Executable xUML을 이용한 U-Home 제어 시스템을 위한 모델링 연구(kips_0360) 김예진*, 김영철(홍익대학교) · 289

5. 표
...
6. 자
...
7. 표
...
8. 표
...
9. 사

(0350) 인공지능 연구 논문집 인공지능 연구 논문집 인공지능 연구 논문집

구 두

1. 상태 인식에 따른 자율 주행 에이전트 시스템(kips_0035) 정슬기*, 이태경(동국대학교) · 295
2. 얼굴패턴 검출 문제에서 WFMM 신경망 기반의 피부색 검출 기법(kips_0070) 조일국*, 김호준(한동대학교) · 299
3. 사용자 행동패턴을 기반으로 한 멀티 에이전트 시스템 구조(kips_0077) 김민경*(한국전자통신연구원) · 303
4. 제품 영상을 이용한 제품 설계 정보 검색 시스템(kips_0129) 이형재*(전남대학교), 김용일(호남대학교), 양형정(전남대학교) · 307
5. 전자상거래에서 멀티 이슈 기반의 에이전트 협상 방법(kips_0166) Zhang Xiaoxuan*, 조근식(인하대학교) · 311
6. FMM신경망을 이용한 OSD 메뉴 검증기법(kips_0251) 이진석*, 백정민, 김호준(한동대학교) · 315
7. 유비쿼터스 환경에서 자가 치유를 지원하는 하이브리드 예측 모델(kips_0344) 유길종*, 박정민, 이은석(성균관대학교) · 319

1.
2.
[
1.

포 스타

1. 다중 계층 퍼셉트론의 교대학습 알고리즘(kips_0027) 최범기, 이주홍, 박태수*(인하대학교) · 325
2. 계획 시스템에서 이정표상태 생성에 관한 연구(kips_0087) 신재혁*, 한현구(한국외국어대학교) · 329
3. ASN.1 기반의 온톨로지를 이용한 시각 미디어 서비스 제공자 랭킹 방법(kips_0092) 민영근*, 이복주(단국대학교) · 333
4. 문맥광고에서 관련 사이트 추천을 위한 연관 키워드 마이닝기법(kips_0096) 김성민*, 이성진, 이수원(숭실대학교) · 337

1
2
3
4
5

실시간 임베디드 소프트웨어 모델링을 위한 xUML 확장에 관한 연구

김우열*, 김영철

홍익대학교 컴퓨터정보통신 소프트웨어공학연구소
e-mail:john*@selab.hongik.ac.kr

A Study on Extension of Executable UML for Modeling Real-time Embedded Software

Woo-Yeol Kim*, R. Young-Chul Kim

Dept of Computer Information and Communication,
Hong-Ik University

요 약

현재까지는 실시간 임베디드 소프트웨어 개발을 위한 효율적인 소프트웨어 모델링 언어가 부족하다. 그런데 모델 자체가 코드처럼 수행 가능한 통합 모델링언어를 xUML(Executable UML)이라 한다[2,4,7]. 이는 기존의 UML x.x에 실행과 관련된 개념과 시간에 관련된 규칙을 더한 것이다. 다시 말해 xUML의 모델은 실행과 테스트, 디버깅이 가능하다[2,4]. 본 논문에서는 기존의 UML x.x버전들과 xUML이 실시간 임베디드 소프트웨어를 모델링 하는데 적합한지를 비교/분석한 후, 임베디드 소프트웨어 모델링에 xUML을 적용하고자 부족한 면을 보완 및 확장하였다. 확장된 xUML의 노테이션은 병렬과 실시간 처리까지도 표현이 가능하도록 제안하였다. 사례 연구로서 두개의 터치센서로 동작하는 실시간 임베디드 시스템의 모델링을 보여준다.

1. 서론

최근 실시간 임베디드 시스템의 소형화 및 고성능화가 진행됨에 따라 제품 핵심이 하드웨어 기술에서 소프트웨어 기술로 이동하고 있다. 초창기 임베디드 소프트웨어는 간단한 제어 프로그램만으로 산업용 기기를 제어하는데 그쳤으나, 최근에는 멀티미디어 처리나 항공 시스템등과 같은 복잡하고 커다란 기기를 제어하고 있다. 이렇듯 복잡한 기능을 제공하는 소프트웨어를 좀 더 효율적이며 안정적으로 개발하기 위해 임베디드 소프트웨어를 모델링 하는 연구가 활발히 진행 중이다[1].

보통 일반적인 소프트웨어를 모델링하기 위해서 UML을 사용한다. 기존의 UML x.x은 소프트웨어 시스템을 모델링하기 위한 언어로서 사용하며 소프트웨어 개발자들은 시스템에 대한 모델을 만들 수 있다[2]. 하지만 복잡하고 주위환경에 신속하게 대처

해야 하는 실시간 임베디드 시스템은 서비스의 질(QoS or Quality of Service), Low-level 프로그래밍 그리고 안전성과 신뢰성에 대한 특별한 고려가 필요하다[3].

일반적인 소프트웨어의 모델링과는 달리 모델 자체가 코드처럼 수행 가능한 통합 모델링언어를 xUML(Executable UML)이라 한다[2,4,7]. xUML은 기존의 UML x.x에 실행과 관련된 개념(executable semantics)들과 시간에 관련된 규칙(timing rules)들을 더한 것이다. xUML의 모델은 실행과 테스트, 디버깅이 가능하며, 시스템 성능 측정까지 가능하다.

본 논문에서는 기존의 UML x.x버전들과 xUML이 실시간 임베디드 소프트웨어를 모델링 하는데 적합한지를 비교/분석한 후, 임베디드 소프트웨어 모델링에 xUML을 적용하고자 부족한 면을 보완 및 확장하였다. 확장된 xUML의 노테이션은 디지털 공학의 논리게이트 모양을 참고하였으며 Fork/Join, Concurrency 등 병렬과 실시간 처리까지 표현이 가능하다.

* 본 연구는 정보통신연구진흥원 정보통신기초기술연구지원사업(B1220-0501-0321) 지원으로 수행되었음.

본 논문의 구성은 다음과 같다. 제2장에서는 관련연구로서 Executable UML에 대해 알아보고 xUML과 각 버전별 UML을 비교/분석한다. 제3장에서는 제안한 xUML의 확장에 대해 설명한다. 제4장에서는 확장된 xUML 노테이션을 이용하여 실시간 임베디드 소프트웨어를 모델링 해본다. 마지막으로 제5장에서는 결론 및 향후연구를 언급한다.

2. 관련연구

Executable UML[2, 4]은 기호로 스펙을 설명하는 언어이다. Executable UML은 UML(Unified Modeling Language) 표기법에 "실행에 관련된 개념(executable semantics)들"과 "시간에 관련된 규칙(timing rules)들"을 더한 것이다. Executable UML을 이용하면, 클래스(class)와 상태(state)와 액션 모델(action model)로 이루어진, Executable 시스템 스펙을 만들 수 있다. Executable 시스템 스펙은 하나의 완전한 프로그램처럼 실행된다. 기존 스펙과 달리 Executable 스펙은 실행과 테스트, 디버깅이 가능하며, 시스템 성능 측정을 위해서도 이용할 수 있다. 스펙(모델)에 대한 테스트가 끝나면, 이를 타겟 코드(target code)로 변환(translate)할 수 있다.

타겟 코드는 단일 프로세서 환경에서 실행될 수도 있고, 여러 개의 프로세서들로 이루어진 프로세서 네트워크 환경에서 실행될 수도 있다. 기존의 모호한 모델과 달리, Executable 모델은 시간에 관련된 문제와 동기화에 관련된 문제, 그리고 자원의 획득과 이용에 관련된 문제들을 자세히 서술한다. 결론적으로, Executable UML은 복잡한 실시간(realtime) 분산(distributed) 임베디드(embedded) 시스템의 스펙을 서술하기에 적합하며 많이 이용된다.

<표 1>은 xUML의 장점을 파악하기 위해 각기 다른 버전의 UML들을 비교해 놓은 것이다. UML은 크게 분석 능력과 설계 능력, 구현 능력, 그리고 테스트/디버깅 능력으로 나누어 비교할 수 있다. 우선 xUML은 분석 능력에서 이벤트를 모델링하고 구현 전에 설계 변화를 분석할 수 있다는 장점이 있다. 설계 능력으로는 동적 모델링에서 중요시 되는 동시 발생 문제를 표현하고 모델과 구현의 일관성이 유지된다는 장점이 있다. 또한 설계 변경이 용이하므로 여러 플랫폼(타겟)에서 재사용이 가능하고, 설계 도중 문제를 발견 및 수정할 수 있다. 구현 능력으로는 xUML은 기존의 스펙레톤 코드가 아닌 동적 요소를 포함한 완벽한 코드가 발생된다는 것이다. 마

지막으로 테스트/디버깅 능력은 실행 모델이기 때문에 디자인 레벨에서의 디버깅이 가능하다.

<표 1> UML 버전별 비교[2]

능력 구분	UML 1.x	UML 2.0	Embedded UML	xUML
주요 시장	실업적/미즈니스 어플리케이션	상업적/미즈니스 어플리케이션/실시간 모델	SoC	임베디드/실시간 모델
분석 능력	x	o	o	o
이벤트 모델링	x	o	-	o
구현 전 설계 변화 분석	x	o	-	o
설계 능력	o	o	-	o
Fork/Join 기능	x	x	x	x
동적 모델링에서 Concurrency 모델링	x	o	-	o
설계와 구현 사이의 일관성이 유지	x	o	x	o
설계 변경이 용이	x	x	-	o
여러 플랫폼에서의 설계 재사용	x	o	-	o
설계 단계에서 문제를 발견 및 수정	x	o	-	o
구현 능력	x	x	x	o
코드 생성	스펙레톤코드	VM 기반 코드	VM 기반 코드	동적요소 포함한 완벽한 코드
모델과 코드의 연관	x	o	x	o
실시간/임베디드 시스템의 구현능력	x	o	-	o
테스트/디버깅 능력	x	x	x	o
실행 모델	x	x	x	o
디자인 레벨에서의 디버깅	x	x	x	o

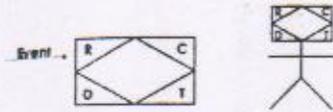
하지만 이러한 xUML 조차도 실시간 임베디드 소프트웨어를 모델링 하는데 한계점이 존재한다. 그래서 다음 장에서 확장된 xUML을 제안하였다.

3. 확장된 xUML 노테이션

기존의 상호작용 다이어그램들[2, 5]은 서로 유한 기호들과 의미를 내포하고 있으나 병렬과 실시간에 대해 지원하지 않고 있다. 그러나 임베디드나 병렬/실시간 시스템은 모든 이벤트들이 순차적으로 발생하는 경우에만 존재하는 것은 아니다. 이런 문제를 해결하기 위해서 xUML의 확장과 관련된 아이디어가 필요하다.

31. 객체의 역할

(그림 1)과 같이 확장된 xUML에서 객체는 ERCDT(Event/Recognition/Communication/Decision/Transaction)의 역할 구조를 가지고 있다. 이는 객체가 각자의 역할을 수행하는 것을 의미한다. 그리고 액터 역시 객체의 하나로서 역할을 가진다. 본문에서는 액터의 역할이 분명하도록 시각적으로 나타내었다.



(그림 1) 객체의 역할

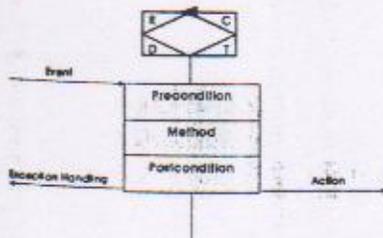
(그림 2)에서 (a)와 같이 인터페이스 오브젝트는 이벤트가 들어오면 인지해서 통신하는 ERC의 구조를 가지고 (b)와 같은 컨트롤 오브젝트는 인터페이스의 기능 외에도 결정을 내리고 수행하는 ERCDT의 모든 구조를 가질 수 있다. 또한 (c)에 해당하는 서비스 오브젝트는 이벤트가 들어오면 수행을 하게 되는 ET의 구조를 가지게 된다.



(그림 2) 객체의 역할 예

32. 객체의 규칙

(그림 3)은 ECA(Event/Condition/Action) 규칙을 가진 객체의 내부 처리과정을 도식화 한 그림이다. 이벤트가 들어오면 객체 내부에서 조건에 맞는지 검사한 후, 조건에 맞으면 액션을 수행하게 되고 맞지 않으면 예외처리를 하게 된다.

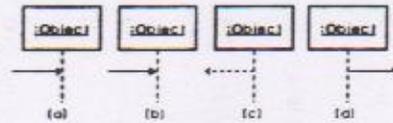


(그림 3) 객체의 규칙

33. 메시지

확장된 xUML의 메시지는 (그림 4)와 같이 4가

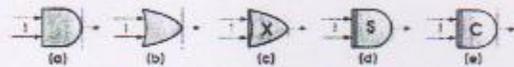
지 타입으로 구성된다. (a)는 일반적인 Flat Message를 나타내며 (b)는 동기 메시지고 (c)는 리턴 메시지 (d)는 비동기 메시지를 나타낸다.



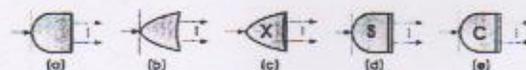
(그림 4) Messages

34. 이벤트

(그림 5, 6)은 Incoming Events와 Outgoing Events로서 (a)는 AND 개념을 나타내는 것으로 이벤트가 모두 들어오면 실행이 된다. (b)는 OR 개념을 나타내며 이벤트 중에서 하나만 들어오게 되어도 다음 액션이 실행된다. (c)는 Exclusive OR을 나타내는데 이벤트 중 하나를 선택해서 액션이 실행되고, (d)는 Sequential 개념으로서 이벤트가 차례대로 들어온다. (e)는 Concurrent 개념으로 동시에 이벤트가 들어오고 다음 액션이 수행된다. 이벤트들은 다시 Binary Events와 Multiple Events로 나뉜다.



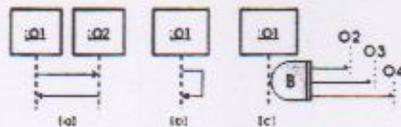
(그림 5) Incoming Events



(그림 6) Outgoing Events

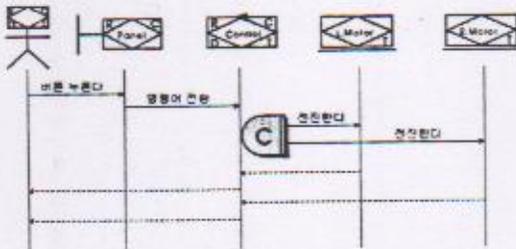
35. Communication

확장된 xUML은 3가지의 통신 방법이 있다. (그림 7)은 Communication의 그림이다. (a)는 객체 간 통신인 Peer-to-Peer를 나타내고, (b)는 자기 자신과의 통신인 Message to Self를 나타내며, (c)는 비동기 메시지를 보내기만 하는 Broadcasting을 나타내고 있다.



(그림 7) Communication

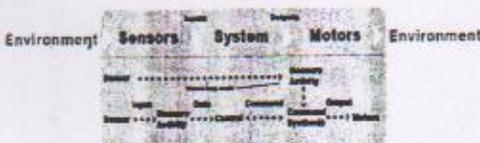
(그림 8)은 자동차가 두 개의 모터에 동시에 메시지를 전달하는 모습을 확장된 xUML을 이용하여 그려낸 것이다.



(그림 8) 자동차의 전진

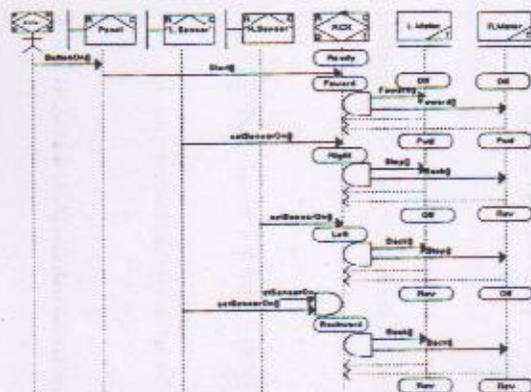
4. 확장된 xUML을 사용한 모델링 사례

본 논문에서는 임베디드 소프트웨어를 모델링 할 수 있는 확장된 xUML을 제안하였다. (그림 9)는 실시간 임베디드 소프트웨어 시스템의 구조를 나타낸다. 이는 센서에 값이 들어오면 시스템에 메시지를 보내고 시스템은 실시간으로 반응하여 모터에 움직임 값을 보내주는 것을 도식화 한 것이다.



(그림 9) 실시간 임베디드 S/W 시스템의 구조[6]

(그림 10)은 확장된 xUML을 사용하여 두개의 터치센서로 동작하는 실시간 임베디드 시스템의 동적 모델링을 나타낸 것이다.



(그림 10) 동적 모델링

이는 정상적인 주행 중에 왼쪽 터치 센서에 충격이 가해지면 RCX라 불리는 제어시스템이 왼쪽 모터에는 전진 메시지를 보내고 오른쪽 모터에는 멈춤 메시지를 동시에 보내 오른쪽으로 방향을 전환하게 된다. 오른쪽 센서에 충격이 가해지면 반대로 동작한다. 또한 두개의 터치센서에 동시에 충격이 가해지면 양쪽 모터 모두 후진 메시지를 보내게 된다. 또한 다이어그램에 각 객체의 상태 변화도를 포함시켜 한 장의 다이어그램만으로 임베디드 시스템의 동적 모델링을 표현할 수 있었다. 그림에서 알 수 있듯이 데이터 값을 포함하는 액티브 오브젝트만이 상태 변화도를 가진다.

5. 결론

본 논문에서는 확장된 xUML을 제안함으로써 임베디드 시스템의 동적 모델링을 향상시켰다. 이는 기존의 시퀀스 다이어그램에 AND/OR/XOR 등의 개념과 Concurrency와 Fork/Join등의 개념을 추가함으로써 임베디드 소프트웨어 시스템의 동시 발생 문제 등의 모델링이 가능하였다.

향후 연구과제로 xUML을 이용하여 개발한 모델을 검증하기 위한 연구와 시뮬레이션뿐만이 아닌 코드 생성까지 자동화 할 수 있는 도구가 필요하다. 최종적으로는 우리가 확장한 xUML을 간단한 로봇이 아닌 커다란 통신 시스템에도 적용되도록 연구를 진행 중이다.

참고문헌

- [1] Axel Jantsch, Modeling Embedded System and SOCs, Mogan Kaufmann, 2004.
- [2] 김우열, MDA 기반의 임베디드 소프트웨어 모델링에 관한 연구, 홍익대학교 석사학위논문, 2005.
- [3] 이호수, 유비쿼터스의 핵심기술: 임베디드 시스템, IBM, 신기술 신경염, Vol.9, Spring, 2004.
- [4] 김인기, UML 설계와 응용: 클래스 모델 만들기, 정보문화사, 2003.
- [5] Rumbaugh, J., Object-Oriented Modeling and Design, Prentice Hall, Englewood Cliffs, NJ, 1991.
- [6] Jean-Louis Houberton, Jean-Philippe Babau, "MDA for embedded systems dedicated to process control," Workshop on MDA in SIVOEES, in conjunction with UML'2003, October 2003.
- [7] Mellor, K. Scott, A. Uhl, D. Weise, "MDA Distilled", Addison-Wesley, 2004

1. 서론
오늘날 가장
효율적인 프
와 같은 소
하고 있다.
태에 대한
제공한다[1].
제시하고 있
효율적인
CMMI와 같
과 위험을 소
환경에 적용
을 높여야한
지속적인
성능을 측정