보안 임베디드 소프트웨어 개발을 위한 융합 모델

김행곤[1)]

Security Convegence Model for Embedded Software Development

Haeng Kon Kim[1)]

요 약

최근 융·복합에 대한 관심이 높아지면서 융합형 임베디드 소프트웨어 응용 분야 및 이를 개발하기 위한 다양한 방법론 및 도구에 대한 연구가 활발히 진행되고 있다. 특히, 임베디드 소프트웨어의 생산성 및 품질 향상을 위한 개발 방법론 및 테스팅에 관한 연구가 필수적으로 요구되고 있다. 따라서 임베디드 응용에 적합한 개발 모델 및 절차에 대한 체계적 연구가 필요하며 인증된 모델 융합이 요구된다.

본 연구에서는 소프트웨어 신기술인 소프트웨어 모델인 MDA(Model Driven Architecture) 와 프로덕트 라인 그리고 CBD기술을 도입하여 응용 임베디드 도메인에 적합한 소프트웨어 개발 도구 및 환경 지원을 위한 통합 모델을 연구 한다. 플랫폼-독립적인 임베디드 소프트웨어 개발 모델과 플랫폼-종속적인 영역을 관심의 분리를 통해 분리하여 추후 구현단계에서 융합하는 임베디드 소프트웨어 개발 지원 통합 모델 기법을 제안하고 임베디드 소프트웨어 생산성 및 품질을 높이게 한다.

핵심어 : 임베디드 소프트웨어, 개발 방법론 및 테스팅, 모델 융합, MDA(Model Driven Architecture), 플랫폼-독립적 개발 모델, 플랫폼-종속적 개발 모델,

Abstract

Recently, There are many active researches of methodologies and tools for embedded software applications to meet convergence and integration areas. It requires development methodologies and tools for high productivity and quality. First of all, procedural and aproved model convergence is required to meet it.

In this paper, We suggest software development tools and environment, as convergence model, using MD, Product line and CBD for specific model. We approach two different models as PIM (Platform Independent Model) and PSM (Platform Specific Model) with seperate of concern. We finally describe the convergence model for embedded software for high quality and productivity.

Keywords : Embedded Software, Develpment Methodology and Testing, Model Convergence, MDA, PIM (Platform Independent Model) and PSM (Platform Specific Model)

접수일[2010년08월15일), 심사의뢰일(2010년08월16일), 심사완료일(1차:2010년08월30일, 2차:2010년09월12일)

게재일(2010년10월31일)

[1]712-702, 대구가톨릭대학교 공과대학 컴퓨터정보통신공학부 교수

email: hangkon@cu.ac.kr

* 이 논문은 2010년도 대구가톨릭대학교 교내연구비 지원에 의한 것임

1. 서론

임베디드 시스템의 기능이 다양화되고 복잡해질수록 임베디드 소프트웨어도 복잡해지며 이에 대한 체계적인 개발 방법, 도구 및 유지 보수 방안이 적용되어야 한다. 이러한 고 난이도 임베디드 시스템을 안전하게 개발하기 위해서는 시스템 특성에 따른 다양한 요구를 반영한 기능을 효과적으로 개발하는 환경 및 도구의 개발이 절실하다[1,2]. 임베디드 시스템은 사용자나 외부의 입력에 대하여 어떤 응답 또는 처리를 하도록 설계되어 있다. 즉, 시스템 내에 논리적으로 정의되었던 기능들이 순서에 따라 정확하게 수행되어야 하며 실시간을 중요시 여기는 임베디드 시스템인 경우 기능의 수행이 정확한 시간 내에 이루어져야 한다는 조건을 만족 시켜야 한다. 이러한 시스템은 일반적인 응용 소프트웨어와는 다른 방식으로 개발되며 소프트웨어 외에 하드웨어에 대한 지식이 요구되므로 신뢰성 있는 소프트웨어 시스템을 개발한다는 것이 매우 어려운 일이다. 임베디드 시스템의 기능이 다양화 될수록 임베디드 소프트웨어는 시스템에서의 역할 비중은 더욱 높아질 것이다. 좋은 임베디드 시스템을 개발하기 위해서는 하드웨어와 소프트웨어 기술이 여러 제약사항을 고려하여 균형 있게 적용되어야 한다[3,4].

기존 소프트웨어 개발 방법론으로는 최근의 임베디드 소프트웨어 개발 적용에 한계가 있다. 따라서 소프트웨어 신기술인 소프트웨어 모델인 MDA(Model Driven Architecture)[5] 와 프로덕트 라인[6] 그리고 CBD기술을 도입하여[7] 응용 임베디드 도메인에 적합한 소프트웨어 개발 도구 및 환경 지원을 위한 통합 모델을 연구 한다. 플랫폼-독립적인 임베디드 소프트웨어 개발 모델과 플랫폼-종속적인 영역을 관심의 분리를 통해 분리하여 추후 구현단계에서 융합하는 임베디드 소프트웨어 개발 지원 통합 모델 기법을 제안한다.

2. 관련 연구

2.1 MDA

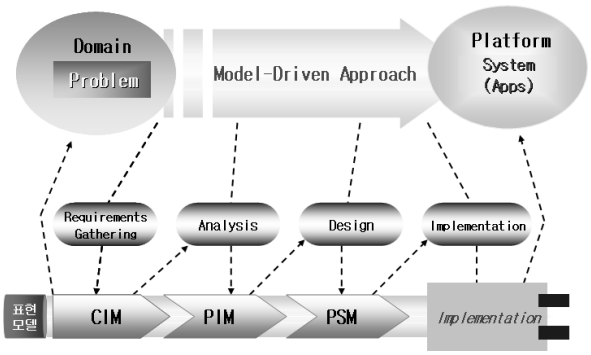
소프트웨어 개발을 위한 아키텍처인 MDA는 OMG(Object Management Group)에 의해서 정의되었다[8]. MDA는 설계 모델을 점진적으로 변환하여 S/W를 자동으로 생성하는 새로운 개발 방식이다. CBD(Component Based Development)의 특징은 도메인 컴포넌트를 통해 큰 단위의 재사용이 가능하고, 여러 컴포넌트 사용자 마다 서로 다른 요구사항을 지원하기 위한 커스트마이제이션 기법을 제공하고, 인터페이스와 구현을 분리함으로서 유지보수성을 증가시킬 수 있다. 그러나 컴포넌트 제품은 컴포넌트 미들웨어 플랫폼에 종속적이다[9]. 따라서 컴포넌트의 미들웨어가 변경된다면, 그 컴포넌트 미들웨어에 적용하기 위해 새로 설계하고, 개발해야 하므로 중복 노력이 필요하고, 생산성이 낮아지게 된다. 이를 지원하기 위한 MDA는 Platform 기술로부터 Business 혹은 Application Logic을 명확하게 분리하는 새로운 Software 개발 접근 방법으로 3개의 설계 모델을

다음과 같이 제안하고 있다[10].

·요구사항에 대한 정보를 수집하여 생성되는 모델의 CIM(Computing Independent Model)

·구현에 사용될 Program Language, System S/W, Networking등 특정 환경에 종속적이지 않는 설계 모델인 PIM(Platform Independent Model) 구현환경의 특징을 고려한 상세 설계 모델인 PSM(Platform Specific Model)

·모델지향적 구조인 MDA를 이용한 시스템 개발에 대한 기초적인 개념은 도메인 영역에서 제시되는 문제에 대해 요구되어지는 플랫폼 환경에 맞는 어플리케이션을 위한 모델 지향적 접근을 통하여 해결되어지는 방법이다. MDA의 기초적인 개념을 [그림 1]에서 나타낸다.



[그림 1] MDA의 기초적인 개념

[Fig. 1] MAD Basci Concepts

표현모델에서의 CIM단계는 요구사항에 대한 정보를 수집하여 생성되는 모델이다. CIM에 CIV(Computing Independent Viewpoint)라는 컴퓨팅에 대한 독립적인 관점을 통해 분석단계를 거치게 된다. 다음 단계는 PIV(Platform Independent Viewpoint)로써 플랫폼에 독립적인 관점을 통해 구현에 사용될 Program Language, Sysetm S/W, Networking등 특정 환경에 종속적이지 않는 설계 모델인 PIM을 생성하게 된다[11].

이때 분석 다음 설계단계의 과정을 수행하기 전에 구현 환경의 특징을 고려한 플랫폼에 종속적인 PSV(Platform Specific Viewpoint)를 통하여 상세 설계 모델인 PSM을 생성하고 그 다음 .exe와 같은 컴포넌트로 생성되는 구현단계를 거쳐 플랫폼에 맞는 어플리케이션을 도출한다[12].

2.2. YES : Yamacraw Embedded System

소프트웨어 공학 기술들을 임베디드 소프트웨어 개발의 영역으로 도입하기 위한 효율적인 통합 환경을 구축하며 프로세스는 도메인 모델링, 제품 계열 설계 및 구현, 제품 개발을 위한 3단계로 구성되며, 각 심볼들의 의미는 다음과 같다[13].

·Domain Knowledge{W} : 도메인 환경에서 잘 알려진 주요 정보 요소들

·Scope{d} : 전략적 목적이나 조직의 임무
·Domain Model{Wd} : 특정 도메인{d} 내의 물리적 세계{W}에 대한 지식
·Requirement{R} : 최종 사용자(end-user) 관점에의 요구
·Specification{S} : 최적의 제품{P}이 구축하기 위한 {Wd}의 정확한 서술
·Target Platform, Machine{M} : 제품과 자산이 실행될 특정한 컴퓨팅 환경
·Generic Solution{G} : 문제에 대한 설계 대안들(설계 패턴, 프레임워크 등)

모델과 제품의 V&V(Validation&Verification)를 지속적인 사이클을 통해 수행함으로써, 임베디드 시스템의 개념적 요소들과 구체적인 시스템 가공물로의 명확한 매핑을 지원하며 소프트웨어 공학의 기존 기술들을 바탕으로 제품 계열 개발 원칙들과 다중(Cross-cutting) 관점들을 캡슐화하여 모델을 개발하고 이를 단일화된(Universal) 표기법을 통해 제공하는 재사용가능한 IP 라이브러리를 구축한다[14].

2.3 MoBIES (Model Based Integration of Embedded Software)

응용 도메인마다 가지는 특정한 요구에 따라서 시스템 모델들을 조립하거나 맞춤하는 차세대 시스템/소프트웨어 co-design 기술 개발을 목적으로 한다. 여러 연구 기관에서 MoBIES 전체의 목적 달성을 위한 세부 기술들을 분리하여 독립적으로 수행한다. 이 시스템은 모델 기반 통합 기술(Model-Based Integration technology)을 근본 기술로하며 임베디드 시스템의 각 관점에 따른 다양한 형상들에 대한 모델 기반의 프로그래밍 환경 및 모델 기반 코드 생성 기술을 개발한다. 이를 위해 물리적 관찰 내용과 정보 프로세스가 통합된 모델을 구축하고 관리하며, 모델에 대한 다양한 관점들을 프레임워크 모델의 맞춤 인터페이스로 대응시키며 모델 분석 도구 개발을 위한 모델 언어의 처리 기술을 개발하고, 물리적 제약 상의 다중 수행(cross-cutting)을 만족하는 임베디드 시스템을 구성한다[15].

2.4 PECOS

임베디드 시스템의 Component-based software development를 가능하게 하며 이를 위해 소프트웨어 모델 기반의 임베디드 시스템의 명세화(specification), 조립(composition), 형상 점검(configuration checking) 및 배치(deployment)를 지원하는 환경을 제공한다. 프로젝트의 주요 결과물은 다음과 같다[16].

·Component model : 행위 명세, 비기능적 속성 및 제약사항을 나타내는 임베디드 시스템 모델
·Composition language : Component model에 기반한 언어로 모델과 조립 명세화 하는데 사용됨
·Composition environment : 모델로부터 임베디드 시스템을 조립하고 기능 및 비기능적 조립 제약사항을 검증하는데 활용됨. 또한 임베디드 디바이스를 위한 실행 가능한 응용 프로그램을

생성하고 그 프로그램을 모니터링하는 데에도 활용됨

·Ultra-light component environment : 자원 제약적인 임베디드 시스템에 모델 기반의 응용 프로그램을 설치, 운영, 시험 및 조정 위해 활용됨 모델의 구성요소

임베디드 시스템에 적합한 세 가지 모델 유형은 다음과 같다.

·Active component : 자신의 thread of control을 갖고 장시간 동안의 활동 및 진행상태를 모델링하는데 사용됨 (e.g., ModBus)

·Passive component : 자신의 thread of control을 갖고 있지 않으면서 synchronous한 행위를 짧은 시간동안 수행하는 모델을 모델링하는데 사용됨 (e.g., ProcessApplication)

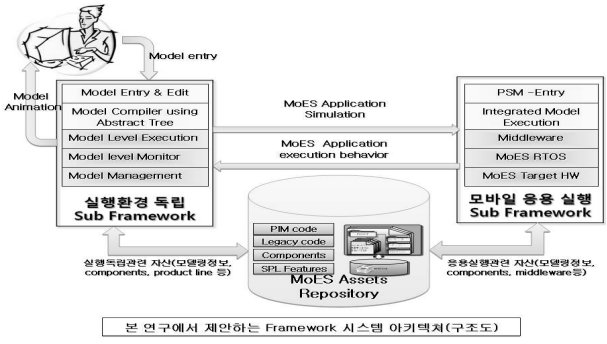
·Event component : 이벤트에 의해 자신의 기능이 유발되는 모델을 모델링하는데 사용됨

·Port : 모델 간의 정보교환을 하는데 사용하며 name, type, range, direction의 값을 갖고 있음

·Connector : Ports 간의 data-sharing 관계를 정의하며 그것이 연결하는 name, type 및 그 것이 연결하는 ports의 리스트를 갖고 있음

3. 융합 모델 프로세스 및 프로토타이핑

본 논문에서 제안하는 융합 모델은 모바일 임베디드 소프트웨어 개발을 위한 다양한 수준별 형상 모델 처리를 위한 실행환경 독립 관점에서 요구 분석을 진행한다. 분산된, 이질적 환경 및 실시간 처리를 고려하며 행위 타입으로서 모델 정의와 실행과 이들 실행 환경을 관리하는 모니터링 기법이 요구되며 이들 이잘적 자료들을 저장, 조직 그리고 검색을 지원하는 자산의 저장소 구축과 연관된 기술이 필요하다. 이를 위한 모바일 임베디드 S/W 개발 프로세스 상에서 재사용 가능한 자산들의 체계화 모델과 컴포넌트 그리고 프로덕트 및 아키텍처의 효과적인 관리 지원 기술등이 요구된다. 한편 관리 모델에서 자산 코드의 자동 생산 기술과 다양한 수준의 다양한 플랫폼에 실행가능한 모바일 임베디드 소프트웨어 환경 구축에 관한 기술 두구의 설계 및 구현이 필요하게 된다. 본 논문에서 연구되는 융합 모델의 범위와 개발 도구의 전체 구조도는 [그림 2]와 같다.

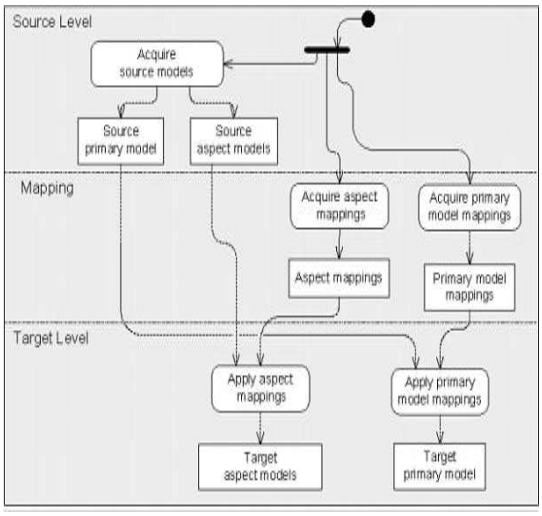


[그림 2] 모델 융합 지원 시스템 아키텍처(구조도)

[Fig. 2] Model Convergence System Architectures

3.1 실행환경 독립 서브 프레임워크 개발

상위 단계에서 모바일 임베디드 관련 도메인 요구사항 분석과 소프트웨어 아키텍처 분석/개발, 모델 개발 단계를 가진다. 하위 단계에서 추상화 트리를 사용하여 모델을 컴파일하며 모델 단계에서 실행, 모니터링 그리고 관리 기능을 가진다(이상 PIM). 또한 MoES(Mobile Embedded Software) 제품 생산을 위한 컴포넌트와 소프트웨어 생산라인을 개발하고, 실제 다양한 MoES 제품 생산과정에서 공용으로 사용되어질 자산 즉, MoES를 위한 융합 모델들을 통합 관리하는 저장소(repository)와 긴밀한 상호 협력(interaction)을 가진다. 구축할 임베디드 소프트웨어 도메인 모델이 어떤 것인지를 결정하며 더 높은 수준의 표현을 제공하기 위해 기존의 UML 메타모델을 확장한다. 또한 모바일 소프트웨어 모델링 정보를 소스와 타겟 그리고 이들 간의 관련을 지원하는 mapping 단계와 모델 융합 단계의 적용 프로세스를 제안 한다. 새로운 구성요소로 작성되는 다이어그램을 기술하며 모델 개념과 관련성을 기호와 요약 정보를 제시한다. 이를 바탕으로 MoES에 포함될 모델의 범위와 제품들의 특징을 추출한다. 또한 분산된, 이질적 환경 및 실시간 처리를 고려하는 모델지원 기술과 추후 요구될 다양한 모델에 대한 사전 분석을 포함하며, 도메인 모델을 생성하기 위해 기존 시스템의 설계와 아키텍처 등의 요구사항을 분석한다.



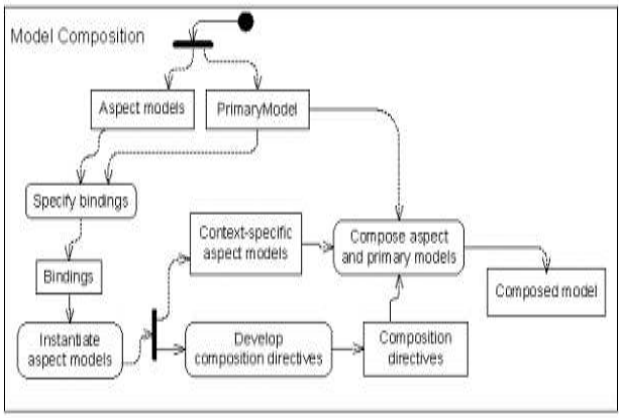
[그림 3] 모델 개발 지원을 위한 초기단계 적용 프로세스

[Fig. 3] First Stage Adatation Process for Model Development

원시단계에서 기본모델과 aspect모델을 획득하여 목표단계로 보낼 때 매핑규칙을 적용시킨 다음 모델을 정제하여 중간단계의 매핑단계를 거친다 [그림 3].

[그림 3] 에서와 같이 원시단계에서 기본 모델과 aspect 모델을 획득하여 최종 목표 단계로 보내는 매핑 규칙을 적용시킨 다음 모델을 정재하는 중간단계 메핑을 정의하는 프로세스를 제안한다.

모델 통합은 aspect를 기본 모델로 바인딩 시켜 모델 인스턴스를 생성하며 문맥 명세 aspect 모델과 통합하여 새로운 통합 모델을 생성하여 플랫폼 독립 모델을 완성시킨다.[그림4]

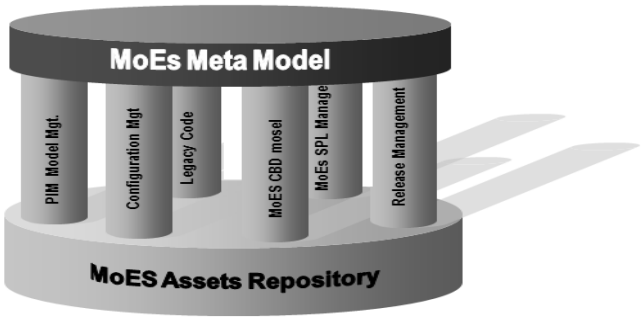


[그림 4] 통합 모델 프로세스 예

[Fig. 4] Convergence Model Procee Example

3.2 Model Integration 지원 저장소(Repository)

모바일 임베디드 소프트웨어에서 재사용과 연계되어 지원이 필요한 소프트웨어 핵심자산들을 할당하고 사용을 조절하며 모니터링 하는 저장소를 개발한다. 이 Reusable 저장소는 미래에 요구되는 임의의 다중 모바일 임베디드 시스템 개발의 솔루션으로 재사용이 가능한 패키지화된 자산이며 이를 통해 임베디드 소프트웨어 개발을 위한 공통성을 위한 설계(Design-for-Commonality)와 다형성의 제어(Control-of-Variability)를 실현할 수 있다.



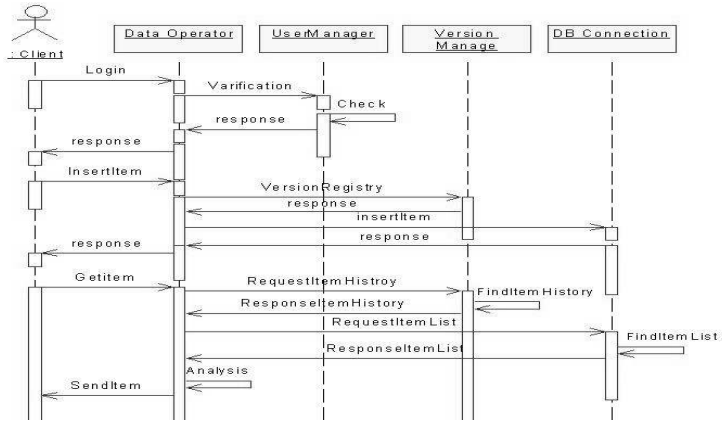
[그림 5] 제안된 MoES 저장소 구조도

[Fig. 5] Proposed MoES Repository Structures

MoES 에 의해 정의되어진 프로세스에 따라 고객이 요구하는 제품에 연관된 모델링 및 구현 관

련 핵심 자산재사용 지원을 위해 필요한 관한 활동들을 제어한다. 수반되는 활동은 기술적·조직적 관리를 포함한 저장소 형상관리, 프로세스 정의, 도메인 스코핑, 기술적 위험 분석이 따른다[그림 5].

기존에 제시되어진 모델 명세 기법에 새롭게 요구되는 모바일 임베디드 관련 명세 특성들을 추가하여 모델링 명세를 작성한다. 이는 실행환경 독립 모델 개발을 위한 명세로서 명확한 의미적인 플러깅 지점을 확보하고 비즈니스 프로세스의 계층적 실현을 위해 모델 명세 항목을 정의한다. 모델링 명세는 임베디드 자산 획득을 위해 작성되는 가이드라인의 역할을 한다. MoES 개념 모델, 정적 모델, 동적 개념 모델과 인터페이스 식별과 명세 활동으로 나뉜다. 실행환경 독립 모델은 요구사항 단계의 Use Case 모델을 중심으로 모바일 관련 모델, 컴포넌트 등을 식별하고 설계하는 전반적인 과정을 포함한다[그림 6]. 분석된 MoES의 목표, 역할, 상호작용, 아키텍처 모델에서 속성들을 모델 개발을 위한 속성과 행위들로 매칭 시키기 위해 모델간 매칭을 시키고 인터페이스를 식별해내는 과정이 수반된다. 이때 작성된 명세와 연관된 자산은 검증단계를 거쳐 저장소에 저장· 관리되어 추후 재사용되도록 한다.



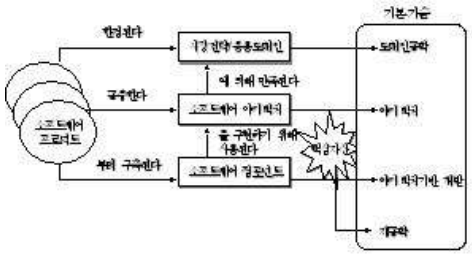
[그림 6] MoES 실행환경 독립 모델 명세 예

[Fig. 6] Platform Independent Model Specification for MoES

MoES에서 동일한 프로덕트 군에 속하는 프로덕트들은 많은 공통성을 가지지만 프로덕트 사이의 모델 변화성도 있다. 모델 변화성은 다양한 사용자와 다양한 설계, 그리고 구현 요구사항들에 따라 제공되고, 도메인 분석 시에 식별된다. 따라서 MoES 모델 변화성은 프로덕트라인 아키텍처 설계 단계에서 고려되어지고, 코드 레벨이 아닌 아키텍처 레벨에서 다루도록 한다. 또한 모델 변화성은 MoES 컴포넌트 조립 시 컴포넌트 행위가 변경될 수 있는 특정 변화점에서 가능하고 컴포넌트 설계 동안에 결정된다. 한 MoES 프레임워크는 응용시스템이나 서브시스템 개발에 필요한 여러 모델들을 모델링하여 포함시키며, 프레임워크는 동일 도메인 내의 여러 응용시스템에 공통성을 기반으로 만들어지며 가변성이 제공되므로 여러 응용시스템에 재사용가능 하도록 한다. 프레임워크

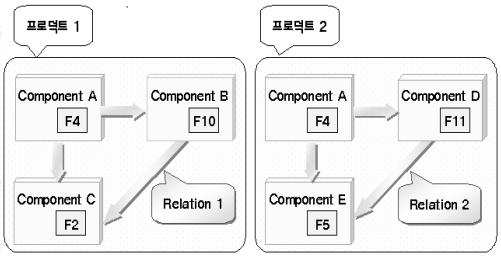
는 그 내부에 응용시스템들이 공통적으로 포함하고 있어야하는 전체적인 계층구조와 계층별 모델들의 상호작용 규칙을 정의한다. MoES 모델들 간의 상호작용은 모델이 가지고 있는 인터페이스를 사용하며 컨넥터는 호출하는 소스 모델의 use 인터페이스와 서비스를 제공하는 타겟 MoES 모델의 provide 인터페이스 사이에 존재하게 한다.

이들 변화성 지원을 위해 프로덕트라인 관련 자산 및 모델을 제안하는 저장소에서 통합 관리하도록 한다[그림 7].



[그림 7.a] .프로덕트 라인 핵심자산과 기술

[Fig. 7.a] Product Line Core Assets

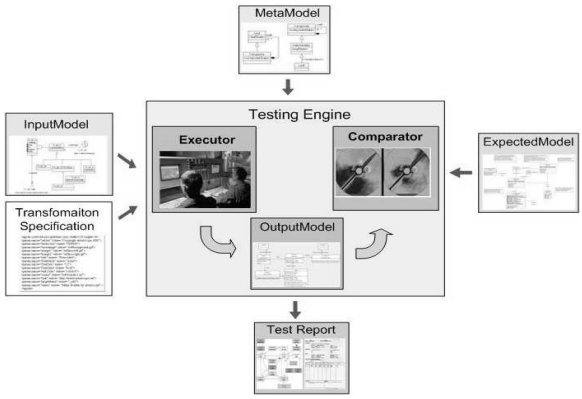


[그림 7.b] 프로덕트라인과 컴포넌트 모델

[Fig 7.b] Product Line and Component Model

3.3. 모바일 서비스 실행 서브 프레임워크

본 서브 프레임워크에서는 실제 환경과 동일한 실행 환경을 제공하여 MoES를 검증하게하고 필요시 플랫폼 독립적 프레임워크에 feedback 하여 update 가능하게 하는 기능을 포함한다.



[그림 8] 모바일 서비스 실행 환경 프레임 워크

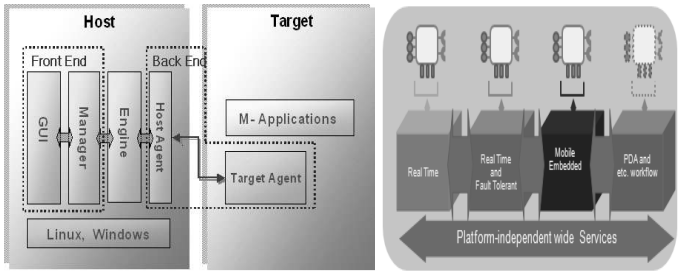
[Fig. 8]Mobile Service execution Environment Frameworks

[그림 8]에서와 같이 MoES 효과적 개발을 지원하고 양호한 품질의 제품을 생산하고 새로운 호

스트나 타겟에 호환성이 있으며 상세한 시스템 지식 없이 사용할 수 있는 타겟 하드웨어 플랫폼을 제안한다. Front-end의 GUI는 사용자로부터 직접 입력을 받아 엔진으로 전달하고 처리결과를 받아 분석 및 재가공하여 사용자에게 전달하고 Front-end Manager는 엔진과 연동하여 엔진으로부터 명령전달과 처리 결과 분석을 담당하며 명령해석기능을 가지게 한다. 엔진은 Front-end로부터 전달받은 명령을 해석하고 실행하며 호스트에서 처리 가능한 경우 직접 호스트에서 처리하고 타겟에서 처리가 필요한 경우 타겟에 전달하여 처리하게 한다.프레임워크에서 데이터 상호 운용성 확보를 위한 모바일 미들웨어(middleware) 관련 기술 정의 MoES 서비스와 사용자 인터페이스 간의 호환성 제공을 위해 MoES 미들웨어 관련 기술을 정의한다. 고려하는 미들웨어는 standard wireless services, mobile routing, mobile web services, remote upgrade and dynamic resource management 들이다.

3.4 MoES 융합 모델의 실행 및 테스팅

실행환경 독립 서브 프레임워크에서 작성되고 융합된 모델은 최종적으로 타겟 환경에서 실행되고 테스팅 단계를 거쳐 평가된다. 융합된 MoES 시스템에 대한 사용자의 신뢰성을 증가시키기 위해 4가지 타입으로 분리하여 의존성 테스팅 방법론 제안한다[그림 9].



[그림 9] 모델융합 후 테스팅 환경도

[Fig. 9] Testing Environment after Model Convergence

Factual checking : MoES 컴포넌트-by-컴포넌트 의존성 검사로, 개별 모델에 대한 다양한 정보를 파악하여 모델 단독, 혹은 융합 모델 내 상호 간의 요구 검증

Inter-model (Pairwise-Model) : 모델 상호 간의 관계성을 검증하기 위해 Call graphs, 파라미터를 포함하는 인터페이스 요구, 사전 조건, 제약 조건, 버전 충돌등을 조사

Aspects Checking : 모델 의미 및 성능 등 프로시저에서 획득할 수 있는 이슈들을 검사하기 위해 End-to-end deadline/real-time, 일치성 요구, 동시성, 오류 허용, 선점과 비선점, 최적화 등을 검사

General Checking : 전역 속성 검증을 목적으로, Deadlock, Protection degree 등을 검사함

3.5. MDA 모델통합에 의한 프로토타이핑

최종 개발 계획하는 프로타이핑 도구는 다음과 같은 기능을 고려하고 있다.

·상호 대화적 방식의 모델의 조립 및 분석 지원

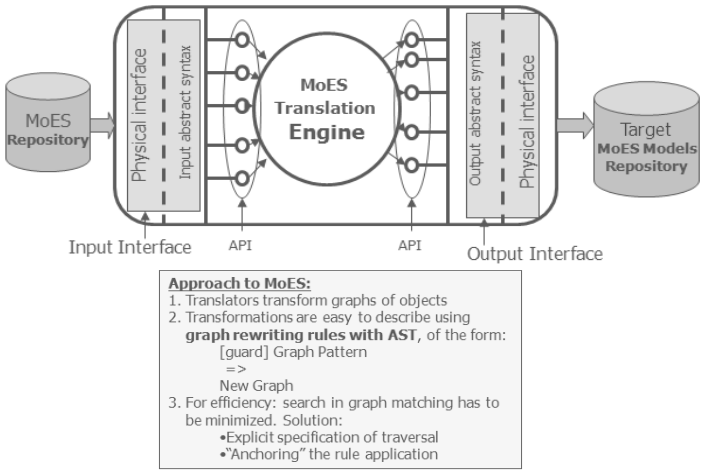
·MoES 자산 관리를 효과적으로 지원하는 저장소

- 도메인 한정적인 S/W, H/W 조립을 통한 Tailored 인프라스트럭처와 제품을 생산하고, 임베디드 시스템 합성 하고 Passive S/W 모델을 active runtime 구조로 매핑
- 런타임 소프트웨어(프로세스, 쓰레드)를 하드웨어로 매핑
- 하드웨어 상에서 aspects 지향의 의존성 체크

·MoES 관련 모델 및 실행 분석

- Real-time 분석, 신뢰성 분석을 위해 off-the-shelf 도구 구동
- memory, power and cost constraints 검사

또한, 기존에 본 연구실에 개발된 임베디드 시스템의 테스팅 및 품질 보증 관련 도구와의 하드웨어, 소프트웨어 요소들 간의 의존성 검사를 위한 다양한 메카니즘을 제공하도록 하며 특히, 모델 조립(composition) 프로세싱 동안에 모델들 간의 의존성 검사를 자동화한다.[그림 10]



[그림 10] MDA 모델통합에 의한 프로토타이핑

[Fig. 10] Prototyping with MDA Model Convergences

4. 융합 모델 기반 교수-학습 시스템 모델링 사례

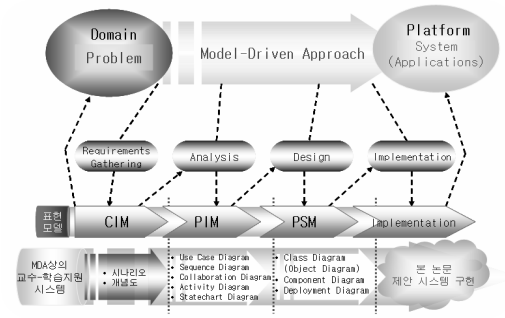
4.1 교수-학습지원 시스템 구성

본 논문에서 연구하는 시스템 구성은 사용자 인터페이스를 통한 웹 서비스와 웹 어플리케이션을 이용하여 교육 컨텐츠를 신규로 등록하고 이용할 수 있는 클라이언트-서버 시스템이다. 학습자들에 대한 개개인의 특성에 맞는 학습컨텐츠를 선택하여 수준별 교육을 위한 학습 활동과 강의형태에 대한 시스템 구조를 전형적인 3-tier로 표현한다. 우선 학습자와 교수자를 위한 인터페이스를 제공하는 사용자 인터페이스와 학습활동을 지원하기 위한 정보들을 관리할 수 있는 웹 서비스 부분이 핵심구성요소이다. 웹 서비스 표준을 기반으로 해서 학습컨텐츠 저장소에 접근하여 교수-학습지원 시스템의 e-Learning 활동을 수행하게 된다.

4.2 교수-학습지원 시스템의 개발 프로세스 아키텍처

교육 도메인 영역에서 제시되는 문제에 대해 요구되어지는 플랫폼 환경에 맞는 교수-학습지원 시스템을 개발하기 위해 모델지향적 접근 즉, MDA의 개발 프로세스를 통해 어플리케이션을 개발하고자 한다. 표현모델의 각 단계별 Asset을 CIM단계에서 요구사항을 수집하여 시나리오와 시스템에 대한 개념도로 표현한다. PIM단계에서는 유즈케이스 다이어그램, 시퀀스 다이어그램, 협력 다이어그램, 활동 다이어그램, 상태 다이어그램과 같이 5개의 동적 다이어그램으로 표현하고 PSM 단계에서는 클래스 다이어그램, 컴포넌트 다이어그램, 배치 다이어그램과 같이 3개의 정적 다이어그램으로 표현한다. 그리고 PSM단계의 표현 모델을 맵핑하여 본 논문에서 제안하는 시스템을 구현하게 된다.

이와 같은 제안 시스템의 개발 프로세스 아키텍처를 [그림 11]에서 나타낸다.



[그림 11] 교수-학습지원 시스템 개발 프로세스 아키텍처

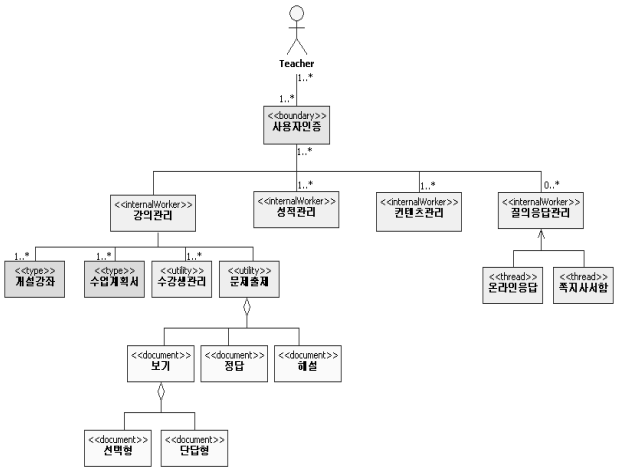
[Fig. 11] Process Architures for Teaching-Learnig System Development

4.3 교수-학습지원 시스템의 CIM단계 표현모델 Asset

학습자 개개인의 특성에 맞는 교육을 위한 학습 활동에 대한 구조를 가지며 학습자와 교수자를 위한 인터페이스를 제공하는 사용자 인터페이스와 학습활동을 지원하기 위한 정보들을 관리할 수 있는 시스템의 필요성이 대두된다. 제안 시스템은 관점에 따라 3가지로 <표 1>과 같이 교수자 관점의 시스템 시나리오를 나타낸다.

[표 1] 교수자 관점의 시스템 요구사항 시나리오
[Table 1] Requirement Scenarios

교수자 관점	사용자인증을 통해 교수자 영역으로 구분된다 강좌개설, 수업계획서입력, 성적입력을 한다 강좌 수강학생명단을 요청한다 개인정보를 입력한다 쪽지사서함을 이용한다 개설강좌의 컨텐츠를 검색한다 온라인 문제를 출제한다 강의접속률을 조회한다 학습자에게 질의응답한다



[그림 12] 교수자 관점의 교수-학습지원시스템 개념도
[Fig. 12] System Concepts for Teaching side

컴포넌트를 개발하기 위한 기술적인 시작단계로써 도메인 분석은 개발하고자 하는 영역에 대한 이해를 목적으로 한다. 또한 개발하려는 시스템이 어느 영역에 포함하는지, 어떠한 개발과정으로 전반적인 분석과정의 단계로 이루어지는지에 대한 시스템 개념도를 작성한다.

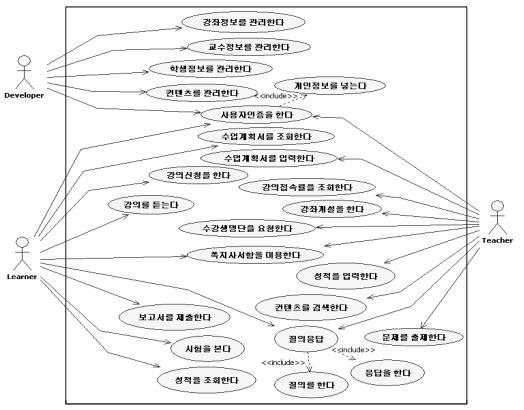
교수-학습지원 시스템에서 학습자의 학습활동과 교수자의 교수활동에 있어서 교수자 관점의 교수-학습지원 시스템 개념모델을 [그림 12]에서 나타내는데 교수 활동에 대한 6가지로 모듈화

(modularity)한 스테레오 타입을 나타낸다.

4.4 교수-학습지원 시스템의 PIM단계 표현모델 Asset

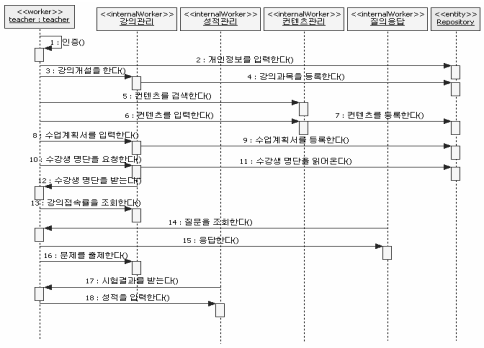
학습자, 교수자, 개발자 측면을 고려하여 시스템을 설계하며 [그림 13]에서 교수활동에서의 요구사항 그리고 학습활동에서의 요구사항 즉, 액터들의 행위를 유즈케이스 다이어그램으로 나타낸다. 우선 사용자 인증으로 교수자 영역으로 인증을 받고 교수활동의 핵심이라 할 수 있는 강의관련 컨텐츠를 검색하고 학습자들의 성적을 관리하며 온라인 강의에 대한 질의응답에 관한 행위를 수행한다.

이러한 유즈케이스 실현은 교수-학습지원 시스템의 교수자 관점의 인터랙션 다이어그램인 시퀀스 다이어그램과 협력 다이어그램으로 표현한다. 교수활동에 대한 시간적 메시지 흐름인 시퀀스 다이어그램을 [그림 14]에서 나타낸다.



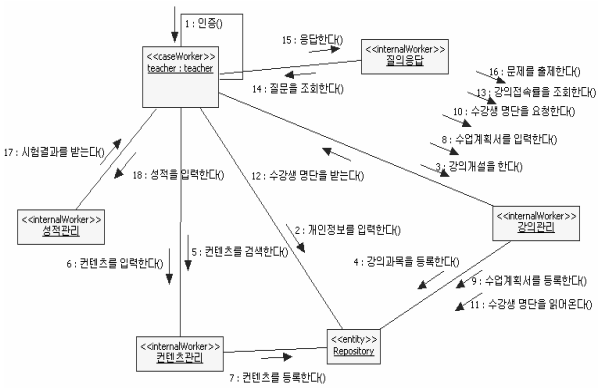
[그림 13] 교수활동과 학습활동에 대한 유즈케이스 다이어그램

[Fig. 13] Use case diagram



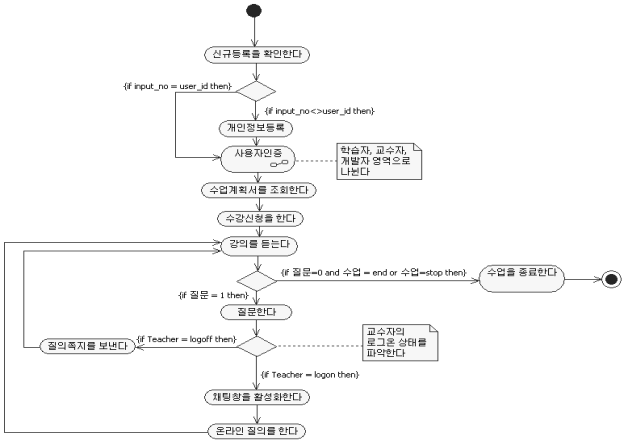
[그림 14] 교수-학습지원 시스템의 교수자 관점 시퀀스 다이어그램

[Fig. 14] Sequence Diagram for Teaching-Learning



[그림 15] 교수-학습지원 시스템의 교수활동에 대한 협력 다이어그램

[Fig. 15] Collaboration Diagram for Teaching-Learning



[그림 16] 학습자관점의 Lifecycle에 대한 액티비티 다이어그램

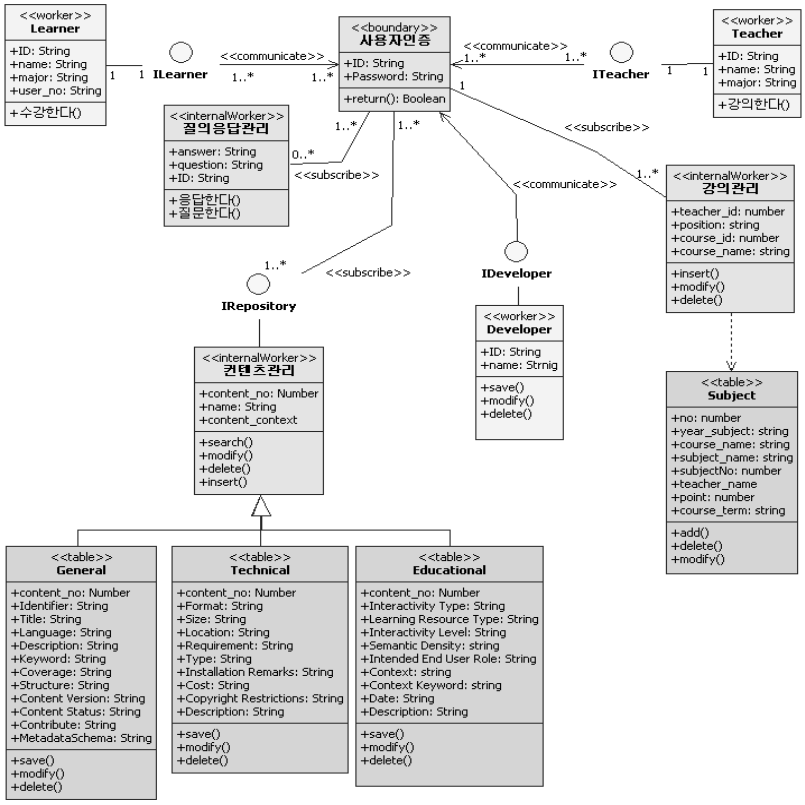
[Fig. 16] Acivation Diagram of Life Cycle for Leaning Side

교수활동에 대한 시간적 메시지 흐름인 시퀀스 다이어그램을 다시 활동적 메시지 흐름에 대한 협력 다이어그램으로 변환하여 [그림 15]에서 나타낸다.

학습자관점의 활동 흐름에 대한 명세를 [그림 16]에서 액티비티 다이어그램 또는 활동 다이어그램으로 나타낸다. 학습자 관점에서 질의응답은 강의를 수강중에 질문 사항이 유발될 경우 교수자의 로그온 혹은 로그오프 상태에 따라 로그온의 경우는 일반 채팅과 같은 기능으로 학습자의 질문에 즉각적인 대응으로 응답이 가능하고 로그오프의 경우는 학생의 질문사항을 쪽지로 보관하여 교수자에게 전달하게 되는 학습자 관점의 라이프 사이클에 대한 흐름을 나타낸다.

4.5 교수-학습지원 시스템의 PSM단계 표현모델 Asset

교수자와 학습자 각각의 활동을 위한 컨텐츠에 대해 관리와 활용측면에서 교수자가 새로운 학습자료를 관리하고자 할 때 국제표준기구에서 제안한 SCORM의 Content Aggregation Model을 이용하여 컨텐츠 메타데이터를 등록하고 수정, 삭제할 수 있는 관리를 위한 구조를 [그림 17]에서 제안 시스템의 PSM단계의 클래스 다이어그램으로 나타낸다.



[그림 17] 교수-학습지원 시스템에 관한 클래스 다이어그램

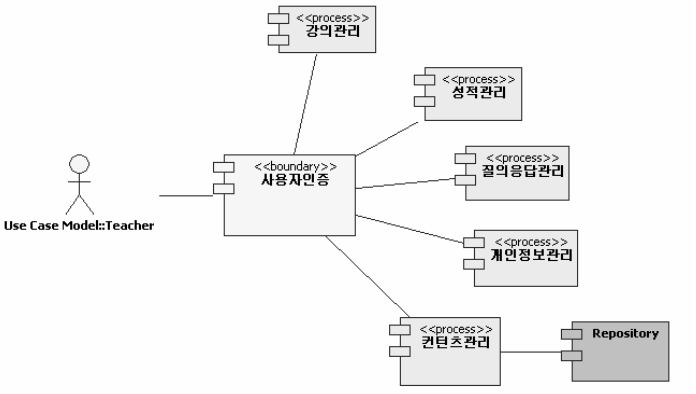
[Fig 17] Class Diagram for Teaching-Learning Systems

학습객체가 가지는 속성과 구조, 연결성 등의 기본적인 사항을 포함하고 있는 메타데이타는 응용 시스템에서의 저장, 검색, 관리, 운영, 유지보수등에 결정적인 역할을 수행함으로 정보의 활용성을 높인다. 컨텐츠 관리에 있어서 교육 컨텐츠의 일반적인 정보에서 지적소유권과 생명주기까지 살펴볼 수 있는 내용을 담고 있는 SCORM의 9개 카테고리를 일반적 컨텐츠 자원에 대한 정보를 설명하는 기본 정보와 기술적 요구사항, 학습자원에 대한 기술성 부분 그리고 자원이 갖고 있는 교육적 특징과 교수법상의 특징들에 대한 교육성과 같이 3개의 카테고리로 재정의하여 <표 1>에서 교수-학습지원 시스템의 컨텐츠 메타데이타 재정의 목록을 나타낸다.

[표 2] 교수-학습지원시스템의 컨텐츠 메타데이터 정의

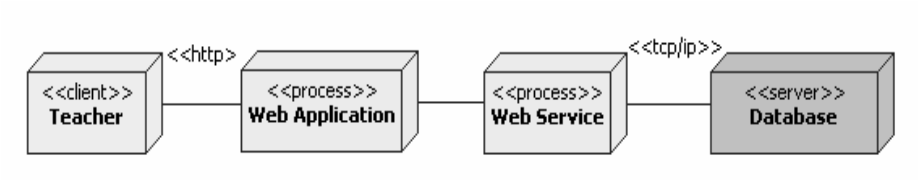
[Table 2] Contents Meta Data Definition for for Teaching-Learning

정보 모델	컨텐츠 메타 데이터 정보		설명
	개략	세부항목	
LOM	기본정보 및 컨텐츠 메타데이터	분류코드 이름 개발언어 분류영역 키워드 적용범위(영상,음향 등) 구성사항 컨텐츠버전 컨텐츠등록일자 제공처 메타데이타스키마	일반적으로 자원에 대한 정보를 설명하며 자원에 대한 상태에 대해 특정 정보를 기술한다.
	기술성	컨텐츠형태 컨텐츠크기 기술영역 요구사항 설치장비 플랫폼요구사항 개발기간 개발비용 지적소유권 기술설명	기술적 요구사항과 학습자원에 대한 특징, 지적소유권 및 사용 조건에 대한 정보를 기술한다
	교육성 및 활용안내	교육활동타입 교육자원타입 교육활동레벨 교육적비중도 학습자영향도 교육활동배경 키워드 교육적본질 관련내용 관련내용설명	자원이 갖고 있는 교육적 특징과 교수법상의 특징들에 대한 정보를 갖으며 자원이 검색가능한 키워드에 대한 정보 포함한다



[그림 18] 교수자 관점의 컴포넌트 다이어그램

[Fig. 18] Componnets Diagram for Teaching Side

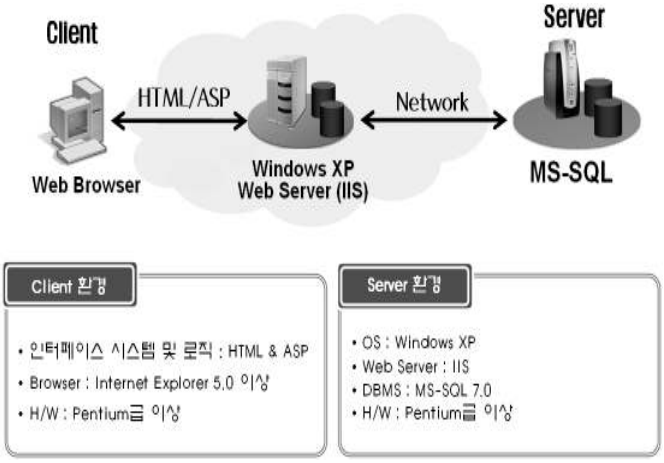


[그림 19] 교수자 관점의 제안 시스템에 관한 배치 다이어그램
[Fig. 19] Batch Diagram for Teaching Side

CIM단계의 개념 다이어그램에서 모듈화(modularity)한 부분을 컴포넌트 다이어그램으로 [그림 18]에서 나타내며 [그림 19]에서는 클라이언트와 서버 그리고 중간매개역할을 수행하는 비즈니스 로직을 배치 다이어그램으로 나타낸다.

5. 교수-학습지원 시스템 구현 환경

본 논문의 제안 시스템 개발을 위한 환경은 [그림 20]에서와 같이 교수-학습지원 시스템 전체 구현 구조도를 나타내고 있다. 서버 시스템으로 Windows XP를 운영체제로 사용하고 IIS 6.1상에서 ASP를 통하여 작성되었으며 데이터베이스는 MS-SQL Server를 사용하였다.



[그림 20] 교수-학습지원 시스템 개발 환경
[Fig. 20] Development Environment For for Teaching-Learning

6. 결론

모바일 임베디드 시스템은 물리적 외부 환경과의 상호작용을 제어하기 위해, 물리적인 규칙들을 시스템상에서 구체화하고, 실시간적으로 실세계 환경의 동적인 복잡성을 제어해야만 할뿐 아니라,

시간적 제약 요소를 많이 포함하고 있다. 모바일 임베디드 응용 소프트웨어 개발 생산성 및 품질 향상을 위해 도메인 요구사항을 최대한 만족하는 방법론과 지원도구 제안이 필요하다. 이를 위해, 실행환경 독립된 핵심 기술요소인 모델링 방법론, MoES(MObile Embedded Software) 아키텍처, MoES 통합 및 조립 기술, 모델링 기술에 대한 기반 연구는 기존의 소프트웨어 개발 방법론을 기반 하여 구체화하며, 모델 자산의 재사용 기법은 본 연구실에서 개발한 CRMS(Component Repository Management System)을 기반으로 구체화한다. 실행서비스 지원 도구는 도메인 응용 모델(코드 라이브러리), 미들웨어 라이브러리, OS 모델, aspects, 개발 라이브러리로 생산된 제품(새로운 제품 라이브러리) 참조를 통해 프로토타이핑 시스템 구축으로 구체화시킨다. 본 연구에서는 융합 모델을 통해 MoES 모델 및 아키텍처를 작성하고, 이를 기반으로 MoES 영역의 효율적인 서비스를 제공하기 위한 전체 시스템 환경을 구축하게 되며, 궁극적으로 사용자의 요구에 맞는 MoES를 기존의 자산과 모델의 조립을 고려한 프로세스의 정의 및 이를 관리하기 위한 전반적인 방법론에 대해 연구한다. 또한, 이를 지원하고 상호운용 할 수 있는 다양한 모바일 플랫폼 영역에 지능적인 서비스를 제공함으로써 개발에서 효율적이고 지속적인 형상 변환이 용이하며 높은 품질의 임베디드 S/W 제품 생산을 가능하도록 구체화한다.

참고문헌

[1] Fei Xie and Guowu Yang, etc. "Component-based hardware/software co-verification for building trustworthy embedded systems.Journal of Systems and Software", Vol.80, No.5, pp.643-654, 2007

[2] 조은숙, 김철진 외 1명, “임베디드 시스템의 재사용 프레임워크를 위한 정적 메타모델 설계”, 멀티미디어학회논문지, Vol.12, No.2, pp.231-243, 2009

[3] Gabor Karsai andSandeep Neema. "Model-driven architecture for embedded software: A synopsis and an example". Science of Computer Programming, Vol.73, No.1, pp. 26-38, 2008

[4] 황위용, 강동수 외 3명, “의존과 관점 기반 임베디드 시스템의 요구사항 우선순위 프로세스”, 정보처리학회논문지, Vol.16, No.5, pp.767-790, 2009

[5] 손현승, 김우열 외 1명, “이종 임베디드 시스템의 멀티태스킹을 위한 MDA(Model Driven Architecture) 기반의 설계”, 정보처리학회논문지, Vol.15, No.3, pp.355-360, 2008

[6] François Coallier and Roger Champagne. "A Product Line engineering practices model". Science of Computer Programming, Vol.57, No.1, pp.73-87, 2005

[7] Antonio Coronato and Giuseppe De Pietro. "Formal design and implementation of constraints in software components". Advances in Engineering Software, Vol.41, No.5, pp.737-747, 2010

[8] Object Management Group. Systems Engineering Domain Special Interest Group . http://syseng.omg.org.

[9] Krishnakumar Balasubramanian and Jaiganesh Balasubramanian, etc. "A Platform-Independent Component Modeling Language for Distributed Real-time and Embedded Systems". Journal of Computer and System Sciences, Vol.73, No.2, pp.171-185, 2007

[10] Fritz Solms and Dawid Loubser, "Generating MDA's platform independent model using URDAD". Knowledge-Based Systems, Vol.22, No.3, pp.174-185, 2009

[11] Shuichi Shimizu and Raju Rangaswami, etc. "Platform-independent modeling and prediction of application resource usage characteristics". Journal of Systems and Software, Vol.82, No.12, pp.2117-2127, 2009

[12] 이재준 , 김경석 외 2명, "특집 : 소프트웨어 재사용 ; 프로덕트 라인 엔지니어링에서 동적 재구성이 가능한 임베디드 시스템 개발을 위한 휘처 중심의 방법", 정보처리학회지, Vol.13, No.6, pp.62- 77, 2006

[13] Teade Punter and René L. Krikhaar, etc. "Software engineering technology innovation − Turning research results into industrial success". Journal of Systems and Software, Vol.82, No.6, pp.993-1003, 2009

[14] 유태권, 라현정 외 1명, "컴포넌트 기반의 MDA 공학 프로세스", 한국정보과학회 학술발표논문집, pp. 325~327, 2005

[15] N.C.W.M. Braspenning and J.M. van de Mortel-Fronczak, etc. "A Model-based Integration and Testing Method to Reduce System Development Effort". Electronic Notes in Theoretical Computer Science, Vol.164, No.4, pp.13-28, 2006

[16] Shi-Ming Huang and Chih-Fong Tsai, etc. "Component-based software version management based on a Component-Interface Dependency Matrix". Journal of Systems and Software, Vol.82, No.3, pp.382-399, 2009

저자 소개

김행곤 (Haeng Kon Kim)

1985년 중앙대학교 전자계산학과(공학사)
1987년 중앙대학교 대학원 전자계산학과(공학석사)
1991년 중앙대학교 대학원 전자계산학과(공학박사)
1978년~1979년 미 항공우주국 객원 연구원
1987년~1989년 한국전기통신공사 전임연구원
1988년~1989년 AT&T 객원 연구원
2001년~2002년 Central Michigan University 교환교수
2007년~2009년 미 ACIS 연구교수
1990년~현재 대구가톨릭대학교 컴퓨터공학과 교수
관심분야 : Mobile Embedded 소프트웨어 공학, 융합 모델링, 사용자 인터페이스, 요구공학 및 도메인 공학