임베디드 시스템 개발방법론 및 재사용 체계

Development Methodology and Reuse Supporting System for Embedded System

임베디드 S/W 기술 동향 및 연구 개발 현황

양영종 (Y.J. Yang) S/W공학연구팀 팀장
조진희 (J.H. Cho) S/W공학연구팀 선임연구원
하수정 (S.J. Ha) S/W공학연구팀 선임연구원
차정은 (J.E. Cha) S/W공학연구팀 선임연구원

목 차

특정 산업용 기기의 제어를 위해 사용되던 임베디드 시스템이 유무선 통신 네트워크와 접목으로 디지털 정보가전, 의료, 항공, 군사 등 전 산업 분야로 확대되는 “Embedded, Everywhere” 시대가 도래하고 있다. 임베디드 시스템은 실시간 처리, 저전력 등의 물리적 특성과 하드웨어(HW)와 소프트웨어(SW)의 동시 설계, 리소스의 절제된 사용 등의 특성을 반영해야 하므로 시스템 개발 전 과정에서 이러한 특성을 만족시키는 개발 체계의 구축이 필요하다. 특히, 임베디드 소프트웨어의 공통·핵심 기술을 자산화하여 체계적으로 재사용할 수 있는 환경 구축은 기술의 중복 개발을 최소화하고, 기술의 가치를 지속적으로 증대시킨다. 따라서, 고품질의 임베디드 시스템을 적시에 경제적으로 개발할 수 있는 임베디드 시스템 개발방법론과 임베디드 소프트웨어 재사용 체계의 개발 및 보급 기술은 소프트웨어 산업 경쟁력 향상에 공통적으로 필요한 기반 기술로 활용될 수 있다. 본 고에서는 신 성장 미래 산업의 기반이 되는 임베디드 시스템을 위한 개발 방법론과 재사용 체계 구축에 대한 동향을 기술한다.

I. 임베디드 시스템 개발 동향

1. 서언

시스템 개발 방법은 과거의 무방법론에서 시작하여 요구사항 수준이 상승됨에 따라 시스템 전략의 요구 사항을 계획에서부터 분석하고 설계하여 구현으로 연계하는 체계적인 방법이 필요하게 되었다. 방법론은 시스템의 정의, 개발, 유지보수까지의 전 활동에 적용될 수 있는 개발 절차와 방법 및 기법, 효율적인 사업 진행을 위한 관리 활동, 이러한 제반 요소들을 지원할 수 있는 환경들을 총칭하며 생명 주기에 따라 시스템을 개발하기 위한 이론적인 기반이 되어 왔다. 방법론은 그 자체가 직접적인 기술이라기 보다는 소프트웨어 기술들을 표준화하고 이해함으로써 개발 프로젝트의 생산성을 높이는 중요한 수단이며, 소프트웨어 개발 패러다임의 변화에 따른 여러 가지 유형의 개발방법론들이 출현하고 있다.

한편, 소프트웨어 생산성과 품질 개선의 근본 해결책으로 소프트웨어 재사용이 끊임없이 제안되고 있다. 실제, Nokia나 IBM, NASA 등에서는 성공적인 재사용으로 소프트웨어 생산 비용의 50% 이상을 절감한 사례를 보고하였다. 특히, 다양한 디지털 제품에 내장되어 생산에서 소비에 걸쳐 여러 계층의 IT 수요를 만족시킬 수 있는 임베디드 소프트웨어를 재사용 가능한 형태로 공급하고 이를 9대 신성장 동력 관련 기술 개발에 적용할 수 있는 체계적 환경을 구성한다면 임베디드 시스템의 고부가가치를 생성, 확대, 유지시킬 수 있다.

본 고에서는 IT 839 등을 비롯한 신 성장 미래 산업의 기반이 되는 임베디드 시스템을 위한 개발 방법론과 재사용 체계 구축에 대한 동향을 기술한다.

2. 개발 방법론의 변화 기로

소프트웨어 기술 발전에 따라 정보시스템 위주로 적용되어 왔던 구조적 방법론이나 정보공학 방법론, 객체지향 방법론, 컴포넌트 공학 방법론은 임베디드 시스템의 특징에 부합하기 어렵다. 각종 전자기기, 가전제품, 제어장치의 임베디드 시스템들은 단순히 회로뿐만 아니라 특정한 기능을 수행하도록 시스템 소프트웨어가 내장되어 있다. 그러나, 시스템 소프트웨어와 하드웨어의 다양한 제어와 기능을 개발하는 개발 인력들은 전통적인 소프트웨어 개발 방법론을 임베디드 시스템이라는 새로운 패러다임에 적용하는 데 있어 개발 환경의 차이와 개발 제약사항 등을 해결하기 위한 새로운 개발 체계가 필요한 상태이다. 또한, 기존의 정보 시스템을 고려한 개발방법론은 임베디드 시스템이 갖고 있는 특정 제한요인들을 위한 개발 기술이 반영되어 있지 않다. 임베디드 시스템이 일반적으로 개발되어 온 정보 시스템과 어떤 다른 특징을 갖는지 살펴 보자.

3. 임베디드 시스템의 특징

임베디드 시스템은 매우 다양하여 특징들을 모두 언급하기 어렵지만 공통적인 특징을 요약하면 다음과 같다.

1) 목적이 정해져 있다.

범용적으로 만들어진 프로세서와 메모리의 경우에는 어떤 애플리케이션이 수행될지 정확히 알지 못하므로 일반적인 모든 프로그램이 빠르게 실행될 수 있도록 구조화된다. 프로세서의 성능은 운영체제, 애플리케이션, 컴파일러 등 많은 각각의 요소에 의해 향상될 수 있다. 그러나 임베디드 시스템은 특정 목적을 위해 최소의 하드웨어와 최소의 소프트웨어만으로 고정된 기능을 수행하기 때문에 단지 소프트웨어만으로 성능 향상을 기대할 수 없다. 따라서 개발 초기 단계에서 하드웨어와 소프트웨어가 함께 구성되는 임베디드 시스템을 설계하고 하드웨어로 구현이 되어야 할지 또는 소프트웨어로 처리되어야 할지에 대해 최적화되어 기능이 분할되고 특화된 소프트웨어와 하드웨어가 구현됨으로써 주어진 목적을 효과적으로 처리할 수 있도록 고려한다.

2) 실시간 처리가 대부분이다.

일반인이 하루에 접하게 되는 임베디드 시스템의 수는 최소 20개에 이른다고 한다. 생활 속에서 발견되는 임베디드 시스템은 냉장고, 세탁기, 휴대 전화기, MP3 재생기, 카메라, 자동차에 이르기까지 다양하다. 이들 임베디드 제품들은 특정 목적을 위해 주어진 자원을 최대한 이용하여 일정한 처리 기한 안에 수행되는 것을 목적으로 한다. 전기적이거나 기계적인 동작을 제어할 수 있도록 정해진 시간 안에 동작하도록 개발되는 경우가 대부분이므로 처리해야 할 작업에 대해 제한 시간 안에 처리를 마칠 수 있는 작업 간의 조정 메커니즘이 필수적이다.

3) 제품군으로 생산된다.

임베디드 시스템은 같은 제품을 다양한 모델로 대량 생산하는 형태를 띠고 있다. 구체적으로 MP3 재생기, 휴대용 전화기 등은 유사한 기능을 공통적으로 갖고 차별화된 다양한 서비스를 갖는 여러 개의 제품군으로 제품의 대량 생산이 이뤄지고 있다. 전자, 의료, 군사 등의 분야로 확장되어 가고 있는 임베디드 시스템은 제품군으로 생산될 때 시장에 대한 경쟁력과 제품의 품질에 대해 고려하여 개발되어야 한다. 임베디드 제품의 시장 경쟁은 계속 높아지고 있고 개발자들의 개발 기간에 대한 압박은 점점 커지는 실정이다.

4. 개발방법론의 해외 동향

해외에서 수행된 임베디드 시스템 개발방법론과 통합 개발 환경의 개발사례를 보면 요구 사항 분석, 소프트웨어와 하드웨어의 통합 설계, 컴포넌트화된 방법론 구성, 비기능적 속성의 향상 등에 중점을 두어 복잡한 시스템을 시각화하고 개발하는 방법에 초점을 맞추고 있다. 미국과 유럽은 산업계, 학계, 연구소가 함께 연계된 임베디드 개발방법론 개발 프로젝트를 진행해 오고 있다.

대표적인 개발방법론으로는 미국 GIT에서 수행된 임베디드 소프트웨어 제품계열 체계를 구축하기 위한 Yamacraw, 버클리 대학의 MoBIES, Nokia와 필리스를 비롯한 유럽 3개국에서 추진된 MOOSE, BMW를 비롯한 유럽 3개국에서 수행된 COMITY, INRIA의 DESS가 있다.

5. Nokia의 성공 사례

정보통신분야에서 제품계열 기반의 개발 패러다임을 선구적으로 적용하여 모범이 되고 있는 곳은 Nokia이다. Nokia는 모바일 브라우저 소프트웨어 개발에 제품계열 기반 방법을 적용하여 휴대폰의 공통 모듈과 가변 모듈을 생성하고 특정 제품계열에 공통으로 사용될 소프트웨어 핵심 자산을 미리 구축하고 재사용하여 개별 제품을 신속히 개발하는 개발방법을 적용하고 있다. Nokia의 핸드폰 플랫폼과 Nokia 모바일 인터넷 툴킷들이 제품계열 기반 방법에 기반하여 생산되어 오고 있다. 휴대폰 업계 세계 1위인 Nokia는 타기업들과의 경쟁에서 우위를 점유하고 있는데 제품계열 기술을 통해 소프트웨어 개발 생산성이 증대되어 매년 4개의 제품군에 대해 6번의 제품 출시가 가능함을 보여 주고 있다[1].

6. 국내 관련 동향

임베디드 시스템의 복잡도가 증가하고 이들 제품에 포함된 소프트웨어와 하드웨어의 크기 및 다양성이 크게 증가하는 상황에서 많은 국내 임베디드 시스템 개발자들은 소비자가 원하는 기능을 적시에 구현하는 데 큰 어려움을 겪고 있다. 이러한 어려움을 극복하기 위한 노력으로 중견 및 일부 대기업에서는 근래에 와서 부분적으로 임베디드 시스템 개발 방법론의 개발이나 도입을 시도하고 있는 상황이다.

본 연구팀에서 실시한 설문의 결과, 실무자들이 임베디드 시스템 개발 시 해결하기 어렵다고 느끼는 사항에 대해 그 중 대표적인 결과를 중심으로 요약하면 다음과 같다.

• 요구 사항 분석 시

요구 분석 차이의 해결방법과 명확한 요구 사항

을 전달할 수 있는 표준화된 요구 사항 정의 및 분석 방법이 필요하다고 답변했다.

• 설계 시

설계 경험 부족으로 고려되지 않은 설계 항목이 발생하는 경우에 대한 대처와 실시간 설계 모델의 표현이 어렵다는 의견이 높았다. 아키텍처 설계와 관련하여 재사용 가능한 하드웨어와 소프트웨어의 설계가 필요하나 실질적으로는 미흡하였다. 그리고, 요구 사항이 변경되었을 때 대처할 아키텍처를 잡는 것이 쉽지 않다고 하여 개발 인력들이 시스템 개발에 있어 재사용성에 대한 필요를 인식하고 있음을 확인할 수 있었고 아키텍처를 효과적으로 설계를 하는 데 어려움을 겪고 있음이 파악되었다.

• 구현과 시험 시

적은 리소스로 기능을 구현하는 데 어려움이 있어 제한적으로 프로그래밍이 되어야 한다고 답변하였다. 그리고, 복잡한 개발 환경으로 시험에 어려움을 겪고 있고 버그를 재현하고 오류 발생 부분을 구분하는 데 어려움이 있다고 답변하였다.

7. 소프트웨어 재사용 관련 연구

소프트웨어 개발의 낮은 생산성과 품질의 문제를 혁신적으로 개선(silver bullet)시켜줄 핵심 기술로 소프트웨어 재사용이 1970년대 이후 지속적으로 제시되고 있다. 그러나, 소프트웨어 재사용은 매우 어려운 작업으로, 특히 정보시스템에서 주요 재사용 대상인 컴포넌트와 같이 범용적이며 큰 규모일 경우는 개별 라이브러리 모듈 개발에 비해 3배의 비용이 요구될 뿐 아니라, 3가지 이상의 다른 응용에 성공적인 적용을 증명해야만 재사용의 신뢰성과 가치를 인정 받을 수 있기 때문에 개발 비용과 위험성이 상대적으로 높다[2].

그럼에도 불구하고, 핀란드 Nokia의 재사용 성공 사례는 임베디드 시스템 개발의 생산성과 품질의 향상에 소프트웨어 재사용의 중요성을 크게 각인시켜 준다. 또한 Motorola, IBM 등 세계적 기업들뿐만 아니라 유럽의 EU ITEA, 미 항공우주국, ETRI 등 국가 주도의 연구 기관에서도 유사 영역에 공통적인 재사용 자산을 개발하고 활용을 지원할 수 있는 기술과 자동화 도구를 개발하는 과제를 추진하였다.

〈표 1〉 임베디드 소프트웨어 재사용 기술 개발 사례

과제 명	수행기관	특징	비고
GTE Data Service AMP (Asset Management Program)	NASA GTE (1986~1994)	- 재사용 자산의 조직적 생성, 조작 - 자산 라이브러리 유지보수 시스템 구축	연 14%의 자산 재사용으로 1500만 달러 절감
Switching 시스템의 BB (Building Block)	Kommunikations Industrie, 독일 (1992~1994)	- 스위칭 시스템에 재사용 컴포넌트 사용 - 통신 서비스 스위치, 모바일 디지털 스위치 등에 재사용	최소 40% 이상의 재사용률 달성이 가능
SPIQ(S/W Process Improvement for better Quality)	노르웨이 오슬로 대학 (1997~1998)	- 인트라넷을 통해 조직의 S/W 개발자들이 공유할 수 있는 정보의 관리 및 운영 - 프로세스, 규칙, 개발 경험의 모든 산출물들에 대한 관리	TTS(Telenor Telecom S/W) 개발에 적용하여 2~3년간 비용절감
SPHERES (Synchronous Position Hold Engage Reorient Experimental Satellites)	MIT, NASA (2002~2004)	- 아주 미세한 위험 관리 및 자동 제어 기술을 위한 재사용 명세 지원 및 도구 개발 - 인공위성 및 로케트 락킹 작업과 같은 임베디드 시스템 개발 및 유지보수에 적용	우주선에 필요한 임베디드 S/W 개발에서 70% 재사용률 달성
RCM (Reuse Cost Model)	Israel IEEE Members (2000~2003)	- 재사용 자산에 대한 재사용 시나리오를 도출 및 비교를 통한 재사용 비용의 평가 - 재사용 시나리오에 따른 재사용 자산의 재사용을 위한 프로세스 및 도구 개발	TES(Tadian Electronic System Ltd.)에서 28~63%의 재사용률 달성
Naval Product Line as the Ship System 2000	CelsiusTech Systems (1986~현재)	- 다양한 군함 생산에 제품계열 기법 도입 - 명령 및 통제, 통신 시스템을 하나로 통합하는 제품계열 환경 제공	공통 자산의 70~80%의 재사용으로 생산 기간 감소

<표 1>은 임베디드 소프트웨어 재사용 기술과 관련된 성공적인 프로젝트들의 예이다.

그러나, 이들 연구들은 특정 영역에서 동일 조직만을 위해 구축된 특별한 재사용 환경으로, 재사용 대상이 되는 재사용 자산에 대한 정확한 정의나 재사용 자산을 개발하기 위한 구체적인 절차나 기법이 매우 미흡하다. 또한, 재사용 자산의 공급자와 사용하는 이해당사자들 간에 재사용 자산의 개발 단계별 자산 형태에 대한 표준화된 동의가 존재하지 않아 재사용에 의한 효과가 극히 미흡하여 재사용은 오직 이론적인 개념으로만 인식하게 하였다. 특히 오늘날의 소프트웨어 산업에서 가치가 급속히 증가하고 있는 임베디드 소프트웨어를 위한 재사용 체계는 전무한 실정이다.

이에 본 연구팀에서는 임베디드 소프트웨어의 개발 생산성을 높이고 동일 조직에서 만들어진 소프트웨어 자산의 가치를 지속적으로 극대화시킬 수 있는 임베디드 소프트웨어의 재사용 체계를 구축하고 있으며, 이에 대한 연구 내용을 본 고에서 개괄적으로 설명한다.

Ⅱ. 개발방법론 EMMA

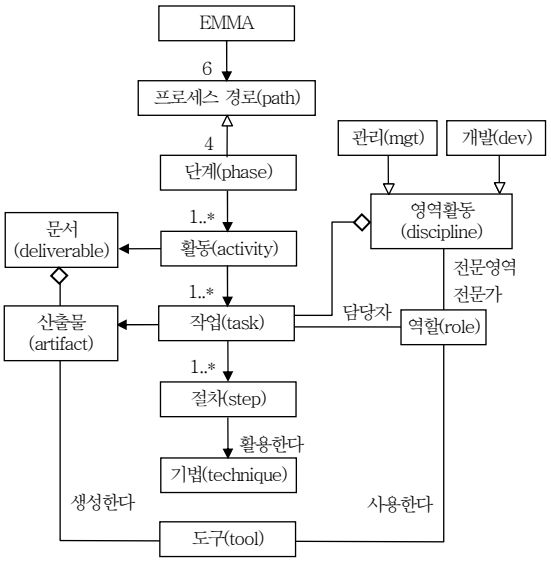
1. EMMA 개요

SW공학연구팀에서 현재 개발하고 있는 임베디드 시스템 개발방법론인 EMMA는 1994년부터 개발되어 온 한국형 개발방법론인 마르미(Magic and Robust Methodology Integrated) 시리즈 중 임베디드 시스템의 개발을 목적으로 하는 버전이다. EMMA는 제품계열(product line)[3] 기술에 기반하여 일련의 유사한 제품들을 동시에 개발하는 개발 프레임워크이다. 즉, 전략적으로 제품개념 정의 단계에서부터 유사 제품군에서 공통적으로 재사용될 모듈을 사전에 분석하고 공통 아키텍처를 설계하여 핵심 자산의 재사용을 극대화 함으로써 제품의 시장 출시 시점에는 제품들을 신속하게 개발하고 출시하는 데 초점을 맞춘 개발방법론이다. EMMA는 또한 최근 소프트웨어공학 분야에서 국제적인 사실표준(de-facto standard)을 주도하는 OMG의 소프트웨어 개발 프로세스 메타모델인 SPEM[4]을 기반으로 EMMA의 전체 구성 요소와 그 관계를 구성하였다. 이를 통해 현재 SPEM 기반으로 변경하고 있는 타 방법론(예: RUP, DMR Macroscope, IBM GS Method, Fujitsu SDEM 등)과 호환될 수 있는 발판을 마련하였다. EMMA는 OMG의 산업계 표준인 UML 2.0[5]을 모델링 표준 표기법으로 채택하고 있으며, 방법론 자체도 UML로 모델링하여 활동간의 연계성뿐만 아니라 그 동안 개발방법론 완성도의 척도로 제기되었던 산출물들간의 유기적 연계성 및 상태 추적성을 확보하여 개발방법론으로서의 완성도를 높였다. 이로써 개발방법론 모델을 하나의 완성된 리파지토리 형태로 구성할 수 있게 되었고, 개발방법론 구성 자체의 일관성(consistency)을 유지하면서도 실제 프로젝트의 특성이나 상황에 맞게 매우 유연하게 조정하는 것이 가능해졌다.

EMMA는 프로젝트 수행에 필요한 역할을 중심으로 해당 역할 수행자들이 프로젝트의 진행 일정에 따라 책임지고 수행해야 할 전문영역활동(discipline)별로 지침서를 제공한다. 또한 6개의 개발 프로세스 경로를 정의하고 있는데, 이들 경로는 프로젝트 관리자가 프로젝트의 특성에 맞게 WBS를 정의하기 용이하도록 작업흐름(work flow)을 제시한다. 그리고 프로젝트 참여자들이 수행하는 활동의 기본은 작업이며, 작업에서 각 역할은 지원도구를 이용하여 특정 산출물을 만들어 낸다. 또한 작업에는 그것을 위한 세부 절차들이 정의된다. 이것을 나타내는 개략적인 개념 모델은 (그림 1)과 같다.

2. EMMA 프로세스

EMMA 프로세스는 임베디드 시스템의 개발 측면에서 전문화된 프로세스와 프로세스의 관련 단위별로 묶인 영역활동을 제공하며, (그림 1)과 같이 개발 프로세스와 관리 프로세스로 구성된다. 개발 프로세스는 임베디드 시스템의 도메인 분석, 아키텍처, 설계, 소프트웨어와 하드웨어 구현, 검증, 배치



(그림 1) EMMA의 개념적 메타모형

의 활동들로 구성되고 자산(asset)과 제품(product)을 개발하기 위한 활동, 절차, 지침, 산출물, 역할에 대해 표준화된 일련의 절차들을 제공한다.

EMMA의 관리 프로세스는 프로젝트, 품질, 형상에 대한 관리 영역활동을 정의한다. 각 영역활동은 작업으로 구성되고, 각 작업은 단위 활동과 활동을 담당하는 역할로 정의된다. 각 활동은 일련의 순차적인 절차를 포함하고 선택적으로 사용 가능한 기법들을 포함하여 이를 이용하여 산출물들을 작성한다.

1) 개발 프로세스

도메인 영역활동은 제품이나 제품 개발을 위한 자산에 대한 요구 사항을 시장과 관련 이해당사자들로부터 수집하고 체계적으로 구성하여 개발할 시스템의 예상되는 기능과 품질에 대한 포괄적인 범위와 내용을 결정하고 제품계열 아키텍처와 특정 제품의 아키텍처를 유도할 수 있는 기반을 마련한다.

아키텍처 영역활동은 시스템을 구성하는 소프트웨어와 하드웨어 컴포넌트들의 전체 구조를 정의하는 시스템 아키텍처 설계를 통해 충돌 가능한 시스템의 요구 사항들을 조정하기 위해 시스템 구성 요소를 식별하고 요소들간의 구조를 설계하고 검증하도록 지원한다.

설계 영역활동은 아키텍처 결정 사항을 반영하면서 시스템의 구조 및 행위를 모델링하여 시스템의 구조를 정의한다.

구현 영역활동은 아키텍처와 설계 영역활동에서 정의된 제품 특성에 맞도록 컴포넌트들을 구현하여 제품의 특성에 맞게 설정하고 생성한다.

테스트 영역활동은 분석, 설계된 결과대로 개발되었는지 확인하는 것으로써 분석모델과 설계모델에 대한 검토와 구현물에 대한 테스트를 통하여 확인한다.

2) 관리 프로세스

프로젝트 관리 영역활동은 주어진 예산, 시간, 인원 등을 효율적으로 관리하여 개발의 생산성과 품질 향상을 위한 체계화된 관리 행동을 수행한다.

품질 관리 영역활동은 프로젝트 관리 활동 및 프로젝트 수행을 위한 다양한 지원 활동들을 수행함으로써 품질 보증 활동, 형상 관리 활동, 측정 및 분석 활동, 의사 결정 활동과 같은 프로젝트를 지원하는 활동으로 구성된다.

3. 개발방법론 검증

개발방법론의 사상과 개발 지침이 실제 프로젝트에서 유용한지 검증하기 위해 소규모 파일럿 프로젝트를 2회 수행하며 1차 검증을 마무리하였다. 이를 통해 임베디드 시스템 개발 시 하드웨어와 밀접한 관련이 있는 작업들에 대한 지침을 보완할 수 있었다.

첫째, 파일럿 프로젝트는 자동차 네비게이션 시스템에 사용되는 장착된 PDA 플랫폼 개발을 전문으로 하는 회사와 공동으로 진행하였고, 주로 하드웨어 설계 및 제작, OS 포팅과 같은 소프트웨어와 하드웨어와의 병행 개발에 초점을 맞춰 진행하였으며, 결과적으로 목표했던 체계적인 소프트웨어와 하드웨어의 표준화된 정보 공유 시점 및 내용을 파악하는 데 매우 유용한 결과를 획득할 수 있었다.

둘째, 파일럿 프로젝트는 스마트 카드를 이용한 결재 시스템인 POS를 개발하는 것이었는데, 이는 주로 임베디드 소프트웨어를 중심으로 UML2.0 표

준 표기법을 준수하여 체계적으로 설계하는 데 초점을 맞춰 진행하였으며, 2005년 12월 현재 구현 및 테스트 과정이 진행중이다.

Ⅲ. 방법론 지원도구 PRiME

방법론 지원도구는 소프트웨어 개발 조직의 구성원들로 하여금 자발적이고 지속적으로 개발방법론을 활용하고 활용 결과를 바탕으로 개발방법론의 개선을 유도할 수 있도록 개발방법론을 정의하고, 조정하며, 참조할 수 있도록 지원한다. 뿐만 아니라, 보다 적극적으로 개발방법론을 프로젝트에 적용하고 프로젝트 수행 경험(수행사례)을 지식화하여 전사적인 지식 자산을 공유할 수 있는 환경을 제공한다. 즉, 개발방법론의 도입 시, 도구를 적시에 소프트웨어 개발 프로젝트에 적용시켜 개발방법론의 활용성을 높이며, 프로젝트 및 자산의 체계적인 관리를 통하여 개발방법론을 개선하고자 하는 것이다.

이러한 점을 고려하여 현재까지 다양한 개발방법론 지원도구가 개발되어 출시되었다[6]-[10]. 일반적으로 개발방법론 지원도구는 개발방법론을 정의하고, 조정하고, 참조하며, 각종 가이드를 제공한다. 그러나, 출시되어 있는 기존의 지원도구들은 소프트웨어 실무 개발자들에게 프로젝트 수행과의 연계활동이나 적용사례 공유 및 자산 관리 기능, 그리고 프로세스 개선 기능 등이 제한적으로 지원되거나 매우 미흡한 형편이다. 이에 최근 대표적인 개발방법론 지원도구들의 기능분석을 통하여 개발방법론 지원도구의 필수 요구사항들을 파악하고 기존 개발방법론 관리도구의 문제점을 개선한 PRiME 시스템을 개발하였다. 특히 PRiME 시스템은 다양한 개발방법론을 적극 수용할 수 있고, 필요에 따라 확장을 지원할 수 있도록 호환성을 고려하여 SPEM[11]을 기반으로 방법론의 정의 및 조정 기능을 구현하였다.

<표 2>는 개발방법론 지원도구들의 다양한 사례를 통해 기능을 분석하고 비교한 것이다.

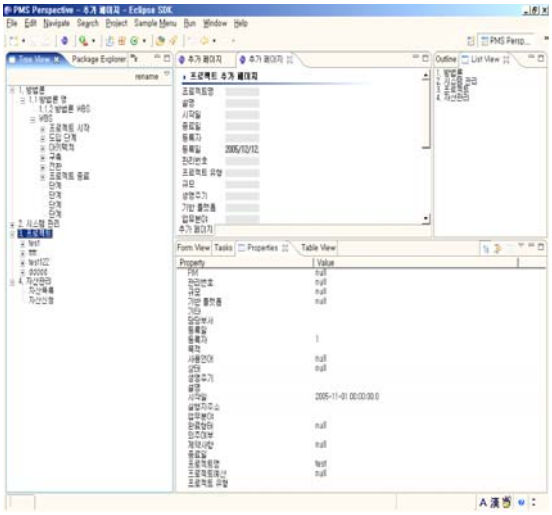
PRiME 시스템은 프로젝트 수행 시, 개발자들에게 다양한 분석, 설계 도구 및 개발도구와의 자연스러운 연계작업을 지원하기 위해 Eclipse 플랫폼의 플러그 인 형태로 개발하였다. (그림 2)는 PRiME의 프로젝트 관리화면을 보여준다.

현재 PRiME 시스템은 임베디드 시스템 개발 업체에서 전사적 차원으로 EMMA의 보급 및 교육의 목적으로 임베디드 시스템 개발에 대한 체계적인 프로세스, 프로젝트 및 자산관리 기능을 지원하고 있다. 앞으로 SPEM 기반의 다양한 개발방법론과 시범 프로젝트 사례 등록을 통해 프로젝트 및 자산 데이터를 축적하여 개발방법론 및 프로세스 개선을 지원할 예정이다.

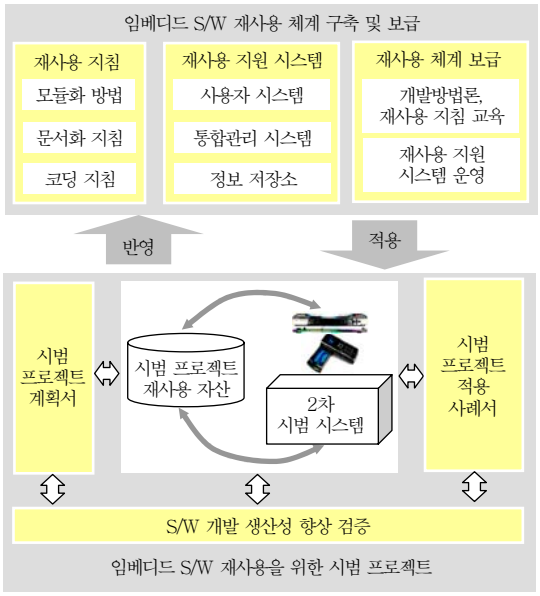
〈표 2〉 개발방법론 지원도구 기능 비교

기능 항목	C/S 10000	AllFusion	IRIS	Select Solutions	PRiME
방법론 기반의 WBS 정의	●	●	●	●	●
MS Project와 연동 또는 비연동 계획수립	○	○	●	○	●
메트릭 관련 정보 수집	×	○	●	○	○
메트릭을 활용한 프로젝트 수행도 평가	×	×	×	×	○
품질 관련 표준과의 연계 (예: CMMI 또는 SPICE 지원 여부)	○	○	●	○	●
프로세스 및 방법론 개선	○	○	○	○	●
산출물 재사용을 위한 자산화 기능	○	○	×	×	●
정보의 XML 형태 저장	×	×	○	●	●

주) ●: 포괄적 지원, ○: 부분적 지원, ×: 지원 안함



(그림 2) PRiME 실행 화면–프로젝트 관리도구



(그림 3) 재사용 체계의 구성

Ⅳ. 재사용 체계

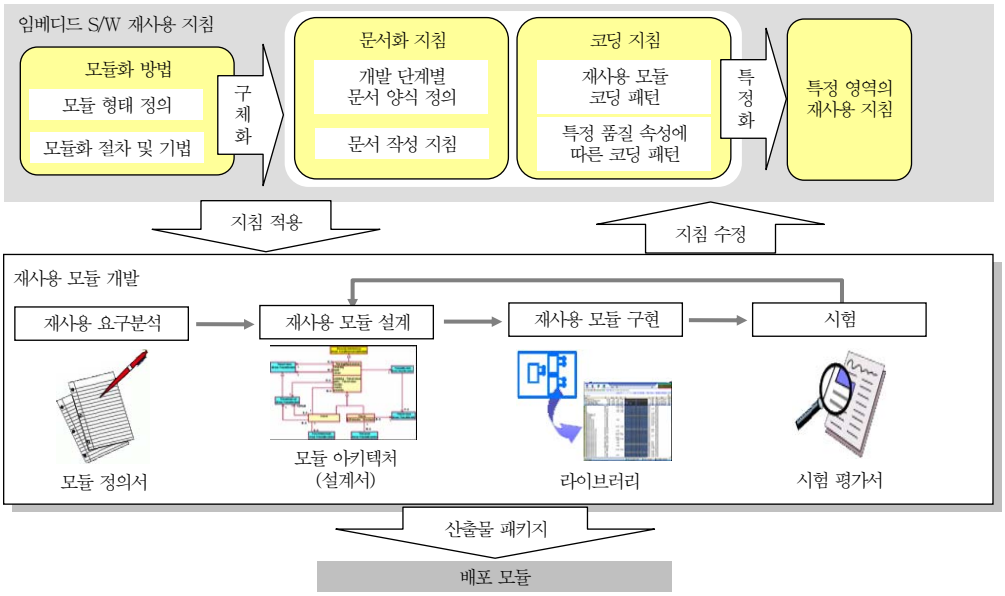
1. 재사용 체계 개요

오늘날의 대부분의 소프트웨어에서 요구하는 실시간성, 다양성, 이동성 등의 특성은 임베디드 소프트웨어가 실현해야만 하는 공통 특성들이므로, 임베디드 소프트웨어 기술의 발전은 타분야 소프트웨어 산업 기술 향상을 유도할 수 있다. 따라서 정보 기술(IT) 분야에서 파급 효과가 큰 임베디드 소프트웨어의 공통·핵심 기술을 자산화하고 이를 체계적으로 재사용할 수 있는 환경을 구축하는 것은 신 성장 동력 사업 수행에 공통적으로 필요한 핵심 기술들의 중복 개발을 최소화하고, IT 839 산업 간의 기술적, 경제적 파급효과를 증대시킬 수 있는 중요한 접근방법이다. 즉, 다양한 디지털 제품에 내장되어 생산에서 소비에 걸쳐 여러 계층의 IT 수요를 만족시킬 수 있는 임베디드 소프트웨어를 재사용 가능한 형태로 공급함으로써 관련된 산업분야에서 임베디드 시스템의 고부가가치를 생성, 확대시킬 수 있다. 그러므로 임베디드 소프트웨어 개발에 필요한 다양한 지식을 체계적으로 분석하고 모형화함으로써 임베디드 소프트웨어를 자산화하고 이를 목표 시스템 개발을 위해 창의적으로 재사용하도록 지원하는 체계가 요구된다.

(그림 3)은 본 연구팀에서 개발중인 임베디드 소프트웨어 재사용 체계의 개발 내용을 도식화한 것이다. 임베디드 소프트웨어 재사용 체계 구축 및 보급에서는 임베디드 소프트웨어 재사용 지침과 임베디드 소프트웨어 재사용 지원 시스템을 개발하고 그 결과물들을 보급하며, 실제 임베디드 소프트웨어 재사용을 위한 시범 프로젝트 수행을 통해서 임베디드 소프트웨어 재사용 자산을 개발 및 확장하여 성공적인 재사용 사례를 구축하고 평가함으로써 재사용 체계의 타당성을 검토한다. 또한 소프트웨어 재사용을 통한 정량화된 소프트웨어 생산성 향상을 확인하기 위한 소프트웨어 생산성 검증 방법도 함께 연구한다.

2. 재사용 지침

다양한 환경 제한적인 특성들을 갖는 임베디드 소프트웨어를 공유 가능한 개별 단위의 재사용 자산으로 구현하기 위해서는 체계적이며 명확한 개발 절차와 기법이 필요하다.

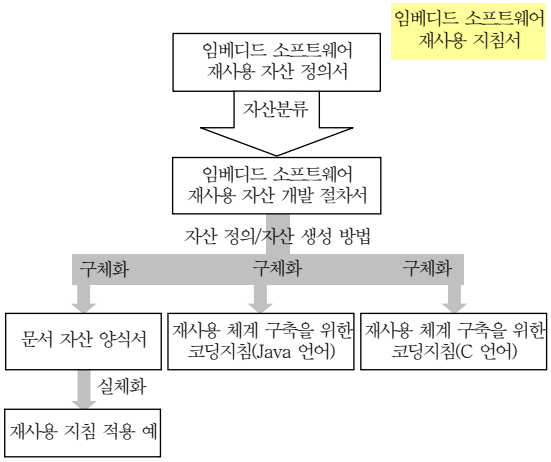


(그림 4) 재사용 지침의 적용

그리고 조직 내 소프트웨어 재사용 문화 확산을 통한 생산성 향상을 목적으로 단순하고(small), 실질적인(practice) 재사용 템플리트(reusable template)를 제공하는 재사용 지침을 개발중이다.

(그림 4)는 본 연구팀에서 수행중인 재사용 지침의 연구 내용 및 적용 과정을 도시화한 것이다. 재사용 지침에서는 재사용 자산의 형태 정의, 개발 절차 및 기법을 정의하고 재사용 자산의 개발 단계별 문서화 형태를 규정하여 제공함으로써 임베디드 소프트웨어 자산의 가독성과 재사용성을 높인다. 또한 C 언어와 Java 언어로 개발되는 소프트웨어 재사용 자산을 위한 코딩 스타일을 개발하고 성능이나 보안 등의 특정 품질 속성을 반영하는 코딩 패턴을 개발하여 제공한다. 그리고 실제 재사용 지침을 적용한 성공적인 사례를 개발하여 보급한다.

(그림 5)는 임베디드 소프트웨어 재사용 지침의 구성이다. 재사용 지침은 임베디드 소프트웨어의 자산 가치를 지속적으로 증대시켜 중복 개발 비용을 최소화하고 개발 경험을 공유하도록 한다. 또한 간소화된 문서 양식의 보급을 통해 조직 내 개발자들이 동의하는 재사용 필수 항목들의 양식을 표준화하고, 코드의 가독성을 높일 수 있는 코딩 지침을 권장



(그림 5) 재사용 지침의 구성

하여 소프트웨어 자산의 재사용 효과를 높일 수 있도록 지원한다.

3. 재사용 지원 시스템

재사용 지원 시스템은 개발자가 아닌 제3자가 필요에 따라 이미 구축되어 관리되고 있는 소프트웨어 자산들을 이해하고 획득하는 것을 지원하기 위한 자동화된 시스템을 의미한다. 기존의 소프트웨어 재사

용 시스템은 단순한 코드 검색을 위한 라이브러리 시스템 개념이 많아 실제 개발자가 원하는 개발 및 유지보수 정보는 제공하지 않는 한계점이 있다. 따라서 개발자 관점에서 임베디드 소프트웨어를 체계적으로 수집, 분석, 모형화, 탐색하고 활용할 수 있는 실질적인 접근 방법을 소프트웨어 재사용 지원 시스템을 통해 구축한다면 소프트웨어 재사용 문화의 정착이 실제적으로 가능해진다.

본 연구팀에서는 임베디드 소프트웨어 재사용 자산의 보급 확산을 위한 공유 체계의 제공을 목적으로 임베디드 소프트웨어 재사용 자산 및 이와 관련된 산출물과 메타데이터를 등록하고 기술 수요자의 필요에 따라 조회 및 재사용이 가능한 시스템을 구축 및 보급한다.

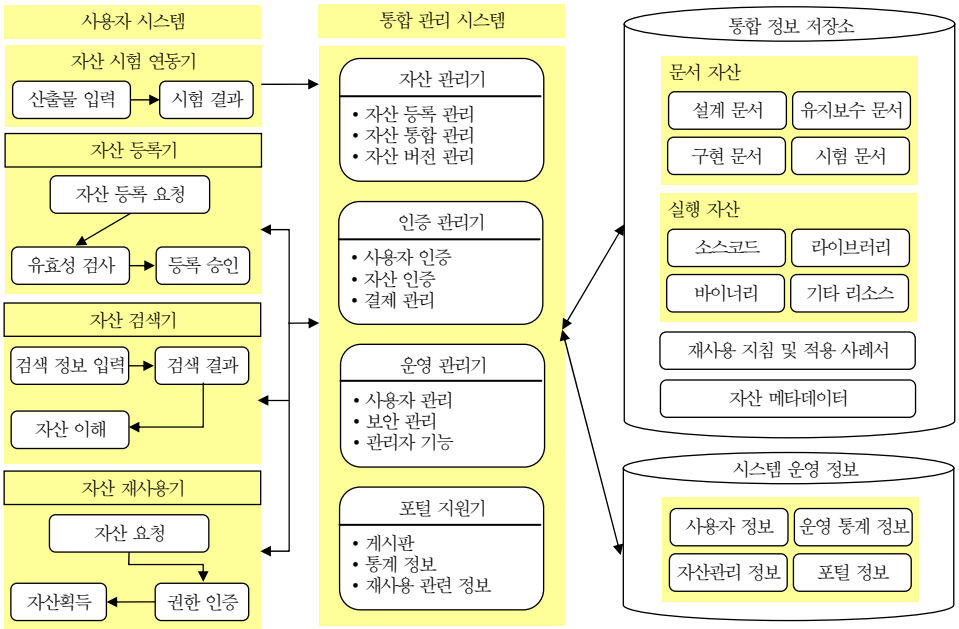
(그림 6)은 임베디드 소프트웨어 재사용 지원 시스템의 구성도다. 사용자 시스템의 자산시험 연동기는 자산의 재사용 지침 준수 여부를 검증하기 위한 것이며 자산 재사용기는 재사용 자산의 요청, 획득 및 결재과정 등을 지원하기 위한 것이다. 그리고 통합 관리 시스템에서 자산 관리기는 자산의 등록, 수정, 통합, 버전관리 등을 지원하기 위한 모듈이며 인증 관리기는 사용자 및 자산의 인증, 결재관리를 지원한다.

운영 관리기는 시스템 운영 관리 및 보안 기능을 지원한다. 마지막으로 포털 지원기는 사용자용 게시판, 각종 통계 정보 및 재사용 관련 정보를 지원한다.

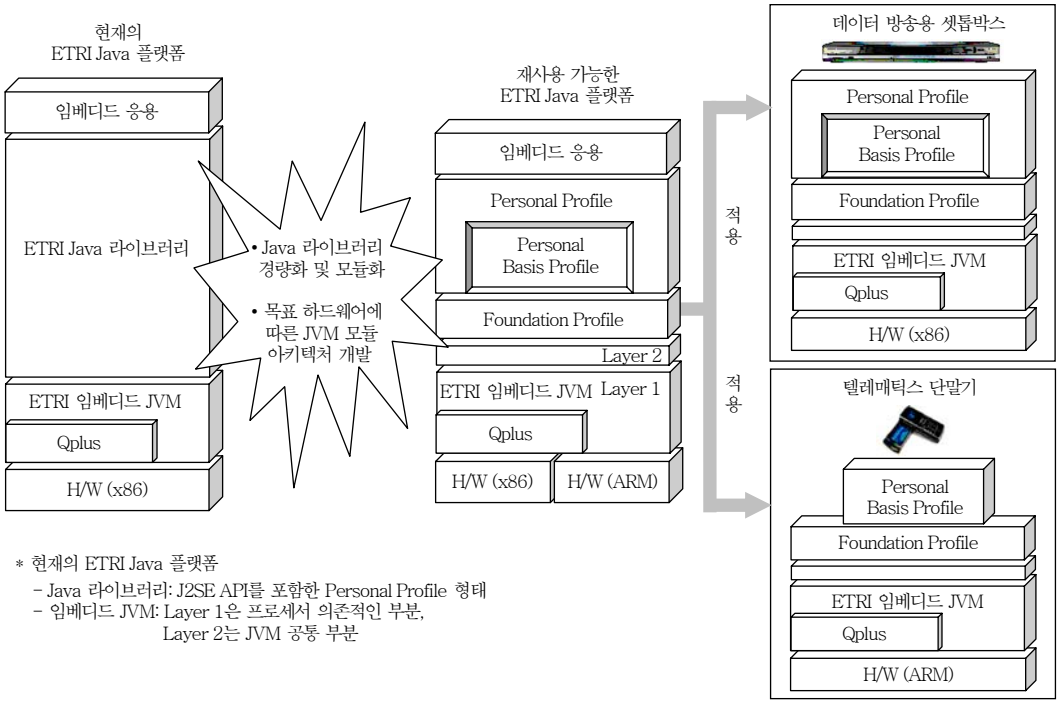
4. 재사용 시범 프로젝트

본 연구팀에서는 임베디드 소프트웨어 재사용 자산을 개발하여 다양한 임베디드 제품 개발에 재사용하는 성공 사례 도출을 위해, 재사용에 의한 임베디드 소프트웨어 개발 생산성 향상을 검증하고, 임베디드 소프트웨어 자산들을 연구원 내 신성장 동력 사업 등 임베디드 소프트웨어 기술 개발 사업에 적용하는 시범 프로젝트를 추진중이다. 1차년도에는 ETRI 임베디드 Java 플랫폼의 경량화 및 모듈화를 목표로 재사용 시범 사례를 구축하고 있다.

(그림 7)은 재사용 시범 프로젝트 구축의 연구 내용을 표현한 것이다.



(그림 6) 임베디드 소프트웨어 재사용 지원 시스템의 개념적 구조



(그림 7) 재사용 체계 시범 프로젝트의 내용 및 적용 사례

이를 위해 Java 라이브러리의 경량화 및 모듈화를 구현한 후, 데이터 방송용 셋톱 박스와 텔레매틱스 단말기에 적용하고 그 결과를 정량적으로 평가하여 시범 프로젝트 결과서를 작성하여 보급할 예정이다.

V. 향후 연구 방향

개발 방법론은 3개년 연구 중 2차년도로써 1차 검증을 위한 소규모 파일럿 프로젝트를 실시하여 검증하고, 이를 통해 EMMA 버전 1.0을 확정하였으며 관련 업계에 기술이전을 실시할 계획이다. 마지막 3차년도에는 중규모 이상의 실 프로젝트를 수행하고 그 결과를 반영하여 미비한 점을 보완한 후 방법론 지원도구와 함께 2006년 말에 버전 2.0을 보급할 예정이다.

2005년 1차년도로 수행된 재사용 체계는 조직 내 새롭게 개발할 재사용 자산의 개발과 활용에 초점을 두었다. 따라서, 재사용 자산의 개발 기법과 자산의 형태 및 코딩 패턴을 정의하고, 이를 바탕으로 임베디드S/W연구단 내 Java VM 개발에 적용하여 그 산출물들을 재사용 지원 시스템에서 운영하도록 하였다. 2006년에는 개발한 재사용 체계의 적용 및 보급에 중점을 두어 모범적인 사례를 만들어 내고 재사용에 의한 효과를 객관화시킬 수 있는 메트릭을 제시함으로써 연구원 내 재사용 문화 정착을 유도할 계획이다. 이를 위해 기존 소프트웨어를 재공학(reengineering)을 통해 자산화하는 방법을 개발하고 연구원에서 생산된 소프트웨어 자산의 효율적 관리 및 운영을 위한 지원 시스템을 확대 개발하여 보급할 예정이다.

약어 정리

EMMA	EMbedded MArmi
MARMI	Magic and Robust Methodology Integrated
OMG	Object Management Group
POS	Point of Sales
PRiME	PRocess innovation & Methodology

	Enhancement
SPEM	Software Process Engineering Metamodel
UML	Unified Modeling Language
WBS	Work Breakdown Structure

참 고 문 헌

[1] A. Jaaksi, "Developing Mobile Browsers in a Product Line," Software, IEEE, Vol.19, Issue 4, July-Aug. 2002, pp.73-80.

[2] BiggerStaff, Ted, and Alan J. Perlis, Software Reliability, New York: ACM Press, 1989.

[3] Paul Clements and Linda Northrop, "Software Product Lines," Addison-Wesley, 2001.

[4] Software Process Engineering Meta Model, http://www.omg.org

[5] UML, http:///www.omg.org

[6] http://www.cscl.com: Product Name: CS/10,000, CS/10,000 Release 1.1.136: Client/Server Connection Ltd., 1997.

[7] http://www3.ca.com: Product Name: AllFusion: Computer Associates, 1998.

[8] http://www-106.ibm.com/developerworks/rational: Product Name: Rational Process Workbench: IBM Rational, 2003.

[9] http://www.selectbs.com; Product Name: Select Process Director Solution: Select Business Solutions, Inc., 2003.

[10] http://www.osellus.com/: Product Name: IRIS Suite; Osellus, 2004.

[11] Software Process Engineering Metamodel, http://www.omg.org/technology/documents/formal/spem.htm