모바일 임베디드 소프트웨어 프로세스 개선 프레임워크☆

Framework for Improving Mobile Embedded Software Process

이 성 욱* 김 행 곤** 김 성 원***
Seung-Woo Shin Haeng-Kon Kim Soung Won Kim

요 약

유비쿼터스 시대에 핵심인 모바일 임베디드 시스템은 제품의 신뢰성 및 상품성을 위해서 과거에는 하드웨어에 초점을 두었지만 오늘날 소프트웨어에 더 초점을 둔다. CMMI와 SPICE와 같은 프로세스 개선모델은 일반 소프트웨어의 품질향상에서뿐만 아니라, 모바일 임베디드 소프트웨어 품질향상에서 또한 필요하다. 하지만 높은 비용과 무거운 프로세스로 인하여 모바일 임베디드 소프트웨어 프로세스 개선하기는 쉽지 않다. 반면 고객과 소통, 반복 개발의 특징을 가진 XP는 고객의 빈번한 요구사항과 특정 목적에 따라 하드웨어를 제작하는 모바일 임베디드 소프트웨어 개발에 적합하다. 따라서 본 논문에서는 모바일 임베디드 소프트웨어 개발 조직에서 XP의 실천사항에 기반하여 CMMI 수준 2 또는 3을 달성할 수 있는 소프트웨어 프로세스 개선 프레임워크를 제안한다. 더불어 개선 프레임워크를 지원하는 MESPIS를 설계하고 구현한다.

향후 연구로는 제안한 프레임워크를 실제 프로젝트에 적용하여 결과 분석에 따라 개선하고, 이를 바탕으로 MESPIS의 기능 보강이 필요하다.

ABSTRACT

The embedded software has been become more important than the hardware in mobile systems in ubiquitous society. The improvement models such as CMMI(Capability Maturity Model Integration) and SPICE(Software Process Improvement and Capability dEtermination) are used to improve the quality of software in general systems. Software process improvement is also necessary for mobile embedded software development to improve its quality. It is not easy to apply the general software improvement model to the mobile embedded software development due to the high cost effectiveness and heavy process. On the other hand, XP has the characteristics on focused communications with customers and iteration development. It is specially suitable for mobile embedded software development as depending on customer's frequent requirement changes and hardware attributes.

In this paper, we propose a framework for development small process improvement based XP(eXtreme Programming)'s practice in order to accomplish CMMI level 2 or 3 in mobile embedded software development at the small organizations. We design and implement the Mobile Embedded Software Process Improvement System(MESPIS) to support process improvement. We also suggest the evaluation method for the mobile embedded software development process improvement framework with CMMI coverage check by comparing other process improvement model. In the future, we need to apply this proposed framework to real project for practical effectiveness and the real cases quantitative. It also include the enhance the functionality of MESPIS.

☞ KeyWords : 모발일 임베디드 소프트웨어(Mobile Embedded Software), 소프트웨어 품질 평가(Software Quality Evaluation), 애자일 방법론(Agile Method), 모델스토리카드(Model Story Card)

1. 서 론

유비쿼터스 시대가 급속도로 진행되면서 과거 특정 산업용 기기 제어를 위해 사용되던 임베디

* 준 회 원 : 대구가톨릭대학교 컴퓨터정보통신공학과 석사과정 selab@cu.ac.kr
** 정 회 원 : 대구가톨릭대학교 컴퓨터공학과 교수 hangkon@cu.ac.kr(교신저자)
*** 정 회 원 : 안양대학교 전기전자공학과 교수 swkim@anyang.ac.kr
[2009/01/02 투고 - 2009/01/10 심사(2009/07/13 2차) - 2009/08.07 심사완료]
☆ 이 논문은 2009년도 한국연구재단 지역대학 우수과학자 지원사업 지원을 받아 수행된 연구임 (No..R 2009-0067365)

드 시스템은 오늘날 많은 디지털 기기의 보급과 함께 널리 이용되고 있다. 임베디드 시스템은 일반 컴퓨터와는 달리 특정 목적에 맞도록 설계된 하드웨어와 이를 제어하고 사용하기 위한 소프트웨어로 구성된다[1][2]. 따라서 임베디드 시스템 개발은 하드웨어와 소프트웨어를 동시에 제작해야하는 어려움이 따르며 최근 이러한 프로젝트에서 성공보다는 실패하는 사례들이 더 많다. 프로젝트의 성공 및 제품 품질 향상과 신뢰성을 위해 과거에는 하드웨어 중심으로 노력하였으나 최근에는 소프트웨어에 더 많은 초점을 두고 있다[3].

소프트웨어의 품질향상을 위한 방법에는 제품 자체에 대한 평가와 개발 과정 자체를 평가하는 두 가지 방법이 있다. 개발된 소프트웨어의 평가로 제품의 질을 높이기에는 한계가 있으며, 소프트웨어 개발 과정을 모니터링하며 결과에 대한 문제점 분석으로 최종적인 소프트웨어 제품 향상을 위한 프로세스 개선이 더 많이 요구된다[4].

따라서 모바일 임베디드 소프트웨어는 고수준의 신뢰성이 요구되며 이들 소프트웨어의 품질 향상과 신뢰성을 향상시키기 위해서는 개발 프로세스에 대한 개선이 절대적으로 필요하다.

일반적인 소프트웨어 프로세스 개선을 하고자 할 때, 표준으로 알려진 Capability Maturity Model Integration(CMMI)나 Software Process Improvement and Capability dEtermination(SPICE) 같은 모델을 도입하며, 이 모델에서 표현되어지는 수준 달성에 의해 소프트웨어 개발 조직의 전체, 팀, 프로세스가 평가된다[5][6]. 이는 대외적으로 그 개발 조직이 어느 수준에서 제품을 개발하고 있다는 기준이 되기 때문이다. 모바일 임베디드 소프트웨어 개발 조직 역시 CMMI나 SPICE의 수준 달성으로 평가 받을 수 있지만 프로세스 중심에 중소규모 조직에서는 부담이 될 수밖에 없다. 따라서 모바일 임베디드 소프트웨어 품질향상 및 신뢰성을 높이기 위해서는 개발 조직에 맞는 작은 개선 모델이 필요하다[3].

프로세스 중심의 개선 모델과 병행해서 불필요한 산출물은 버리고 꼭 필요한 가치 실현에 중심을 두는 Agile 방법론이 있다. Agile 방법론은 개발하고자하는 소프트웨어에 대한 활동을 중요시하며 이 방법론에서 가장 많이 쓰이는 것으로 eXtreme Programming(XP)과 Scrum, Crystal, Feature-Driven Development(FDD) 등이 있다[9].

XP는 소프트웨어 개발 조직에서 꼭 필요한 실천사항만을 제시하며 이 실천사항들이 실제 프로젝트에서 수행되기를 명시하고 있다[7].

본 논문에서는 모바일 임베디드 소프트웨어의 제품 향상을 위해 CMMI 수준 달성과 개발 프로세스의 개선을 목표로 한다. 무거운 CMMI를 좀 더 단순화하며 기본 바탕을 XP와 같은 Agile 방법론의 원칙에 기준한 개선 프레임워크를 제안한다. 개발 조직에서 CMMI 수준 2 또는 3을 달성하기 위한 조건 확인과 Agile 방법론 및 XP 분석을 통하여 모바일 임베디드 소프트웨어 개발 프로세스에 적합한 활동을 정의하고, 추출한다. 이는 모바일 임베디드 소프트웨어 개발 프로세스의 개선 모델을 제시하고, 이를 통해 Plan-Do-Check-Act(PDCA) 기반의 개선하는 적용 흐름을 제시한다.

2. 관련 연구

2.1 CMMI

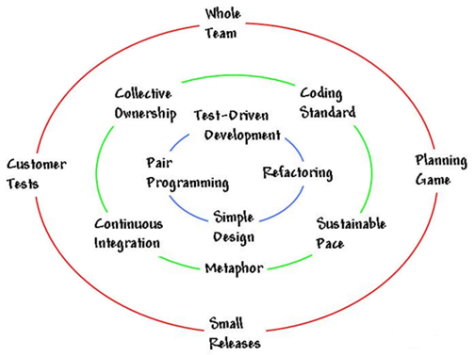
CMMI는 1991년 미 국방부의 요청으로 미국 카네기멜론 대학 소프트웨어 공학연구소에서 개발한 소프트웨어 프로세스 성숙도 평가 모델로 조직의 프로세스 개선 활동을 효율적으로 지원하는 모델이다. 2006년 8월 SEI에서 발표한 1.2 버전에는 총 22개의 프로세스 영역을 가지며 조직, 프로젝트, 프로세스에 대해서 단계적 표현과 연속적 표현으로 나타낸다[5][7][8].

CMMI의 모델 구조는 성숙단계 또는 능력단계를 시작으로 프로세스 영역(Process Area: PA)에서 일반 목적(Generic Goal: GG)과 특정 목적(Specific Goal: SG)의 공동 수행 항목(common feature)과 일반 실천사항(Generic Practice: GP), 특정 프랙티스(Specific Practice: SP)를 정의한다.

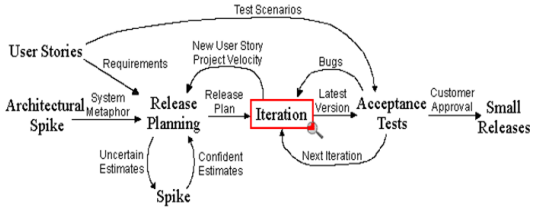
조직에서 해당업무를 얼마나 체계적으로 수행하고 있는지를 나타내는 지표로 성숙단계는 단계 1에서 5까지 5 단계로 구분하고 능력단계는 단계 0에서 5까지 6단계로 구분한다. 프로세스 영역은 해당 프로세스를 위해 수행되어야 하는 활동들을 모아 놓은 것으로 하나의 프로세스 영역은 반드시 성숙단계 2~5중 하나에 포함되며 능력단계에서는 22개의 프로세스를 프로세스 관리(process management), 프로젝트 관리(project management), 엔지니어링(engineering), 지원(support) 영역(Category)으로 그룹화 했다. 이는 개별 프로세스 영역별로 능력 단계를 부여할 수 있도록 했기 때문이다.

2.2 XP (eXtreme Programming)

Agile 개발 방법론에서 가장 대표적인 방법인 XP(eXtreme Programming)는 단순성, 상호소통, 피드백, 용기 등의 4가지 원칙에 기반 한다. 요구사항의 변경으로 인한 비용이 개발 기간에 상관없이 일정하게 유지되도록 하는 것이 핵심이다.



(그림 1) XP의 실천사항

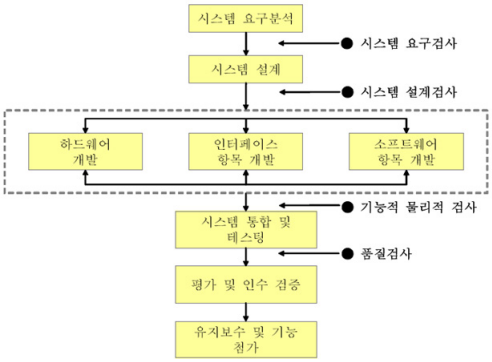


(그림 2) XP Project

그림 1은 XP의 4가지 원칙을 바탕으로 계획, 설계, 개발에 따라 구분지은 실천사항에 대한 간략한 그림이다[7][11]. 그림 2는 실천 사항을 바탕으로 프로젝트 수행에서의 가장 기본적인 프로세스를 나타내며 빠른 개발 속도를 위한 반복개발이 핵심이며, 반복개발에 따라 주기적으로 작은 배포를 수행한다.

2.3 모바일 임베디드 S/W 개발 방법론

임베디드 시스템은 요구되는 하드웨어와 소프트웨어에 따라 다양하게 설계되어 개발된다. 과거에는 단순한 하드웨어와 고정된 소프트웨어의 결합으로 보드상에서 구현되었던 것이 일반적이었지만, 현재는 다양한 에뮬레이터 환경에 의해 구현 환경에서 미리 테스트를 거쳐 최종 하드웨어에 소프트웨어를 올릴 수 있기 때문에 조금은 개발이 자유로워진 편이다. 그림 3은 이러한 임베디드 시스템의 개발 절차를 보여준다.

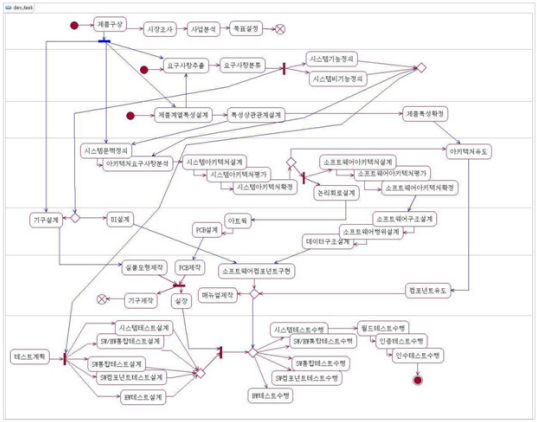


(그림 3) 임베디드 시스템 개발 절차

하드웨어 개발과 소프트웨어 항목 개발이 나누어지며 하드웨어와 소프트웨어 항목간의 인터페이스 개발이 중요하다. 또한 이에 따라 테스트를 수행하는 부분은 소프트웨어 테스트, 소프트웨어 통합 테스트, 시스템 통합 테스트, 인수 테스트 등 다양하다. 모바일 임베디드 소프트웨어 개발 절차 역시 이와 같은 절차를 따른다[12][13].

2.4 마르미

국내에서는 최근 한국전자통신연구소(ETRI)에서 마르미-Ⅳ라는 임베디드 시스템 개발 프로세스를 개발하였다[14]. Software Process Engineering Metamodel(SPEM)을 기반으로 정의되어 방법론과 프로세스의 이원화된 메커니즘으로 설계되었다. 때문에 프로젝트의 규모와 성격에 따라 쉽게 커스터마이징 할 수 있는 장점이 있으며, 또한 임베디드 시스템의 개발 특성을 잘 반영하고 있다[15]. 마르미-Ⅳ는 개발 경로 지침서, 영역활동 지첨서, 기법서, 산출문서정의서, 적용사례 등으로 구성되며 다음 그림 4는 마르미-Ⅳ의 전체 개발 프로세스를 나타내고 있다.



(그림 4) ETRI 마르미-Ⅳ의 개발 프로세스

마르미는 전체 프로젝트를 미니 프로젝트 별로 세분하여 단계적으로 구현하게 되어 있다. 이때, 계획 단계와 아키텍쳐 단계를 통해 미니 프로젝트를 정의하고 개발 계획을 수립하고 점진적 개발단계를 통해 미니프로젝트별로 구현하여 인도단계를 거쳐 시스템 인도를 하게 된다. 이런 점진적 반복적 접근을 하게 되면 필요한 기능별로 신속한 구현 및 적용이 가능하고 프로젝트의 위험이 감소하게 되며, 프로젝트 자원을 효율적으로 사용할 수 있게 된다. 이런 점진적 개발 방법을 할 때 고려해야 할 사항은 상세한 프로젝트 계획이 필요하다는 것이다. 기준과 종료를 확인할 수 있는 사항 및 기준이 필요하다.

3. 모바일 임베디드 프로세스 개선 프레임워크

3.1. 개요

모바일 임베디드 소프트웨어 개발 조직에 CMMI 및 SPICE를 바로 적용할 수 있지만 프로세스가 무거워 적용이 실패할 확률이 매우 높다. 그 이유는 프로세스 중심의 산출물 비중이 너무 많이 차지하여 중소규모 조직에 해당하는 모바일 임베디드 소프트웨어 개발조직에서는 부담이 될 수밖에 없고, 컨설턴트를 위한 비용에 대한 부담도 크다.

Agile 및 XP는 무거운 프로세스를 탈피하고, 실질적 개발 활동에만 초점을 맞춘 경량화 프로세스 모델이다. 반복 개발과 짝 프로그래밍이라는 독특한 개발 활동을 명시하고 있지만 이는 잦은 고객의 요구사항 변경과 하드웨어 특성에 의존적인 모바일 임베디드 소프트웨어 개발에 적합하다.

따라서 CMMI, Agile 방법론, XP를 분석하여 모바일 임베디드 소프트웨어 개발 활동을 중심으로 개발 프로세스의 개선으로 접근한다.

3.2. 프로세스 분석

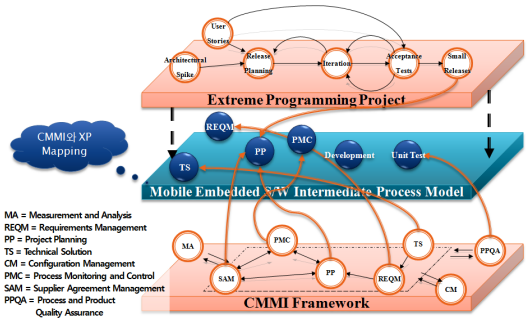
3.2.1 중간 프로세스 모델 분석

모바일 임베디드 소프트웨어 개발을 위한 프로세스 및 활동 추출을 위해 먼저 모바일 임베디드 소프트웨어 개발에 적합한 XP의 활동을 바탕으로 CMMI 프로세스 영역의 활동과 매핑을 통해 분석을 수행한다. 이는 아직 모바일 임베디드 개발 프로세스를 살펴보기 전 단계로 중소규모 조직에 해당하는 중간 프로세스 모델로 살펴볼 수 있다. XP의 활동을 기준으로 CMMI의 프로세스 영역별 활동을 살펴보면 다음과 같다.

- 계획 세우기(Planning Game) – 고객의 스토리 설명에 따라 배포 계획과 반복 계획을 세우고 속도에 대한 평가를 한다. 이는 CMMI의 프로젝트 계획 프로세스의 특정 목적 1항목 '견적 수립 활동'과 특정 목적 2항목 '프로젝트 계획 개발 활동'에 해당된다.
- 작은 배포(Small Release) – 짧은 주기 동안에 고객에게 새 버전의 간단한 제품을 제공할 수 있어야하는 실천 사항은 CMMI의 프로젝트 계획 프로세스에서 특정 프랙티스 1.3항목 '프로젝트 생명 주기 정의 활동'에 해당된다.
- 은유 및 단순 설계(Metaphor) – 은유는 전체 시스템이 어떻게 동작하는지를 설명하는 프로토타입이며, 단순 설계는 가능한 고객의 스토리만 충족할 수 있게끔 간단하게 설계해야 한다. 이 두 가지 실천 사항에 완벽하게 대응하는 CMMI의 프로세스는 없지만 기술 솔루션 프로세스의 특정 목적 2항목 '설계 계발 활동'에서 이와 관련된 내용이 언급된다.
- 고객 테스트, 단위 테스트(Customer Test, Unit Test) - XP의 승인 테스트는 고객 스토리로부터 생성된 테스트를 바탕으로 하는 블랙박스 시스템 테스트로 반복의 끝에 배포될 때 마다 고객에게 테스트 사항을 받아 품질 보증을 하는데 XP에서 가장 중요한 부분이다. 이는 CMMI의 프로세스 및 제품 품질 보증 프로세스에 특정 목적 1항목 '객관적인 프로세스/작업산출물 평가 활동'과 관계가 있다. XP의 단위 테스트는 자동화된 단위 테스트를 구축하고 이 테스트가 항상 완벽하게 실행되도록 하는데 이는 CMMI의 기술 솔루션 프로세스에 특정 목적 3항목 '제품 설계 구현 활동'에서 단위 테스트를 포함해야 한다고 명시하고 있다. 이 단위 테스트는 코드 공동 소유를 가능하게 하고 리팩토링을 준비한다.
- 리팩토링(Refactoring) – 설계 개선을 위해 단위 테스트에서 시작된다. 완벽하게 대응하는 CMMI 프로세스는 아니지만 단위 테스트에서 시작되고, 재사용을 위한 리팩토링 성격을 가지는 CMMI의 기술 솔루션 프로세스에 특정 프랙티스 2.4항목 '개발, 구입, 재사용 분석 활동'에 관련된 내용을 포함한다.
- 짝 프로그래밍 및 코드 공동 소유(Pair Programming, Collective Ownership) – 짝 프로그래밍은 하나의 컴퓨터에서 두 사람이 일하는 것이고 코드 공동 소유를 통해 XP의 가치인 용기를 북돋아 준다. 이는 CMMI의 확인 프로세스에 특정 목적 2항목 '동료 검토 활동'에 선정된 작업산출물에 대하여 동료검토를 수행한다고 명시하고 있다. 또 통합 프로젝트 관리+통합 제품 및 프로세스 개발 부분에서 팀 구성과 운영에 관한 명시를 하고 있다.
- 지속적 통합(Continuous Integration) – 비동기적 방식은 매일 빌드하고 동기적인 방식은 짝 프로그래밍으로 에피소드가 하나 끝날 때 통합한다. 이는 CMMI의 형상관리 부분에서 특정 프랙티스 1.2항목 '형상 관리 시스템 구축 활동'에 나타나는 도구를 통해 지원할 수 있다.
- 페이스 유지(Sustainable Pace) – 일하는 시간에 관한 내용으로 CMMI 에서는 대응하는 프로세스 부분이 없다. 하지만 측정 및 분석 프로세스 부분에서 특정 목적 1항목 '측정과 분석활동'을 통해 이 부분을 커버할 수 있다.
- 고객 참여(On Site Customer) – XP에서 전체 팀은 하나의 사무실에 개발자와 고객이 함께 프로젝트를 수행해야 한다. 이는 중소규모에 맞춰 CMMI 프로세스 4가지 범주 중에 조직 관리를 하나로 묶어 하나의 프로세스로 식별한다.
- 표준 코딩(Coding Standard) – 코드는 동의된 표준으로 포맷되어야 한다. 이는 CMMI에서 일치하는 프로세스 부분이 없지만 기술 솔루션 프로세스에 추가한다.

3.2.2 중간 프로세스 모델 추출

CMMI와 XP 사이의 활동에서 이뤄지는 중간 모델을 추출 하는 과정은 그림 5와 같다. CMMI의 프로세스인 Technical Solution(TS), Project Planning(PP), Process Monitoring and Control(PMC),



(그림 5) 모바일 임베디드 S/W 개발 중간 프로세스 모델 추출

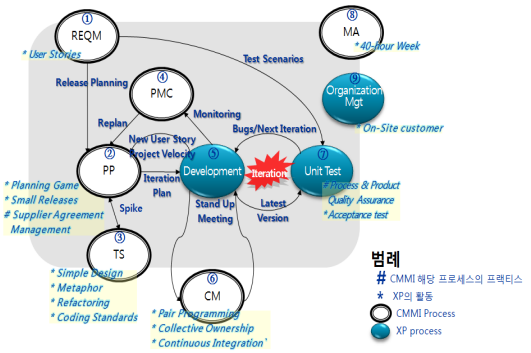
Requirements Management(REQM)의 활동에 추가적인 XP 활동을 포함시키거나 반대로 개발과 테스트 부분에서는 XP 실천 사항 중심에 CMMI 프로세스 활동을 수정하여 추가한다. Measurement and Analysis(MA), Configuration Management(CM)은 독립적 프로세스로 추출하며 조직 관리에 해당하는 프로세스는 하나의 프로세스로 묶어서 표현한다. 따라서 제안 모델 중간층에 표현되는 모바일 임베디드 소프트웨어 프로세스 모델은 기존의 CMMI 프레임워크와 XP 프랙티스 사이의 중간 계층으로 XP 활동에 기반을 두어 CMMI 수준 달성을 위한 부분 활동의 집합이다.

3.2.3 중간 프로세스 모델 활동

그림 6은 중간 프로세스 모델 추출을 통해 총 9개의 프로세스 영역으로 나눠서 제안되며, 프로세스 영역간의 관계에서 전반적인 개발 흐름을 간결하게 표현한다. 다음은 9개의 프로세스 영역에 대한 설명은 다음과 같다.

① 요구사항 관리 프로세스(REQM) - 프로젝트 시작점으로 유저스토리 작성에 관한 활동을 포함한다.

② 프로젝트 계획 프로세스(PP) - XP의 계획 세우기, 작은 배포 활동과 CMMI의 공급 계약 관리 프로세스 활동을 포함한다.



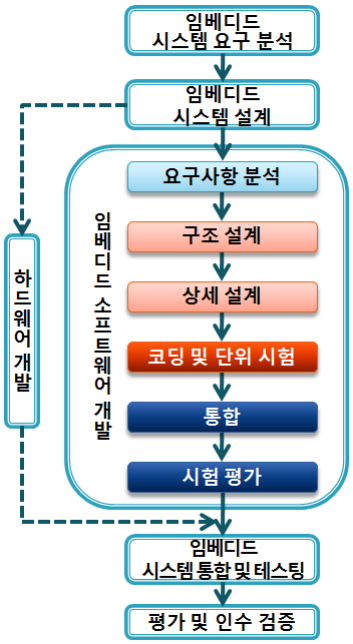
(그림 6) 모바일 임베디드 S/W 개발 중간 프로세스 모델 활동

③ 기술 솔루션 프로세스(TS) - 설계에 관한 활동 명시 및 기능별 코드 재사용에 관한 활동을 포함한다.

④ 프로젝트 모니터링 및 제어(PMC) - 프로젝트 계획 프로세스와 개발 프로세스 사이에서 모니터링하며 재계획에 관한 제어활동을 명시한다.

⑤ 개발 프로세스(Development) - 반복 개발을 기본적 활동을 바탕으로 기능 별 단위로 작성이 완성되었을 시 단위 테스트 프로세스 활동이 수행되도록 한다.

CMMI 주요 프로세스	영역	매핑
Casual Analysis and Resolution	Support	
Configuration Management	Support	코딩 및 단위 시험
Decision Analysis and Resolution	Support	
Integrated Project Management + IPPD	Project Mgt	
Measurement and Analysis	Support	
Organizational Innovation and Deployment	Process Mgt	
Organizational Process Definition+IPPD	Process Mgt	
Organizational Process Focus	Process Mgt	
Organizational Process Performance	Process Mgt	
Organizational Training	Process Mgt	
Product Integration	Engineering	통합, 시험평가
Project Monitoring and Control	Project Mgt	
Project Planning	Project Mgt	구조설계, 상세설계
Process and Product Quality Assurance	Support	통합, 시험평가
Quantitiative Project Management	Project Mgt	
Requirements Development	Engineering	요구사항 분석
Requirements Management	Engineering	요구사항 분석
Risk Management	Project Mgt	
Supplier Agreement Management	Project Mgt	
Technical Solution	Project Mgt	구조설계, 상세설계
Validation	Engineering	통합, 시험평가
Verification	Engineering	통합 시험평가



(그림 7) 임베디드 개발 프로세스와 CMMI의 매핑

⑥ 형상관리 프로세스(CM) - 개발을 지원하기 위한 프로세스로 짝 프로그래밍, 코드 공동 소유, 지속적인 통합 활동을 지원하기 위해 기본적인 형상관리 도구를 사용하도록 명시한다.

⑦ 단위 테스트 프로세스(Unit Test) - 프로세스 제품 품질 보증 관련 활동으로 인수 테스트 활동까지 함께 포함한다.

⑧ 측정 및 분석 프로세스(MA) - REQM, PP, TS, PMC, CM, Development, Unit Test 프로세스에 대한 메트릭을 측정하고 분석하는 활동을 포함한다.

⑨ 조직 관리 프로세스(Organization Mgt) - CMMI의 프로젝트 관리 프로세스 영역에 해당하는 프로세스 활동들을 하나로 묶어 기술 교육과 같은 조직 관리에 필요한 최소한의 활동을 포함한다.

3.2.4 임베디드 개발 프로세스와 CMMI 매핑

요구사항 분석 프로세스의 활동은 Requirements Development(RD), Requirements Management(REQM) 프로세스 영역에서 명시하는 고객 요구사항 개발, 요구사항 관리 등의 활동과 매핑된다. 구조설계, 상세 설계 프로세스의 활동은 Project Planning(PP), Technical Solution(TS) 프로세스 영역에서 명시하는 공수/비용 견적 활동, 제품-컴포넌트 해법 선택 활동 등과 매핑된다(그림 7).

코딩 및 단위 시험 프로세스 활동은 어떤 프로세스에도 개발에 주된 활동은 없지만 소스 관리를 위한 형상관리에 관한 활동이 Configuration Management(CM) 프로세스 영역에서 매핑된다. 끝으로 통합, 시험평가 프로세스의 활동은 Product Integration(PI), Process and Product Quality Assurance(PPQA), Validation(VAL), Verification(VER) 프로세스 영역에서 V&V에 기반을 둔 활동과 프로

세스 및 품질 보증을 위한 산출물 평가 활동 등이 매핑된다. 여기에서 매핑된 CMMI의 프로세스 영역에 해당하는 모든 활동을 가져오는 것이 아니라 개발 사례를 살펴보면 하드웨어 부분을 제외하였기 때문에 소프트웨어 부분에서 프로세스는 간단히 4단계로 분석, 설계, 구현, 테스트로 간략히 할 수 있고, 이에 모바일 임베디드 소프트웨어 개발 프로세스에 적합한 활동을 기준으로 CMMI의 프로세스를 검색한다. 반면 개발 프로세스만을 살펴보았기에 조직 관리에 해당하는 프로세스 활동은 하나도 발견되지 않음을 알 수 있다.

3.3. 프레임워크 설계

모바일 임베디드 소프트웨어 개발 프로세스 개선 프레임워크는 개선 모델과 이를 적용하는 방법, 그리고 지원 도구로 구성된다. 개선 모델은 네 가지 프로세스 영역에서 성숙도에 따라 나뉘어진 개선 활동의 집합 및 목표, 산출물로 구성되는 지침서라고 할 수 있다. 이 지침서를 바탕으로 적용하는 방법 및 개선 그룹 구성에 관한 내용을 포함한다. 지원 도구는 개선 활동함에 있어서 개선 그룹이나 개발 조직에서 사전에 수준을 판단하기 위한 지원도구이다. 이 모든 것을 포함하는 프레임워크는 다음 4가지 특징을 가진다.

첫째, 주요 개선 프로세스는 요구사항 관리 프로세스, 아키텍처 설계 프로세스, 구현 프로세스, 테스트 프로세스로 구성된다.

둘째, 개발 조직의 성숙도 수준에 따라 비 관리, 정성적 관리, 정량적 관리, 예측 관리 4단계로 표현한다. 0단계 비 관리는 어떤 프로세스나 개발 방법론이 없는 조직을 나타낸다. 1단계 정성적 관리에서는 일반적으로 모바일 임베디드 소프트웨어 개발 프로세스 모델을 순차적으로 수행하는 수준이다. 2단계 정량적 관리는 소규모 반복 개발하는 수준으로 팀에서 제품 출시 이전에 모듈별 사전 테스트나 에뮬레이터 환경을 통한 사전 테스트가 가능한 수준이다. 3단계 예측 관리는 4가지 프로세스에서 수행하는 활동을 모니터링하는 수준으로 데이터를 측정하여 분석하는 활동을 수행한다.

셋째, 각 프로세스 내에 단계별 목표와 이를 달성하기 위해 수행해야하는 활동 및 산출물이 존재한다. 또한 3단계 예측관리에 도달하기 위한 기본 메트릭과 모바일·임베디드 소프트웨어 특성상 유동적인 활동을 정의한다.

넷째, 모바일 임베디드 소프트웨어 개발 프로세스를 개선하기위해 데밍의 PDCA기법을 바탕으로 개발 조직의 프로세스를 사전 평가거나 혹은 개선 모델을 참조하여 바로 실행에 옮길 수 있다. 후에 개선 효과를 분석하여 개발 조직의 개발 프로세스로 정의한다.

3.3.1 주요 프로세스 활동 정의

그림 8은 개선 모델에서 프로세스 영역의 성숙도 수준을 표현하며 1단계보다 3단계에서 커버하는 활동이 많음을 나타낸다. 각 프로세스의 활동은 다음과 같이 정의한다.

(그림 8) 모바일 임베디드 개발 프로세스 영역 활동 및 성숙도 표현

가. 요구사항 관리 프로세스

표 1은 요구사항 관리 프로세스로 고객과 함께 하는 활동을 포함하며, XP에서 명시하는 유저 스토리를 산출물로 작성해 이를 바탕으로 요구사항 명세서를 작성하는 활동들로 나타낸다.

1단계에서는 요구사항 관리가 이뤄져야하며 그에 따른 활동으로는 유저스토리 작성과 개발자, 관리자, 고객이 한 자리에 모여야 하는 활동을 명시하며, 모바일 임베디드 소프트웨어 특성을 고려한 요구사항 분석 활동을 명시한다.

2단계에서는 1단계보다는 더 나아가 요구사항 변경에 대한 관리 활동을 명시한다. 이는 모바일 임베디드 소프트웨어 특성상 자주 발생하는 요구사항 변경에 따른 대안 마련을 활동으로 명시하여 언제든지 요구사항 변경에 대비하고 이를 관리해야할 것이다.

3 단계에서는 요구사항이 설계 혹은 개발 단계에서 이뤄져야 할 부분을 미리 검증을 통해 예측관리를 할 수 있도록 하는 것이다. 이에 따른 활동으로는 요구사항 변경 사항 자료를 항상 모니터링 하고 분석해야하는 것이다.

(표 1) 요구사항 관리 프로세스 활동

요구사항 관리 프로세스		
산출물		유저스토리, 요구사항명세서
1단계	목표	요구사항을 관리한다.
	활동	· 모바일 임베디드 소프트웨어 요구사항을 유저스토리 혹은 그에 준하는 문서를 작성한다. · 기능적/비기능적 요소를 식별한다. · 개발자, 관리자, 고객이 한 자리에서 모임을 갖는다. · 모바일 임베디드 소프트웨어 특성을 고려하여 요구사항을 분석한다.
2단계	목표	요구사항 변경을 관리한다.
	활동	· 고객의 요구사항 변경사항을 수집, 관리한다. · 문제 발생에 대한 대안을 마련한다.
3단계	목표	요구사항 추적성을 확보하고 요구사항을 검증한다.
	활동	· 요구사항 추적성을 확보하기 위한 변경 사항 자료를 분석한다. · 모바일 임베디드 소프트웨어가 탑재될 하드웨어 제약사항을 검토하고, 요구사항명세서를 검증한다.

나. 아키텍처 설계 프로세스

표 2는 아키텍처 설계 프로세스로 요구사항 관리 프로세스 단계에서 고객과 함께 작성한 요구사항 문서를 바탕으로 고객도 쉽게 알아 볼 수 있는 문서로 작성하는 것이 주요 목표다. 그렇기 때문에 산출물 중 프로토타입 역시 은유적 표현을 포함하도록 하며 간단한 아키텍처 설계서를 서술하게 된다.

1단계에서 목표는 앞서 설명된 은유적 설명이 포함된 설계에 관한 활동을 명시한다.

2단계에서는 이렇게 작성된 모바일 임베디스 소프트웨어 아키텍처를 관리해야한다. 단순히 고객과 소통을 위한 설계서로 그치게 된다면 개발에 문제가 발생할 수 있고, 또한 모바일 임베디드 소프트웨어 특성으로 고려하여 관리하도록 활동을 명시한다.

3단계에서는 구현단계에서 예측할 수 있도록 아키텍처를 모델링하여 최소 템플릿 소스까지 나올 수 있도록 활동을 명시한다.

(표 2) 아키텍처 설계 프로세스 활동

아키텍처 설계 프로세스		
산출물		프로토타입, 아키텍처 설계서
1단계	목표	은유적 설명이 포함된 간단한 설계를 한다.
	활동	· 모바일 임베디드 소프트웨어 아키텍처를 고객이 이해할 정도로 심플하게 설계한다. · 프로토타입을 작성한다. · 기능 구현 관련하여 비용을 따져 구현/재사용/구매를 결정한다.
2단계	목표	모바일 임베디드 소프트웨어 아키텍처를 관리한다.
	활동	· 모바일 임베디드 소프트웨어 아키텍처와 하드웨어 제약사항을 참고하여 평가, 검증한다.
3단계	목표	모바일 임베디드 소프트웨어 아키텍처를 모델링한다.
	활동	· 템플릿 소스작성이 가능한 모델링을 한다. · 모바일 임베디드 시스템 아키텍처로 확장한다.

다. 구현 프로세스

표 3은 구현 프로세스로 XP에서 명시하는 짝프로그래밍과 단위 테스트 등과 같은 활동을 명시하여 유연하고 반복적인 개발활동을 수반한다. 따라서 산출물에는 아키텍처 설계 프로세스의 활

동에 따라 템플릿 소스가 나올 수 있으며, 기능단위 모듈 소스 그리고 전체 소스가 된다.

1단계에서는 기능단위 모듈 소스를 개발하는 활동을 명시하여 조직이 반복 개발이 가능하도록 한다.

2단계에서는 1단계의 기능 단위 모듈 소스 개발이 내재화가 이뤄진다면 이를 바탕으로 테스트 주도 개발이 되도록 한다. 이는 모바일 임베디드 소프트웨어 특성상 예측하기 힘든 예외상황을 미리 찾아내기 위한 활동이다. 그에 따라 시스템 에뮬레이터 환경 구축 혹은 시스템과 독립적인 테스트 수행 활동을 명시한다.

3단계에서는 기능별 모듈을 관리하는 활동을 명시한다. 그에 따라 형상관리를 수행하도록 하며 이때 부수적으로 형상 관리 도구를 함께 사용하도록 하는 활동을 명시한다.

(표 3) 구현 프로세스 활동

구현 프로세스		
산출물		템플릿소스, 기능단위 모듈 소스, 전체 소스
1 단계	목표	기능단위 모듈 소스를 개발한다.
	활동	· 기능별 릴리즈를 통한 반복 개발을 한다. · 모바일 임베디드 시스템에 독립적 수행이 가능한 코드로 작성한다.
2 단계	목표	테스트 주도로 개발 한다.
	활동	· 모바일 임베디드 시스템 에뮬레이터 환경을 구축한다. · 매일 매일 지속적 통합을 한다. · 모바일 임베디드 시스템과 독립적 테스트를 수행한다.
3 단계	목표	기능별 모듈을 관리한다.
	활동	· 2명이 한 조로 짝 프로그래밍을 한다. · 리팩토링으로 프로그래밍 한다. · 형상 관리한다.

라. 테스트 프로세스

표 4는 테스트 프로세스로 고객이 직접 확인할 수 있는 활동으로 유도한다. 그에 따라 고객이 작성한 테스트 케이스 카드나 혹은 유저스토리를 바탕으로 테스트케이스 카드를 작성하며 정량적 관리 혹은 예측 관리하기 위해 버그 리포트 산출물을 생성한다.

1단계에서는 기본 단계인 모듈별로 개발된 소스를 통합 테스트 수행 활동을 명시하며 유저스토리카드 혹은 테스트 케이스 카드를 바탕으로 고객이 인수하여 테스트하는 활동을 명시한다.

2단계에서는 결함을 관리하는 활동을 명시한다. 이에 적합한 활동으로 단위 테스트 수행과, 버그 리포트 작성이 된다.

3단계에서는 요구사항 명세서로부터 결함을 미리 예측할 수 있는 활동을 명시한다. 요구사항 추‘적 데이터와 테스트 결과를 비교 분석하는 활동을 명시하며, 아키텍처 설계 프로세스에서 나오는 설계서로부터 시스템 통합 테스트 수행할 수 있는 활동을 명시한다.

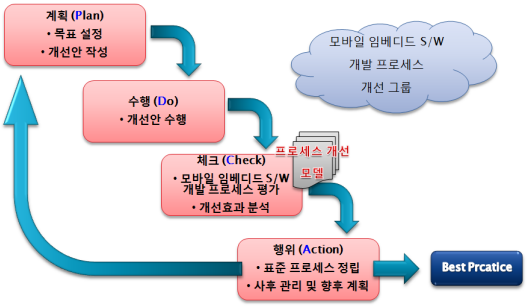
(표 4) 테스트 프로세스 활동

테스트 프로세스		
산출물		테스트 케이스 카드, 버그리포트
1 단계	목표	모바일 임베디드 소프트웨어 테스트를 수행한다.
	활동	· 모바일 임베디드 소프트웨어 통합 테스트를 수행한다. · 유저 스토리카드를 바탕으로 인수 테스트를 수행한다.
2 단계	목표	모바일 임베디드 소프트웨어 테스트 결함을 관리한다.
	활동	· 단위 테스트를 수행한다. · 고객이 직접 테스트케이스 카드를 작성한다. · 버그 리포트를 작성한다.
3 단계	목표	요구사항 명세서로부터 결함을 예측한다.
	활동	· 요구사항 추적 데이터와 테스트 결과를 비교 분석한다. · 모바일 임베디드 아키텍처 설계서로부터 시스템 통합 테스트를 수행한다.

3.3.2 중간 프로세스 모델

4개의 프로세스에 대한 개선을 위한 연관성 적용은 그림 9와 같다. 조직에서 개선을 위한 그룹을 구성하고, 구성원은 개발자 혹은 관리자로 구성한다.

개선 흐름은 데밍의 PDCA(Plan-Do-Check-Action)을 통한 연관성을 정의하며 계획, 수행, 체크, 행위 단계로 적용한다.



(그림 9) 프로세스 연관성 적용 프로세스

- 계획 단계(Plan) - 개선하고자하는 프로세스의 목표를 설정하고 개선안을 작성한다. 개선안 작성 시 개발 프로세스 개선 모델을 참고하여 활동을 제시한다.
- 수행 단계(Do) - 프로젝트 수행 시 개선안을 바탕으로 개발을 진행한다.
- 체크 단계(Check) - 수행 단계에서 진행된 산출물이나 메트릭을 바탕으로 개발 프로세스를 평가한다. 이때 평가 시 개발 프로세스 개선 모델의 활동을 기준으로 올바르게 수행하고 있는지 검토한다. 대부분이 수작업으로 문서를 검토하고, 개발자와 관리자의 인터뷰를 통해 프로세스 활동을 판단하게 되지만 이를 도와줄 수 있는 도구가 있다면 활용해야 한다. 이런 지원 도구는 다음 장에서 구현하도록 한다.
- 행위 단계(Action) - 개선을 통해 조직의 정립된 프로세스가 나오게 되며 이는 프로젝트 수행 시 조직에 활동이 내재화 되도록 관리자들이 유도해야 한다. 또한 가장 잘 수행된 프로세스를 Best Practice로 뽑아낼 수 있다.

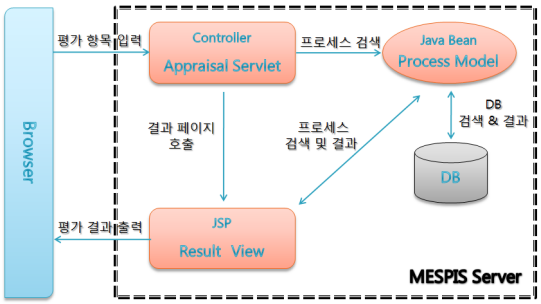
4. 프레임워크 구현 및 평가

4.1 설계

본 논문에서 제안하는 개선 프레임워크를 바탕으로 모바일 임베디드 소프트웨어 개발 프로세스 개선을 지원할 수 있는 프레임워크인 MESPIS(Mobile Embedded Software Process Improvement System)를 구현하였다.

이 MESPIS는 프로세스 개선에 있어서 개선 그룹이 산출물과 스크립트 문서로만 평가하는 것보다 쉽게 각 프로세스를 평가할 수 있도록 도와준다. 뿐만 아니라 개발 조직에서 자체적으로 프로세스 평가를 위해 사용될 수 있다. 따라서 MESPIS의 사용자는 개선 그룹의 임원들뿐만 아니라 프로젝트 관리자, 프로세스 관리자가 될 수 있으며, 최종 보고를 받을 수 있는 조직의 상부까지 포함된다.

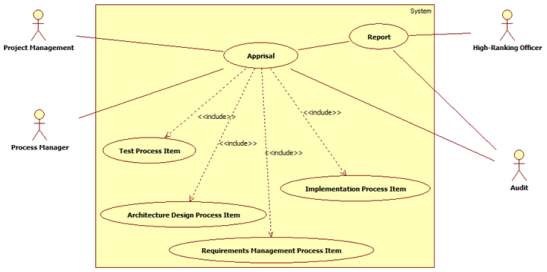
그림 10은 MVC(Model-View-Controller) 패턴 기반으로 간략하게 MESPIS의 시스템 아키텍처를 보여준다. DB에 있는 프로세스 및 활동정보를 Model인 JavaBean에서 참조하고, 이를 사용자가 브라우저로 통해 평가요청을 하면 컨트롤러에서 평가 수행한 후 결과 페이지를 보여주는 시스템이다.



(그림 10) MESPIS 프레임워크 시스템 아키텍처

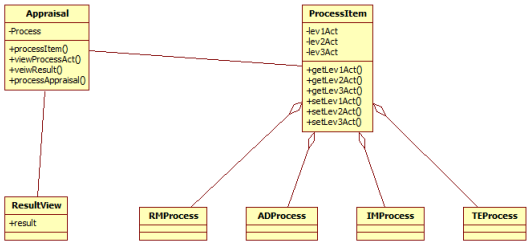
MESPIS가 제공하는 기능으로는 제안한 프레임워크의 4 프로세스 영역에 대한 평가와 이를 개선하기 위한 보고서 형태로 제시하는 기능이다. 이

에 MESPIS에 대한 유스케이스는 간단하게 그림 11과 같이 나타낼 수 있다.



(그림 11) MESPIS의 유스케이스

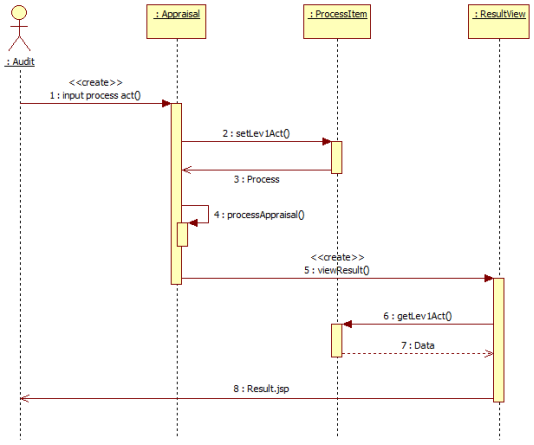
그림 12는 MESPIS의 클래스 다이어그램으로 MESPIS의 2가지 기능에서 상부에 보고되는 Report기능은 4 프로세스 영역에 대한 활동을 체크한 후 결과 페이지로 뽑아내기 때문에 따로 클래스를 두지 않는다.



(그림 12) MESPIS의 클래스 다이어그램

다른 한 가지 4 프로세스 영역에 대한 활동을 Java Bean모델로 구성하여 MVC 패턴 기반으로 평가 클래스에서 각 4 프로세스 Java Bean모델을 사용하도록 클래스를 구성하였다.

그림 13은 MESPIS 시퀀스 다이어그램으로 컨트롤러 역할을 수행하는 Appraisal 클래스와 4 프로세스의 활동을 속성으로 가지는 Java Bean 모델 사이에서 활동에 대한 평가를 Appraisal 클래스에서 계산하여 Result.jsp 페이지를 통해 웹 브라우저에 표시한다.



(그림 13) MESPIS 시퀀스 다이어그램

4.2 구현

MESPIS를 구현하기 위한 개발환경에서는 StarUML을 사용하여 UML 2.0 표기법으로 앞서 소개한 설계하여, MVC 패턴을 적용하기 위해 Java 1.6과 JSP로 작성하였다. 구동 서버환경으로는 Windwos 2008 기반에 J2EE 플랫폼을 설치하여 DB는 Oracle 9i를 사용하고, 서블릿 작동을 위해 Apache Tomcat 6.0으로 구동하였다.

그림 14는 MSEPIS의 실행 및 결과 화면이다. 모바일 임베디드 소프트웨어 개발 조직에서 개발 프로세스를 개선하기 전에 미리 MESPIS 도구를 통해 조직의 성숙도를 살펴볼 수 있으며, 개발 프로세스 개선 그룹에서는 각 프로세스 영역에 대한 코멘트를 적용하여 조직의 개선활동에 도움을 준다. 조직의 각 활동에 대한 점수를 프로젝트 수행에 있어서 산출물이나 인터뷰를 통한 증거로써 점수를 부여하고, 활동에 대한 의견 첨부는 활동의 개선 요구사항을 코멘트로 작성한다. 그에 따른 프로세스 평가를 수행하면 결과를 볼 수 있다.

(그림 14) MESPIS 평가 실행 및 결과 화면

4.3 CMMI 프로세스 커버리지 평가

모바일 임베디드 소프트웨어 개발 조직에서 프로세스 정립과 CMMI 수준을 달성을 위한 CMMI 프로세스 영역에 대한 커버리지는 표 5와 같다.

(표 5) CMMI 프로세스 커버리지

CMMI 주요 프로세스	모바일 임베디드 S/W 개발 프로세스 커버리지
Casual Analysis and Resolution	부분지원
Configuration Management	지원
Decision Analysis and Resolution	부분지원
Integrated Project Management + IPPD	지원
Measurement and Analysis	지원
Organizational Innovation and Deployment	미지원
Organizational Process Definition+IPPD	미지원
Organizational Process Focus	미지원
Organizational Process Performance	미지원
Organizational Training	미지원
Product Integration	지원
Project Monitoring and Control	지원
Project Planning	지원
Process and Product Quality Assurance	지원
Quantitative Project Management	지원
Requirements Development	지원
Requirements Management	지원
Risk Management	부분지원
Supplier Agreement Management	부분지원
Technical Solution	지원
Validation	지원
Verification	지원

제안한 프레임워크를 검증하기 위한 방법으로 CMMI 프로세스 영역의 활동이 모바일 임베디드 소프트웨어 개발 프로세스 활동에서 명시하고 있는지를 평가한다. 임베디드 개발 프로세스, Agile 방법론, XP 위주의 CMMI 활동과 결합하여 만들어진 개선 프로세스 모델이기에 각 프로세스 영역마다 지원하는 부분이 다르며 조직 관리에 해당하는 프로세스 영역부분을 모두 지원하고 있지 않다. 이는 개발 위주의 활동으로 개발 프로세스 개선 모델을 제안하였기 때문이다. 따라서 CMMI의 모든 수준은 달성할 수 없다.

(표 6) 프로세스 개선 모델과의 비교를 통한 평가

분류	Oktaba의 연구[8]	Kuvaja의 연구[17]	KAIST의 연구[16]	본 연구
적용 목표	중소규모 조직 소프트웨어 프로세스 모델 개발	중소규모 조직 소프트웨어 프로세스 평가 모델 개발	중소규모 조직 소프트웨어 프로세스 모델 개발	모바일 임베디드 S/W 개발 프로세스 개선 프레임워크 개발
활용 모델	· SW-CMM · ISO9000-2000 · PMBoK	· BOOTSTRAP	· CMMI · SPICE · SW 사업 관리감독에 관한 세부 지침	· CMMI · Agile 방법론 · XP · 모바일 임베디드 S/W 개발 프로세스
개발 기준	전문가	전문가	연구사례 및 전문가	연구 사례
타 모델 전환 가능성	어려움	가능	가능	가능

표 6은 기존의 여러 프로세스 개선 모델 연구와 비교를 통한 평가이다]. 4가지 개선 모델 모두 타 모델 전환 가능하지만 3가지 개선 모델의 적용 범위는 모두 중소규모 조직으로 본 논문에서 제안한 개선 프레임워크의 프로세스 모델은 중소규모 조직을 위한 중간 모델을 추출한 다음 모바일

임베디드 소프트웨어 개발 조직에 맞는 활동으로 좀 더 세부적인 세부 명세를 함으로써 그 차이점을 두고 있다.

5. 결론 및 향후 연구

Agile 방법론은 개발 프로세스보다 개발 활동에 더 중점을 두어 소프트웨어 품질향상을 꾀하고 있으며, 가장 대표적인 방법론인 XP는 반복 개발 및 고객과의 소통을 필두로 모바일 임베디드 소프트웨어 개발 조직에 적용하기 적합하다.

본 논문에서는 모바일 임베디드 소프트웨어 개발 조직에서 개발 프로세스를 개선하기 위한 프레임워크를 제안하였다. XP의 활동을 바탕으로 모바일 임베디드 개발 프로세스에 적합한 활동으로 재구성하고, 더불어 최종 목표인 CMMI 수준 2 또는 3을 달성하기 위한 활동으로 재구성하였다. 재구성 방법은 개선 프로세스를 식별하고 주요 프로세스 활동을 정의하였으며, 개선 프레임워크를 적용하기 위한 흐름을 제시하였다.

CMMI 프로세스 커버리지 확인을 통해 CMMI 수준 3 달성이 가능하였고, 이로 인한 모바일 임베디드 소프트웨어 개발 프로세스의 정량적으로 관리 및 신뢰성 있는 모바일 임베디드 소프트웨어 개발 지원을 기대할 수 있다. 또한 프로세스 개선 모델과 비교하였을 때 기존의 소프트웨어 프로세스 개선 모델의 범위에서 모바일 임베디드를 지원할 수 있다는 장점이 있다.

향후 연구로는 제안한 프레임워크를 실제 프로젝트에 적용해보고 이에 대한 정량적인 효과 분석과 지원도구 MESPIS에 대한 기능 보강이 필요하다.

참 고 문 헌

[1] 박한솔, 김문희, “임베디드 소프트웨어 기술 동향”, IITA 기술정책정보호단, 7, 2006.

[2] 김효영, 한혁수, “임베디드 시스템 품질개선을 위한 CMMI의 적용”, 정보과학회지,제22권 제6호, pp.50-57, 5, 2004.

[3] 라영호, “임베디드 시스템 개발전략”, 경영과 컴퓨터, 3, 2007.

[4] Bas Graaf, Marco Lormans, and Hans Toetenel, “Embedded Software Engineering: The State of the Practice”, IEEE SOFTWARE, Vol. 20, No. 6, pp.61-69, 2003.

[5] Carnegie Mellon SEI, “CMMI for Development, Version 1.2”, CMU/SEI-2006- TR-008, 2006.

[6] Sung Wook Lee, Haeng Kon Kim and Roger Y. Lee, “Enterprise Process Model for Extreme Programming with CMMI Framework”, ICIS2008, pp.169-180, 5, 2008.

[7] Bob Jarvis and Stephen.P.Gristock, “Extreme Programming (XP) SixSigma CMMI”, http://www.sei.cmu.edu/cmmi/presentations/sepg05.presentations/jarvis-gristock.pdf, 2005.

[8] Software Quality Institute, “CMMI-Staged Representation Mapping to ISO 15504-2”, Defence Materiel Organization, 2001.

[9] Kent Beck, Mike Beedle, Arie van Bennekum, et al., “Manifesto for Agile Software Development”, http://www.agilemanifesto.org, 2006.

[10] Barry Boehm, Richard Turner, Balancing Agility and Discipline: A Guide for the Perplexed, Addison-Wesley, 2003.

[11] Don Wells, “Extreme Programming: A gentle introduction”, http://www.extremeprogramming.org, 2, 2006.

[12] Mark C. Paulk, ″Extreme Programming from a CMM Perspective,″ IEEE Software, vol. 18, no. 6, pp. 19-26, Nov/Dec, 2001

[13] 채종진, 윤희병, “임베디드SW 개발을 위한 제품계역 개발방법론의 비교 및 평가”, 한국컴퓨터종합학술대회 논문집, 제34권, 제1호, pp.129-133, 2007.

[14] 민상윤, “임베디드 소프트웨어 개발방법론

적용 전략과 성공 사례", 한국소프트웨어진흥원, 4, 2007.
[15] 한국전자통신연구원, "임베디드 시스템 개발 프레임워크 구축", 정보통신부, 2006.
[16] 한국과학기술원, "SW 프로세스 개선 모델 최종보고서", 한국소프트웨어진흥원, 2006.
[17] Pasi Kuvaja, Jorma Palo and Adrina Bicego, "TAPISTRY-A Software Process Improvement Apporach Tailored for Small Enterprises", Software Quality Journal, Vol.8, 1999.
[20] 서주영, 최병주, "CMMI 기반의 테스트 프로세스 개선 프레임워크와 단계적 개선방안", 정보과학회지, 제23권 제3호, pp.17-26, 3, 2005.

◐ 저 자 소 개 ◑

이 성 욱
2007년 대구가톨릭대학교 컴퓨터공학과 (공학사)
2007년~현재 대구가톨릭대학교 컴퓨터정보통신공학과 석사과정 재학중
관심분야 : SOA, Agile methodologies, MDA
E-mail : selab@cu.ac.kr

김 행 곤
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
1990년~현재 대구가톨릭대학교 컴퓨터공학과 교수
2001년~2002년 Central Michigan University 교환교수
2007년~2008년 미 SEITI 연구소 연구교수
관심분야 : CBSE, 소프트웨어 재공학, CASE, 유지보수 자동화 툴, 사용자 인터페이스, 요구공학 및 도메인 공학
E-mail : hangkon@cu.ac.kr

김 성 원
1972년 단국대학교 전기공학과(공학사)
1982년 단국대학교 대학원 전기공학과(공학석사)
2000년 대구가톨릭대학교 대학원 전자계산통계학과(이학박사)
1979년~1991년 수원과학대학 전자계산과 교수
1991년~1995 안양대학교 컴퓨터공학과 교수
1995년~현재 안양대학교 전기전자공학과 교수
관심분야 : 컴퓨터프로그래밍분야, 소프트웨어공학, 컴퓨터그래픽스응용, 임베디드 소프트웨어
E-mail: swkim@anyang.ac.kr