XP 기반의 모바일·임베디드 소프트웨어 개발 프로세스 개선 프레임워크에 관한 연구

이성욱*, 김행곤*, 정연기**
* 대구가톨릭대학교 컴퓨터정보통신공학부
e-mail:{sojiro, hangkon}@cu.ac.kr
** 경일대학교 컴퓨터공학부
ykchung@kiu.ac.kr

A Study on Improvement Framework for XP-Based Mobile·Embedded Software Development Process

Sung-Wook Lee*, Haeng-Kon Kim*, Youn-Ki, Chung**
* Dept. of Computer Information & Communication Engineering, Catholic University of Daegu, Korea
** School of Computer Engineering, Kyungil University, Korea

요 약

최근 모바일·임베디드 시스템의 품질 향상 노력의 일환으로 하드웨어보다 소프트웨어 개발에 더 관심을 가지며 비중 또한 증가하고 있다. 모바일·임베디드 소프트웨어는 특정 목적을 위해 개발되는 특성으로 인하여 표준화된 개발 프로세스 없이 개발되는 경우가 많다. 또는 기업 자체 개발 방법론으로 개발하거나 알려진 임베디드 소프트웨어 개발방법론으로 개발하기도 한다. 하지만 잦은 요구사항 변경과 시스템의 결함 등 프로젝트의 실패는 줄어들지 않고, 이에 대한 해결책으로 본 논문에서는 모바일·임베디드 개발 프로세스의 개선으로 품질향상이 필요하다고 주장한다.

소프트웨어 프로세스에 있어서 대기업에서는 CMMI 수준 달성을 위한 노력을 꾀하고 있다. 하지만 이 모델은 일반 중소규모 기업에 적용하기에는 부담이 너무 크다는 단점이 있고 비용 또한 만만치 않다. 이에 본 논문에서는 XP의 실천사항을 바탕으로 모바일 임베디드 소프트웨어 개발 프로세스 개선을 위한 프레임워크를 제안한다. 이를 통해 모바일·임베디드 소프트웨어 개발 기업은 제품 품질 향상과 함께 CMMI 수준 달성에 기여·활용하게 된다.

1. 서론

디지털 시대가 급속도로 진행되면서 과거 특정 산업용 기기 제어를 위해 사용되던 임베디드 시스템은 오늘날 많은 디지털 기기의 보급과 함께 널리 쓰이고 있다. 모바일·임베디드 시스템은 다양한 방법론 및 도구로 개발되고 있다. 하지만 모바일·임베디드 시스템은 결함과 잦은 요구사항 변경 등 제품 개발의 생산성과 품질 달성이 쉽지 않다. 이런 조건 속에서도 제품의 품질 향상을 꾀하는 데 있어서 최근 그 배경을 하드웨어 보다 소프트웨어에 비중을 두고 있다[3].

모바일·임베디드 소프트웨어는 특정 목적을 지닌 소프트웨어로 자체 개발 방법론으로 개발하거나 아예 개발 방법론 혹은 프로세스 없이 개발하는 경우도 많다[6]. 때문에 하드웨어 의존성이 높은 모바일·임베디드 소프트웨어 개발과 품질 향상을 위해 COMET(Concurrent Object Modeling and design meThod)과 같은 개발 방법론이 연구되기도 하였다.

"본 연구는 지식경제부 및 정보통신연구진흥원의 대학 IT 연구센터 지원사업의 연구결과로 수행되었음" (IITA-2008-(C1090-0801-0032))

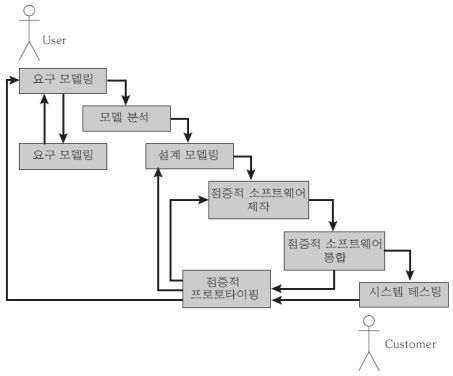
본 논문에서는 방법론이 아닌 개발 프로세스 개선을 통하여 품질향상을 꾀하고자 한다. 이미 CMMI, SPICE와 같은 많은 소프트웨어 개선 프레임워크가 있지만 중소규모 기업에서 적용하기에는 적지 않은 부담이 있다[4]. 하여 프로세스 경량화를 지향하는 애자일 방법론 중 XP(eXtreme Programming)의 실천사항을 바탕으로 모바일·임베디드 개발 프로세스 영역에 제한하는 모바일·임베디드 개발 프로세스 개선 프레임워크(MESPI: Mobile Embedded development Software Process Improvement)를 제안한다.

2. 관련연구

2.1 임베디드 소프트웨어 개발 방법론

모바일·임베디드 소프트웨어 개발은 범용 소프트웨어 개발에 비해 복잡하고 고난도의 기술을 요구할 뿐만 아니라, 개발자의 하드웨어와 소프트웨어에 대한 동시 이해능력을 보유해야만 성공적이며 생산성 있는 제품을 개발할 수 있다. 그에 따라 많은 임베디드 소프트웨어 개발 방법론을 살펴보면 대다수 점진적 개발이 주를 이루고 있다.

개발 방법론의 하나인 COMET(Concurrent Object Modeling and design meThod)의 특징은 하드웨어 및 소

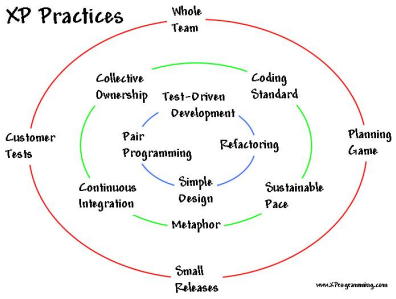


(그림 1) COMET 생명주기도

프트웨어의 동시 설계 위험성을 최소화하기 위하여 그림 1과 같이 폐기형 프로토타이핑 기법과 점증적 프로토타이핑 기법을 함께 적용한다. 그렇게 함으로써 하드웨어와 소프트웨어간의 역할 분담 등 개발 초기 단계에 명확히 하고, 중간 단계에서는 소프트웨어의 기능을 점진적으로 최종 버전까지 반복하여 개발한다[2].

2.2 XP(eXtreme Programming)

XP는 애자일 방법론의 가장 대표 주자로서 개발 방법론이다. CMMI와 달리 애자일 방법론은 프로세스 경량화를 지향하며 그 중 XP는 의사소통, 피드백, 단순성, 용기, 존중 같은 가치들에 바탕을 둔 소프트웨어 철학이다. 원칙과 가치 그리고 그림 2와 같은 실천방법을 제시하며, 그 실천방법에는 짝 프로그래밍, 스토리, 10분 빌드 등 때문에 방법론이나 프로세스가 정립되지 않은 기업에서도 비교적 빨리 적용할 수 있다[1].

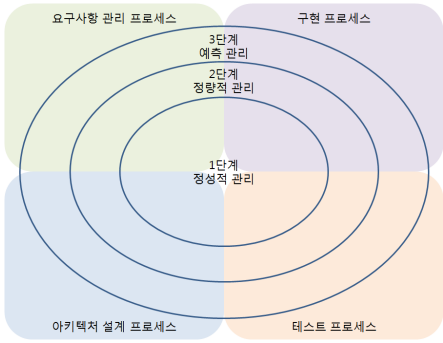


(그림 2) XP의 실천사항

3. 모바일・임베디드 소프트웨어 개발 프로세스 개선 프레임워크

3.1 개요

모바일・임베디드 소프트웨어 개발 프로세스 개선 프레임워크(MESPI(Mobile Embedded development Software Process Improvement) 로 칭함)는 개발 팀의 비정형적인



(그림 3) MES 개발 프로세스 영역

프로세스를 정립하며 보다 신뢰성 있는 제품 개발을 위해 개발 프로세스를 개선하는 프레임워크다. 이 MESPI는 그림 3에서 보듯이 4가지 프로세스 영역에서 성숙도에 따라 나누어진 개선 활동의 집합으로 다음 4가지 특징을 가진다.

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

둘째, MESPI는 개발 팀의 성숙도 수준에 따라 비 관리, 정성적 관리, 정량적 관리, 예측 관리 4단계로 표현한다. 0단계 비 관리는 어떤 프로세스나 개발 방법론이 없는 팀을 나타낸다. 1단계 정성적 관리에서는 일반적으로 MES 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 제시한다.

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

3.2 MESPI 프로세스 영역

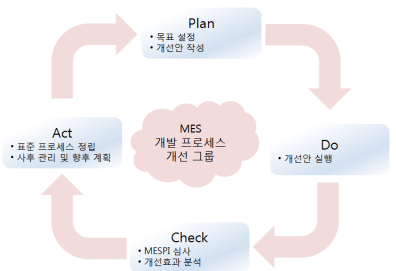
MESPI의 주요 프로세스는 앞서 특징에서 보았듯이 기본적인 4개의 프로세스 영역을 3단계에 걸친 활동으로 표현한다. 각 프로세스마다 산출물은 유저스토리, 요구사항 명세서, 프로토타입, 아키텍처 설계서, 템플릿 소스, 기능 단위 모듈, 전체 소스, 테스트 케이스 카드, 버그 리포트이

다. XP에서는 굳이 문서화는 필요 없다고 이야기하고 있지만 실제 기업에서 핵심 되는 산출물을 XP의 철학에서 벗어나지 않는 범위 내에서 산정하였다. 뿐만 아니라 유저스토리, 테스트 케이스 카드는 XP에서 항상 고객과 소통해야함을 나타내는 항목이다.

표 1은 산출물을 포함한 단계별 목표와 활동을 정리한 것이다. 각 프로세스 별 1단계 정성적 관리에서는 가장 기본적으로 수행해야할 활동을 나타낸다. 모바일・임베디드 소프트웨어 특성에 맞게 XP의 실천 사항이 테일러링 되었다. 2단계 정량적 관리는 각 프로세스마다 제품의 결함을 사전에 발견하기 위한 준비 활동을 포함한다. 예를 들어, 요구사항 관리 프로세스의 “고객의 요구사항 변경사항을 수집, 관리한다.”라는 항목에서 단순히 고객의 요구사항 변경만 받아들이지 않고 변경된 수와 내용을 수집하여 얼마나 많은 변경에 따라 결함이 발생되었는지 측정가능하다. 3단계 예측 관리는 2단계 준비 활동을 기반으로 수집된 데이터를 분석하고 측정하여 결함을 예측하는 활동이 포함된다. 테스트 프로세스에서 “요구사항 추적 데이터와 테스트 결과를 비교 분석한다.”라는 활동은 단위 테스트든 통합 테스트든 결과 값이 설계 이전 요구사항 관리 프로세스에서 요구사항 추적 가능한 데이터와 비교함으로써 좀 더 결함 발생의 원인을 쉽게 찾을 수 있는 활동이 된다. 또한 구현 프로세스에서 “2명이 한 조로 짝 프로그래밍을 한다.”라는 항목은 하드웨어와 소프트웨어가 동시에 개발되어져야하는 모바일・임베디드 시스템의 특성상 소프트웨어 개발에 있어서 테스트 프로세스에서 발견될 결함을 미리 찾게 된다. 모바일・임베디드 소프트웨어 개발에 관련된 활동들의 집합인 MESPI를 각각의 프로세스에서 수행한다면 모바일・임베디드 소프트웨어 개발 기업은 XP의 철학을 따른 활동을 지향하는 모바일 임베디드 소프트웨어 개발 프로세스로 이끌어 준다.

3.3 MESPI 적용 로드맵

MESPI를 적용함에 있어서 PDCA 기법을 활용한다. 평가 문서로부터 개발 프로세스의 문제점을 인지하고 개선안을 작성하여 개선안을 실행하고, 최종적으로 기업 내 프로세스 정립을 목표로 한다. 다음 그림 4는 MESPI 적용하기 위한 로드맵이다.



(그림 4) MESPI 적용 로드맵

<표 1> 프로세스 별 산출물, 목표, 활동

요구사항 관리 프로세스		
산출물		유저스토리, 요구사항명세서
1 단계	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.
2 단계	목표	요구사항 변경을 관리한다.
	활동	・고객의 요구사항 변경사항을 수집, 관리한다. ・문제 발생에 대한 대안을 마련한다.
3 단계	목표	요구사항 추적성을 확보하고 요구사항을 검증한다.
	활동	・요구사항 추적성을 확보하기 위한 변경 사항 자료를 분석한다. ・모바일 임베디드 소프트웨어가 탑재될 하드웨어 제약사항을 검토하고, 요구사항명세서를 검증한다.
아키텍처 설계 프로세스		
산출물		프로토타입, 아키텍처 설계서
1 단계	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.
2 단계	목표	모바일 임베디드 소프트웨어 아키텍처를 관리한다.
	활동	・모바일 임베디드 소프트웨어 아키텍처와 하드웨어 제약사항을 참고하여 평가, 검증한다.
3 단계	목표	모바일 임베디드 소프트웨어 아키텍처를 모델링한다.
	활동	・템플릿 소스작성이 가능한 모델링을 한다. ・모바일 임베디드 시스템 아키텍처로 확장한다.
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.
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.

모바일 임베디드 개발 프로세스를 개선하기 위한 조직을 구성한다. 조직 구성은 기존의 개발 팀 구성원을 포함하여 운영진프로세스 개선 그룹을 결성한다. 대기업이 아닌 이상 대부분의 모바일·임베디드 소프트웨어 개발 기업은 인력이 충분치 않다. 때문에 조직 내에서 기존의 인력으로 MES 개발 프로세스 개선 조직을 구성한다. 이 조직은 다음 단계로 수행한다.

- 계획 – 각각 프로세스영역별로 목표를 설정하여 개발 프로세스 개선안을 작성한다. 처음 MESPI를 접하는 기업이라면 프로세스 활동 문서를 참고하여 개선안을 작성하도록 한다.
- 수행 – Plan에서 작성된 개선안을 바탕으로 프로젝트가 수행될 시 개선도 함께 수행한다.
- 체크 – MESPI 프로세스 영역 문서를 참고하여 각각의 활동에 따른 수행을 측정한다. 측정 결과에 따라 이전 프로젝트 대비 개선효과를 분석한다.
- 행위 – Check에서 개선효과 분석 자료를 바탕으로 기업 내 표준 개발 프로세스로 정립하고 사후 프로세스 테일러링과 교육 관련하여 향후 계획을 수립한다.

4. 평가

본 논문에서 제안한 MESPI의 평가를 위해 표2와 같이 기존의 CMMI 표현 방법 중에서 프로세스 관리, 프로젝트 관리, 엔지니어링, 지원의 4가지 영역으로 나타내는데 있어서 주요 프로세스 영역을 어느 정도 커버하는지를 살펴본다[5].

요구사항, 아키텍처 설계, 구현, 테스트 프로세스를 초점으로 하는 활동이기에 MESPI는 CMMI의 엔지니어링 부분을 모두 지원한다. 반면 모바일·임베디드 소프트웨어 개발 프로세스에 초점을 맞추었기 때문에 프로세스 관리 영역은 지원하지 않는다. 그 외 프로세스 관리, 지원 영역은 MESPI의 활동이 부분적으로 커버하고 있다.

이처럼 모바일·임베디드 소프트웨어를 개발하는 중소규모 기업에서는 프로세스가 무거운 CMMI를 바로 도입할 것이 아니라 모바일·임베디드 소프트웨어 특성상 제안하는 MESPI를 통해 어느 정도 CMMI 수준에 도달할 수 있는 능력을 보여준다.

5. 결론 및 향후 연구

본 논문에서는 모바일·임베디드 소프트웨어 개발 프로세스를 개선을 위한 프레임워크를 제안하였다. 이 프레임워크는 기존의 소프트웨어 개선 모델인 CMMI가 중소규모 기업에 적용하기에 부담이 있다는 점을 감안하여 프로세스 경량화에 초점을 둔 XP의 실천사항을 바탕으로 모바일·임베디드 소프트웨어 개발 기업의 개발 프로세스를 개선한다. PDCA 기법을 적용하여 개발 프로세스의 문제점을 인지하고 개선안을 작성, 실행한 프로세스는

<표 2> MESPI의 커버리지

CMMI 주요 프로세스	영역	MES 개발 프로세스 개선 프레임워크
Casual Analysis and Resolution	Support	부분지원
Configuration Management	Support	지원
Decision Analysis and Resolution	Support	부분지원
Integrated Project Management + IPPD	Project Mngt	지원
Measurement and Analysis	Support	지원
Organizational Innovation and Deployment	Process Mngt	미지원
Organizational Process Definition+IPPD	Process Mngt	미지원
Organizational Process Focus	Process Mngt	미지원
Organizational Process Performance	Process Mngt	미지원
Organizational Training	Process Mngt	미지원
Product Integration	Engineering	지원
Project Monitoring and Control	Project Mngt	지원
Project Planning	Project Mngt	지원
Process and Product Quality Assurance	Support	지원
Quantitiative Project Management	Project Mngt	지원
Requirements Development	Engineering	지원
Requirements Management	Engineering	지원
Risk Management	Project Mngt	부분지원
Supplier Agreement Management	Project Mngt	부분지원
Technical Solution	Project Mngt	지원
Validation	Engineering	지원
Verification	Engineering	지원

CMMI의 모든 프로세스 영역을 커버할 수 없지만 모바일·임베디드 소프트웨어 특성에 따라 일정 프로세스 영역을 커버할 수 있다.

MESPI의 평가 지원을 위한 도구를 개발하여 모바일·임베디드 소프트웨어 개발 기업에서 좀 더 쉽게 프로세스 개선에 접하게 하고, 시스템에서 수집되는 데이터를 바탕으로 모바일 임베디드 소프트웨어 개발 프로세스의 정립 및 표준화를 위한 연구가 향후 요구된다.

참고문헌

[1] Don Wells, "Extreme Programming: A gentle introduction", http://www.extremeprogramming.org/, February 2006.

[2] 한국정보통신인력개발, "ESDP 표준교재", 사이텍미디어, 2004.

[3] 김효영, 한혁수, "임베디드 시스템 품질개선을 위한 CMMI의 적용", 한국정보과학회, 정보과학회지 제22권 제6호, pp.50-57, 2004.

[4] Sung-Wook Lee, Haeng-Kon Kim, Roger Y. Lee, "Enterprise Process Model for Extreme Programming with CMMI", ICIS2008, May 2008.

[5] Software Engineering Institute, "Capability Maturity Model Integration(CMMI) Version 1.1", Carnegie Mellon University, August 2002.

[6] 민상윤, "임베디드 소프트웨어 개발방법론 적용 전략과 성공 사례", 한국소프트웨어진흥원, April 2007.