CMMI 기반의 XP를 위한 형상 관리 프로세스 구축 지침☆

Guidelines for Implementing Configuration Management in Extreme Programming based on CMMI

한 동 준* 한 혁 수**
Dong Joon Han Hyuk-Soo Han

요 약

Agile 소프트웨어 개발의 대표적인 방법론인 XP(eXtreme Programming)에서는 현장에서 바로 활용할 수 있는 아주 기본적인 프로세스만을 정의하고, 개발 그 자체에 집중하여 효율성을 극대화한다. 그러나 XP의 실천사항들은 개발에 집중되어 있어 개발 산출물에 대한 관리는 간과되기 쉬우며, 또한 관리적인 부분에 대한 지침이나 연구는 그 중요성에 비하여 미흡한 실정이다. 개발 과정 중 합의된 산출물의 변경과정에 대한 절차나, 지속적인 통합과 리팩토링을 위한 적절한 절차도 정의되어 있지 않아 산출물의 무결성을 보장하기가 어렵다. 이러한 문제점을 극복하기 위해서는 주요 산출물들에 대한 형상관리가 요구된다.

XP에 형상 관리를 적용하기 위해서는 CMMI의 형상 관리 프로세스 영역을 참조하는 것이 바람직하다. CMMI는 형상 관리의 요건을 정의하고, 그 구현은 환경에 맞게 조정하는 것을 권장하기 때문에, 각 개발방식의 특성에 맞추어 적용할 수 있다. XP 방법론을 채택한 조직들에게 XP의 특성을 살리면서 CMMI에 기반한 형상 관리를 수행할 수 있는 지침이 제공 된다면, XP의 지속적인 통합과 리팩토링, 짧은 릴리즈 주기와 같은 기민함을 유지하면서 산출물들의 무결성도 보장 할 수 있을 것이다.

본 연구에서는 CMMI를 기반으로 XP에서 적용될 수 있는 형상 관리 프로세스 요소를 추출하고, 그 요소들을 기반으로 XP의 형상 관리 활동을 위한 적용 지침을 개발하였다.

Abstract

The XP, the representative methodology of Agile software development, maximizes the effectiveness of the development by focusing on development itself and using primitive and basic process definition that can be easily implemented in the fields. However, the most of XP's practices came from those of engineering and the management practices of work products tend to be overlooked. The research on the implementation of those management practices has not been performed enough. Because of deficiency of processes that guide the change control over major baselines of work products and that describe proper continuous integration and refactoring in XP, the integrity of those work products is difficult to be guaranteed. To fulfill this work product integrity, CM(configuration management) should be hired and CMMI(Capability Maturity Model Integration) is considered to be the best references for that purpose. CMMI defines the required practices of CM and leave implementation details to the organization so that it could customize those practices based on the characteristics of its development methods.

The CM process implementation guidelines based on CMMI could provides work product integrity with a way of keeping XP's agility that includes continuous integration, refactoring and small release. In this research, we selected CM process factors applicable to XP among CMMI's CM practices and based on them we developed the CM implementation guidelines.

☞ keyword : CMMI(Capability Maturity Model Integration), XP(eXtreme Programming), 형상 관리(Configuration Management)

1. 서 론

XP는 프로세스와 문서 표준 등의 제반 여건 확립 보다는 현장에서 바로 활용할 수 있는 아주 기본적인 프로세스만을 정의하고, 개발 그 자체에

* 정 회 원 : 상명대학교 일반대학원 컴퓨터과학과
lexia@smu.ac.kr
** 정 회 원 : 상명대학교 소프트웨어학부 교수
hshan@smu.ac.kr(교신저자)
[2007/09/05 투고 - 2007/09/12 심사 - 2008/02/04 심사완료]

☆ 본 연구는 상명대학교 소프트웨어 · 미디어 연구소의 지원으로 수행되었음

집중하는 기민한 개발 방법이다. XP는 고객의 요구사항은 변할 것이라 가정하기 때문에 점진적이고 진화적으로 개발하며, 고객은 개발팀에 상주해야 하고 관리자는 개발자의 앞에 놓인 장애물을 제거하여 개발자가 개발에만 집중할 수 있도록 한다. 또한 사용하지 않는 문서는 작성하지 말라고 주장한다. 그러나 이러한 XP에서도 주요 개발 산출물들의 작성을 요구하기 때문에, 이러한 산출물에 대해서는 적절한 관리가 필요하다. 반면 XP의 실천사항들은 개발에 집중되어 있어 개발 산출물에 대한 관리는 간과되기 쉬우며, 또한 관리적인 부분에 대한 지침이나 연구는 그 중요성에 비하여 미흡한 실정이다. 개발 과정 중 합의된 산출물의 변경과정에 대한 절차나, 지속적인 통합과 리팩토링을 위한 적절한 절차도 정의되어 있지 않아 산출물의 무결성을 보장하기가 어렵다.

XP에서 산출물의 무결성을 위한 관리를 위해서는 형상 관리의 적용이 필요하다.

형상 관리는 프로젝트 기간 동안 제품의 무결성과 추적성을 확보하기 위한 활동이다. 형상 관리의 대상이 되는 항목을 식별하여 변경을 통제하고, 변경 처리 상태를 기록하고 감사하는 활동을 수행한다.

이를 위해서는 현재 세계적으로 가장 많이 사용되고 있는 미국 카네기 멜론 대학에서 개발된 모델인 CMMI(Capability Maturity Model Integration)의 형상 관리 프로세스 영역을 참조하는 것이 바람직하다. CMMI는 베스트 프랙티스들에 기반을 두고 개발된, 성숙도 높은 소프트웨어 개발 조직이 갖추어야 할 22개의 프로세스 영역을 정의한 모델이다. CMMI는 각 프로세스를 구현하기 위한 요건만을 정의하고, 구현은 조직의 환경에 맞게 조정하는 것을 권장하기 때문에, 각 개발조직은 각각의 특성에 맞추어 CMMI의 프로세스들을 적용할 수 있다.

XP 방법론을 채택한 조직들에게도 XP의 특성을 살리면서 동시에 CMMI에 기반한 형상 관리를 수행할 수 있는 지침이 제공 된다면, XP의 지속적인 통합과 리팩토링, 짧은 릴리즈 주기와 같은 기민함을 유지하면서도 산출물들의 무결성을 보장 할 수 있을 것이다[3][4].

하지만 실제로 XP에 적용하기 위한 형상 관리 프로세스의 구현에 관한 지침이나 연구는 많이 이루어 지지 않고 있다.

본 논문에서는 XP의 개발 활동과 CMMI의 형상 관리 활동들의 매핑을 통해 XP에 적용가능한 CMMI 기반의 형상 관리 프로세스 요소를 추출하였다. 이를 바탕으로 XP에 따라 소프트웨어를 개발하는 조직에게 적합한 형상관리 구현 지침을 연구, 개발하였다.

2. 관련 연구

2.1 XP(eXtreme Programming)

2001년도에 탄생한 애자일 동맹은 프로젝트 수행 시의 계속 늘어가는 프로세스의 폐단을 극복하고자 개인의 상호작용 우선, 동작하는 소프트웨어 우선, 고객과의 협력 우선, 변화에 대한 반응 우선에 가치를 부여하는 애자일 동맹 선언을 발표하였다. '가볍지만 충분한' 특징으로 대표되는 애자일 방법론의 대표적인 프로세스로는 SCRUM, Crystal, ADP 등이 있으며, XP 역시 대표적인 애자일 방법론이다[1].

XP는 1990년대에 Kent Beck에 의해 창시되었으며 단순성(Simplicity), 의사소통(Communication), 피드백(Feedback), 용기(Courage), 존중(Respect)의 5가지 가치와 13가지의 실천사항(Practice)을 가지고 있다[2].

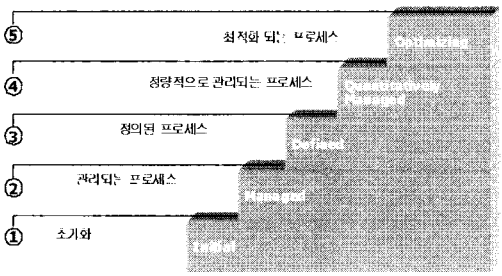
그 중 XP의 기민함을 나타내는 실천사항으로는 다음과 같은 것들이 있다.

- 짧은 릴리즈 주기(Small Release): 고객과의 의사소통을 통한 빠른 피드백의 이용 및 프로젝트의 방향 이탈을 방지하기 위해 작은 시스템을 먼저 만들고 짧은 단위로 릴리즈

- 리팩토링(Refactoring): 개발 기간 동안 내부 코드 구조 개선
- 지속적인 통합(Continuous Integration): 매일 여러 번 소프트웨어를 통합하고 빌드

2.2 CMMI

CMMI는 1993년 SEI에서 개발한 SW-CMM을 기본으로 2000년 8월에 개발되었다[5]. CMMI는 베스트 프랙티스들에 기반을 두고 개발된, 성숙도 높은 소프트웨어 개발 조직이 갖추어야 할 22개의 프로세스 영역을 정의한 모델이다. CMMI의 단계적 표현은 이 22개의 프로세스 영역들을 1부터 5까지의 5단계의 성숙도 수준으로 구성하고 있다.



(그림 1) CMMI 성숙도 수준

본 연구의 주제인 형상 관리 프로세스 영역은 CMMI의 성숙도 수준 2에 속해있다.

2.2.1 형상 관리 프로세스 영역

IEEE 표준에서의 형상 관리는 형상 항목을 식별하여 그 기능적 물리적 특성을 문서화하고, 그러한 특성에 대한 변경을 제어하고, 변경 처리 상태를 기록 및 보고하고, 명시된 요구사항에 부합하는지 확인하는 기술적이고 관리적인 활동으로 정의된다[12].

CMMI 형상 관리 프로세스 영역의 목적 역시 IEEE 표준과 같이 형상 항목 식별, 형상 통제, 형상 상태 보고, 형상 감사를 통해 작업 산출물의 무결성을 수립하고 유지하여 개발 과정 중에 생성되는 주요 산출물들에 대하여 체계적으로 관리하는 것이다.

CMMI의 각 프로세스 영역은 프로세스의 활동을 나타내는 SG(Specific Goal), SP(Specific Practice)와 내재화를 위한 GG(Generic Goal), GP(Generic Practice)로 구성되어 있는데, 형상 관리 프로세스 영역을 위한 SG, SP를 정리해 보면 다음과 같다[5].

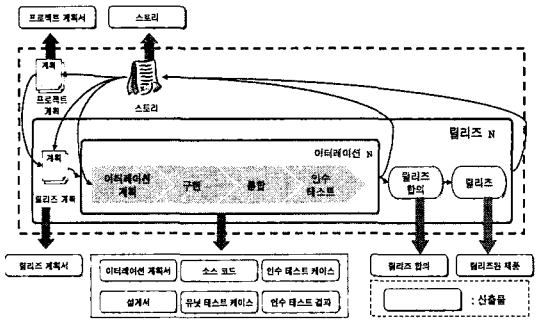
- 특정 시점에서 베이스라인을 구성하도록 선정한 작업 산출물의 형상을 식별
- 형상 항목의 변경을 통제
- 형상 관리 시스템으로부터 작업 산출 명세서를 작성 또는 제공
- 베이스라인의 무결성 유지
- 개발자, 최종 사용자, 고객에게 정확한 형상 상태 및 현재의 형상 데이터를 제공

SG와 SP, GG와 GP의 전체 목록은 다음의 (표 1)과 같다.

(표 1) 형상 관리 프로세스 영역

구분	내용
SG 1	베이스라인 설정
SP 1.1	형상 항목 식별
SP 1.2	형상 관리 시스템 구축
SP 1.3	베이스라인 생성 및 릴리즈
SG 2	형상 관리하의 작업 산출물 변경 추적 및 통제
SP 2.1	변경 요청 추적
SP 2.2	형상 항목 통제
SG 3	베이스라인의 무결성을 수립 및 유지
SP 3.1	형상 관리 기록 작성
SP 3.2	형상 감사 수행
GG 2	관리된 프로세스의 내재화
GP 2.1	조직 정책의 수립
GP 2.2	프로세스 계획

GP 2.3	자원의 제공
GP 2.4	책임 할당
GP 2.5	교육 실시
GP 2.6	형상 관리
GP 2.7	이해관계자의 식별 및 참여
GP 2.8	프로세스 모니터링 및 통제
GP 2.9	객관적 평가
GP 2.10	상위 관리자와의 검토



(그림 2) XP 개발 활동의 산출물

3. XP의 개발 프로세스와 산출물 관리

앞에서 설명한 바와 같이 XP는 가벼운 프로세스를 지향하는 방법론이다. (그림 2)는 XP의 개발 단계들과 각 단계에서의 개발 산출물을 나타내고 있다. XP가 아무리 개발 자체에 무게를 둔다고 해도, 스토리, 소스 코드, 테스트 케이스 등의 문서는 생성이 되고, 이러한 문서들은 지속적인 통합, 리팩토링을 위해 반드시 적절한 관리가 이루어져야 한다. 그러나 적절한 관리가 이루어지지 않으면 다음과 같은 문제점이 발생할 수 있다.

- 릴리즈에서 합의된 개발 산출물의 변경에 대한 통제 프로세스가 존재하지 않아 개발 산출물의 무결성 확보가 어려움
- 고객의 요구사항 변경에 대한 기록이 유지되지 않아 변경에 대한 추적이 어려움
- 구현 후 지속적인 통합을 위한 소스 코드 버전 관리에 대한 프로세스가 존재하지 않아 소스 코드 관리가 어려움
- 구현 후 리팩토링을 위한 소스 코드 및 유닛 테스트 케이스(코드) 버전 관리에 대한 프로세스가 존재하지 않아 소스 코드 및 유닛 테스트 케이스(코드)의 관리가 어려움

이러한 문제점을 해결하기 위해서는 개발 산출물에 대하여 적절한 변경 관리가 필요하다, 예를 들면, 소스 코드는 XP의 특성상 항상 변경이 요구되기 때문에, 동시 수정, 부주의한 코드 손실 등을 막기 위해 변경 절차의 정의가 필요하다. 이러한 절차는 변경에 대하여 고객과 개발팀의 합의 단계, 변경에 관한 기록단계, 변경 결과에 대한 확인 등을 포함하고 있어야 한다. 이러한 활동은 CMMI가 정의하는 형상 관리 프로세스 영역을 구현함으로써 이루어 질 수 있다.

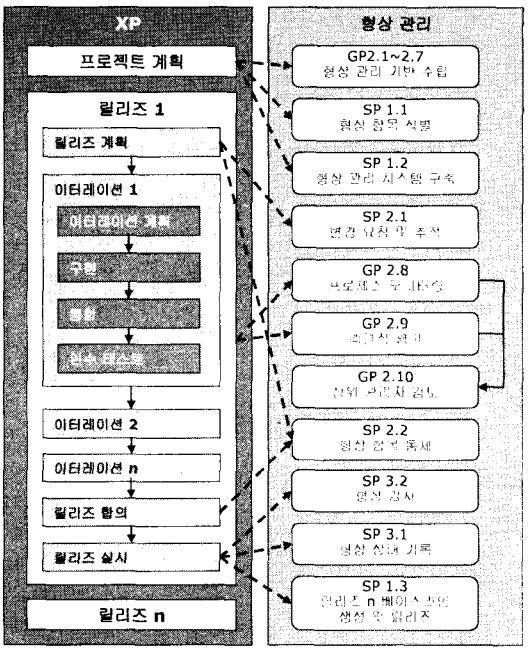
그러나 이러한 구현은 XP의 특성을 반영하여 이루어져야 한다. 즉, XP는 개발자가 개발 활동에만 몰두하게 하여 최대의 효과를 내도록 하는 개발 방법이므로, 개발 활동에 영향을 최소화 하는 방식으로 형상 관리가 가능하도록 구현하는 것이 바람직하다.

4. XP의 개발 활동과 CMMI 형상 관리 프로세스 영역 매핑 및 적용

XP는 고객의 요구사항이 변경될 것이라는 가정 하에 프로젝트에 고객을 상주시키며 짧은 기간의 릴리즈를 통하여 고객의 스토리가 적절히 구현되었는지 확인한다. 따라서 고객의 요구사항인 스토리는 매 릴리즈와 이터레이션 계획에 고객과 프로젝트 팀에게 검토되며, 스토리의 수정이나 추가/삭제가 진행된다. 또한 고객과 프로젝트 팀의 합의 하에 릴리즈와 이터레이션에서 완성될 스토리를 선택하게 된다[7]. 이러한 XP의 특징은 형상 관리 프로세스가 릴리즈를 중심으로 구현된다는 것을 의미한다.

4.1 XP의 개발 활동과 CMMI 형상 관리 프로세스 영역의 매핑

다음 (그림 3)은 XP의 활동들과 CMMI 형상 관리 프로세스 영역의 GP와 SP를 매핑한 것이다.



(그림 3) XP 활동과 CMMI 프랙티스의 매핑

(그림 3)의 매핑을 기반으로 CMMI 형상 관리 프로세스 영역 GP와 SP는 다음과 같이 XP에 적용된다.

4.2 CMMI 형상 관리 프로세스 영역 SP의 적용

• SP 1.1 형상 항목 식별

형상 항목 식별의 목표는 프로젝트 진행 중에 변화 관리를 할 필요가 있는 작업 산출물을 선정하여 관리하는 것이다. 이를 통해 중요 작업 산출물에 대한 여러 명의 중복 변경이나 버전 관리 소홀로 인한 덮어쓰기 등의 실수를 줄일 수 있다. 일반적으로 형상 항목의 식별은 베이스라인으로 관리되는 작업 산출물 외에 버전 관리를 하는 작업 산출물과 회의록과 같이 단순히 저장만 하면 되는 작업 산출물로 구분한다. XP에서의 형상 항목 식별은 다음의 (표 2)와 같이 할 수 있다.

(표 2) 형상 항목 식별

베이스라인 대상 항목	버전 관리 대상 항목	저장 항목
스토리	릴리즈 계획	회의록
인수 테스트 케이스	이터레이션 계획	형상 감사 보고서
인수 테스트 결과	프로젝트 계획	-
외부 인터페이스 명세	코딩 표준	-
-	설계서	-
-	소스 코드	-
-	유닛 테스트 케이스 (코드)	-

XP는 고객의 스토리가 프로젝트 진행의 기준이 된다. 스토리는 사용자 스토리(User Story)라고도 표현되는데, 소프트웨어의 사용자나 구매자에게 가치를 줄 수 있는 기능을 서술한 것이다[6]. 프로젝트 시작시 고객으로부터 받은 스토리는 프로젝트 팀과 고객과의 회의 등을 통해 적당한 크기의 스토리로 조정되고, 추정과 우선순위 선정을 통해 여러 번의 릴리즈로 구분하며, 이 때 전체적인 프로젝트 계획이 수립된다[7].

이후 하나의 릴리즈에 대하여 릴리즈 계획과 릴리즈 내의 이터레이션에 대한 이터레이션 계획을 수립한다. 이 과정의 작업 산출물 중 설계서, 소스 코드, 유닛테스트 케이스(코드)는 일반 소프트웨어 개발에서는 형상 항목으로 관리되지만, XP의 특성상 계속적인 통합과 리팩토링으로 인하여 버전 관리 항목이 된다.

그 외에 버전 관리 항목으로는 짝 프로그래밍과 코드의 공동 소유를 위한 코딩 표준 문서 등이 있다

이러한 식별하는 형상 항목은 XP를 적용하는 조직에 따라 추가되거나 변경될 수 있다.

• SP 1.2 형상 관리 시스템 구축

형상 관리 시스템 구축의 목적은 형상 관리 시스템과 변경 관리 시스템에 접근하기 위한 절차와 도구를 구축하는 것이다.

XP에서는 구현 중 지속적인 소프트웨어의 통합과 리팩토링이 진행된다. 그리고 통합을 위해 별도의 컴퓨터를 배치한다[7]. 이를 위해 소스 코드의 변경 관리를 위한 시스템이 필요하다. 또한 릴리즈 시의 베이스라인의 저장과 통제를 위한 형상 관리 시스템과 문서의 버전 관리를 위한형상 관리 시스템이 필요하다.

이러한 버전 관리와 형상 관리 시스템은 CVS[9]나 Subversion[10]과 같은 오픈소스 소프트웨어를 이용할 수 있다. Subversion의 경우 하나의 서버에 두 개 이상의 저장소를 구축하여 각각 소스 코드의 버전 관리와 문서 버전 관리, 베이스라인 관리로 사용 가능하다. 단, 베이스라인 형상 관리 시스템에는 프로젝트 참여자의 접근은 가능하나 베이스라인의 읽기만 가능하며, 베이스라인 추가 및 삭제의 기능은 형상 관리 담당자에게만 권한이 부여된다.

• SP 1.3 베이스라인 생성 및 릴리즈

베이스라인의 생성 및 릴리즈는 추후 개발의 근간이 되는 작업 산출물의 집합인 베이스라인을 공식적인 검토를 통해서 승인하고 항상 활용 가능한 상태로 준비하는 것에 목적이 있다. XP에서 베이스라인은 매 릴리즈 단계의 완료시 생성될 수 있다. 해당 릴리즈가 종료되면 식별한 릴리즈 종료 시의 베이스라인 대상 형상 항목을 베이스라인으로 설정한다. 이 때 릴리즈 합의 이후 고객에게 릴리즈 되므로, 릴리즈 합의가 베이스라인 설정을 위한 승인으로 대체된다.

• SP 2.1 변경 요청 추적

변경 요청 추적의 목적은 변경 사항에 대한 영향을 이해관계자와 함께 검토하고 변경 요청의 상태를 변경 종료 시까지 추적하는데 있다.

XP는 매 이터레이션이나 릴리즈의 시작 단계에 회의를 통해 고객은 스토리의 수정을 제시하며, 프로젝트 팀원은 스토리에 대한 이해와 추정을 통해 스토리를 받아들이거나 또는 다음 이터레이션이나 릴리즈로 완성을 연기할 수 있다. 이 과정이 변경에 대한 검토가 된다. 그리고 추가되거나 수정된 스토리에 대해서는 인수 테스트 케이스의 수정과 작성을 통해 스토리가 릴리즈 될 때까지 추적된다.

• SP 2.2 형상 항목 통제

형상 항목 통제는 베이스라인의 무결성 유지를 위하여 베이스라인의 형상을 적절히 통제하는 것이다.

베이스라인의 변경과 반영을 위해서는 형상통제위원회의 승인이 필요하다. XP에서의 형상통제위원회는 프로젝트에 상주하는 고객과 한 장소에서 일하는 프로젝트 팀원으로 구성될 수 있다. 따라서 XP의 계획단계에서 스토리의 변경, 인수 테스트 통과 후 릴리즈에 대한 고객과 프로젝트 팀의 합의는 베이스라인 변경과 반영을 위한 적절한 형상 통제 절차라 할 수 있다. 그러나 형상 관리 담당자는 스토리의 변경으로 인한 베이스라인의 변경시 변경된 스토리와 사유를 간단히 명시하며, 형상 관리 시스템에 변경 사유를 등록해야 한다.

• SP 3.1 형상 관리 기록 작성

형상 관리 기록은 베이스라인의 생성 및 변경 시 베이스라인을 위한 형상 관리 시스템에 저장된다. 따라서 형상 관리 시스템이 지원하는 기능을 통해 변경 이력과 사유, 형상 항목의 상태와 베이스라인들 간의 차이에 대한 파악이 가능하다.

• SP 3.2 형상 감사 수행

형상 감사의 목적은 베이스라인의 무결성을 확

인하기 위해 형상 관리 활동 및 프로세스를 감사하는 것에 있다.

XP에서는 형상 관리 담당자가 각 릴리즈 종료 후 저장된 베이스라인이 식별된 모든 형상 항목을 포함하는지 여부와 형상 관리 시스템에 기록되어 있는 형상 관리 정보들의 완전성과 정확성을 확인하고 그 결과를 형상 감사 보고서에 기록한다[8].

4.3 CMMI 형상 관리 프로세스 영역 GP의 적용

GP는 프로세스의 내재화를 위해 수행해야 하는 활동으로써 GG 2에 속한 10개 GP는 XP에서 다음과 같이 적용될 수 있다.

GP 2.1: 형상 관리 프로세스를 반드시 수행해야 한다는 조직의 정책을 조직 차원의 문서에 명시한다.

GP 2.2: 형상 관리 계획을 전체 프로젝트 계획에 포함한다. 이 계획에는 형상 관리 프로세스 수행을 위해 필요한 자원, 역할과 책임, 교육, 활동 계획 등이 포함되어야 한다.

GP 2.3: 대표적인 형상 관리 프로세스를 위한 자원인 형상 관리 시스템을 제공한다.

GP 2.4: 형상 관리 프로세스의 수행을 위한 형상 관리 담당자와 형상통제위원회의 책임자를 임명하고 책임을 할당한다. XP에서의 형상통제위원회는 같은 수행 환경에 있는 고객과 프로젝트 팀원 전체가 된다.

형상 관리 담당자는 형상 관리 계획부터 형상 관리 시스템 운영, 베이스라인 설정에 대한 책임이 있으며, 형상통제위원회는 형상 통제에 대한 책임이 있다.

GP 2.5: 형상 관리 담당자 및 개발자에 대한 형상 관리 시스템에 대한 교육을 수행한다.

GP 2.6: 형상 관리 프로세스에서 생성된 작업 산출물도 형상 관리를 실시한다.

GP 2.7: 같은 수행 환경에 있는 고객과 프로젝트 팀원이 이해관계자로 참여된다.

GP 2.8: 형상 관리 프로세스를 형상 항목 변경 건수나 형상 감사 수행 건수 및 결과 등의 지표를 이용하여 모니터링 한다.

GP 2.9: 형상 관리 프로세스의 준수 여부를 프로젝트 외부의 인원이 객관적으로 평가한다. 만약 조직에 품질 보증 팀이 있다면 품질 보증 팀에서 수행하며, 또는 다른 프로젝트의 형상 관리 담당자가 수행할 수 있다.

GP 2.10: GP 2.8과 GP 2.9를 통해 발견된 이슈에 대하여 상위 관리자와 프로세스 상태를 검토한다.

5. CMMI 형상 관리 프로세스 영역 기반의 XP의 형상 관리 프로세스 구축 지침

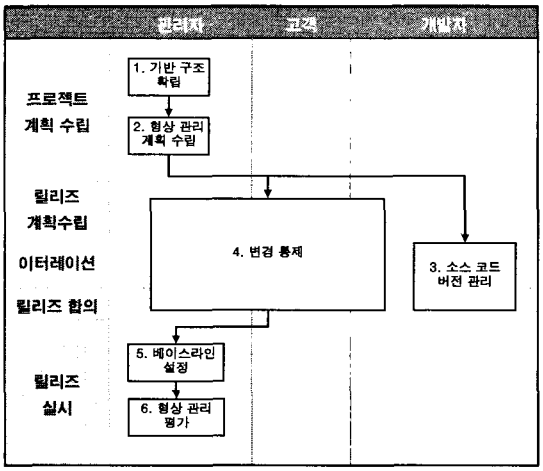
5.1 CMMI 형상 관리 프로세스 영역 기반의 XP에서의 형상 관리 활동 추출

4의 연구를 기반으로 CMMI 형상 관리 프로세스 영역 기반의 XP의 형상 관리 활동을 추출하면 다음의 (표 3)과 같다.

(표 3) XP에서의 형상 관리 활동

형상 관리 활동	관련 CMMI CM PA 활동		내용
	SP	GP	
1. 기반 구조 확립	1.2	2.1, 2.3, 2.4, 2.5	형상 관리를 위한 기반 구조의 확립
2. 형상 관리 계획 수립	1.1	2.2, 2.6, 2.7	형상 관리 계획서 작성 및 형상 식별
3. 소스 코드 버전 관리	2.2, 3.1	-	지속적인 통합과 리팩토링을 위한 소스 코드 버전 관리
4. 변경 통제	2.1, 3.1	-	고객의 스토리 변경에 대한 통제
5. 베이스라인 설정	1.3, 3.1, 3.2	-	베이스라인 설정 통제 형상 감사 형상 관리 시스템 백업
6. 형상 관리 평가	1.2	2.8, 2.9. 2.10	형상 관리 활동 모니터링 및 품질 보증

이러한 XP의 형상 관리 활동을 프로세스화 하여 Swimlane Diagram으로 표현하면 다음 (그림 4)와 같다.



(그림 4) XP 형상 관리 프로세스 Swimlane Diagram

즉, XP에서의 형상 관리 프로세스는 관리자, 고객, 개발자에 의해 수행되며, 고객은 개발자와 함께 변경에 대한 합의를 하는 통제에만 참여하게 되고 개발자는 변경 통제 외에 소스 코드 버전 관리에 참여하게 된다. 그 외의 활동은 형상 관리를 지원하는 도구와 관리자에 의해 수행된다.

5.2 CMMI 형상 관리 프로세스 영역 기반의 XP의 형상 관리 프로세스 구축 지침

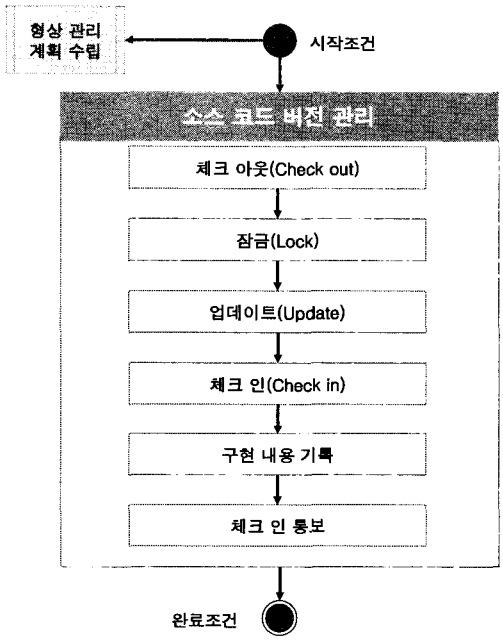
5.1에서 추출된 XP에서의 형상 관리 활동 중, 일반 소프트웨어 개발과 구분되며 XP의 특징이 반영된 소스 코드 버전 관리와 변경 통제, 베이스라인 설정에 대한 구축 지침은 다음과 같다. 본 지침에서는 형상 관리 시스템으로 Subversion을 사용하는 것을 가정한다.

5.2.1 소스 코드 버전 관리

소스 코드 버전 관리는 구현에 따른 지속적인 통합, 코드 공동 소유에 바탕을 둔 리팩토링의 수행을 위해 반드시 필요한 형상 관리 활동이다.

소스 코드 버전 관리를 위한 형상 관리 시스템은 개발자 각각에 대한 ID는 배부하나 패스워드는 설정하지 않아야 한다. 그리고 체크 인/아웃, 잠금, 업데이트의 기능이 요구된다. 또한 소스 코드는 유닛 테스트를 위한 코드와 함께 관리된다.

소스 코드 버전 관리 프로세스를 도식화하면 다음과 같다.



(그림 5) 소스 코드 버전 관리 프로세스

• 체크 아웃(Check out)

개발자는 소스 코드 버전 관리를 위한 형상 관리 시스템의 저장소에서 짝프로그래밍을 위한 컴퓨터로 소스 코드를 체크 아웃 한다. 이 소스코드는 마지막으로 배포된 소스 코드여야 하며, 반드시 모든 소스 코드를 체크 아웃 해야 한다.

• 잠금(Lock)

개발자는 체크 아웃한 소스 코드에서 자신이

수정해야 하는 코드가 있다면 잠금 기능을 이용하여 해당 소스 코드를 다른 개발자가 수정할 수 없도록 한다.

• 업데이트(Update)

개발자는 자신이 작성한 코드의 테스트가 끝나면 통합을 위하여 체크 아웃한 모든 소스 코드의 유닛 테스트를 실시를 위해 소스 코드 업데이트를 수행한다. 이 절차를 통해 개발자가 체크 아웃한 소스 코드는 최신 배포 버전으로 업데이트 된다. 이 후 통합을 위한 유닛 테스트를 수행한다.

• 체크 인(Check in, 또는 Commit)

모든 체크 아웃한 소스코드의 유닛 테스트를 실시하여 모두 통과하면, 개발자는 배포를 위해 체크 인을 수행한다. 체크 인의 형상 관리 시스템 인증 ID는 드라이버의 ID로 수행한다. 이 절차를 통해 개발자가 구현한 스토리는 배포된다.

• 구현 내용 기록

개발자는 체크 인 수행시 형상 관리 시스템의 변경 기록 기능을 이용하여 해당 배포에 구현한 스토리와 기능에 대해 반드시 기록한다. 기록하는 항목은 다음의 내용이 포함될 수 있다.

(표 4) 구현 내용 기록 항목의 예

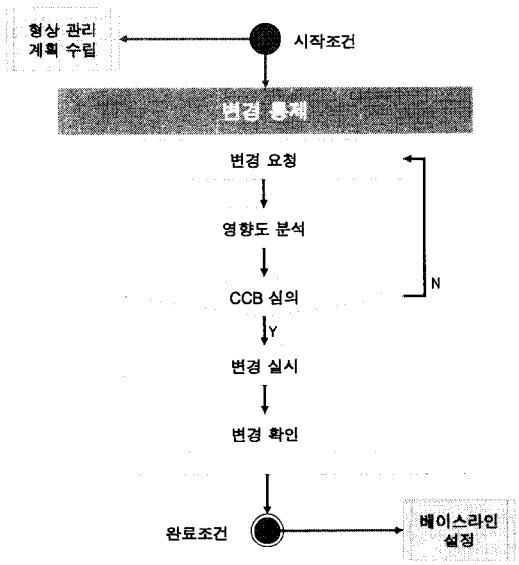
항 목	설 명
스토리	스토리를 구현한 경우 구현의 근원이 되는 스토리
구현 부분 설명	구현한 소스 코드에 대한 간단한 설명
파트너	짝프로그래밍시 함께 작업한 파트너

• 배포 통보

구현 내용 기록을 포함한 체크 인을 완료한 개발자는 배포에 대하여 구두 또는 메일로 프로젝트 구성원에게 통보한다.

5.2.2 변경 통제

변경 통제는 고객으로부터 구현 또는 완성된 스토리에 대하여 변경 요청이 있을 경우 고객과 개발자의 합의를 통해 변경을 수행하는 활동이다. 변경 통제 프로세스를 도식화하면 다음과 같다.



(그림 6) 변경 통제 프로세스

변경 통제 절차는 XP에서의 각 이터레이션 시작시 고객과 개발자가 스토리를 추정/합의하고 구현하는 절차와 동일하다. 따라서 변경 통제는 최소 하나의 이터레이션, 최대 하나의 릴리즈 내에서 종료된다. 그러나 변경 현황을 기록하는 절차가 추가 된다.

• 변경 요청

XP의 특성에 따라 고객은 구현 중이거나 이미 구현된 스토리에 대하여 변경 요청을 할 수 있다. 각 릴리즈 또는 이터레이션 시작시 고객은 수정된 스토리를 제시하며 변경 요청을 한다. 형상 관리 담당자는 변경 요청에 대하여 변경 요청 목록의 해당 부분에 기록한다. 변경 요청 목록에는 변

경 요청자, 변경 요청 단계, 추정치, 변경 여부 등이 포함될 수 있다.

• 영향도 분석

개발자는 고객이 제시한 수정된 스토리에 대하여 추정을 실시한다. 형상 관리 담당자는 추정치를 변경 요청 목록에 기록한다.

• 형상통제위원회(CCB) 심의

형상통제위원회인 고객과 개발자는 개발자의 추정치를 바탕으로 수정된 스토리의 구현 여부를 결정한다. 스토리의 추정치가 클 경우 개발자는 고객에게 스토리를 작은 몇 개의 스토리로 나누어 줄 것을 요청할 수 있다. 변경이 결정되면 각 이터레이션 시작과 같이 해당 스토리의 구현을 원하는 개발자가 구현을 담당하게 된다. 만약 변경 요청이 기각되었을 경우 스토리는 고객에게 반려된다.

형상 관리 담당자는 변경 여부와 변경 결정시 변경 담당 개발자를 변경 요청 목록에 기록한다.

• 변경 실시

수정된 스토리의 변경이 결정되면 개발자는 '소스 코드 버전 관리' 프로세스에 의해 변경을 실시한다. '체크 아웃-잠금-업데이트-체크 인' 절차에 의해 변경이 실시된다.

• 변경 확인

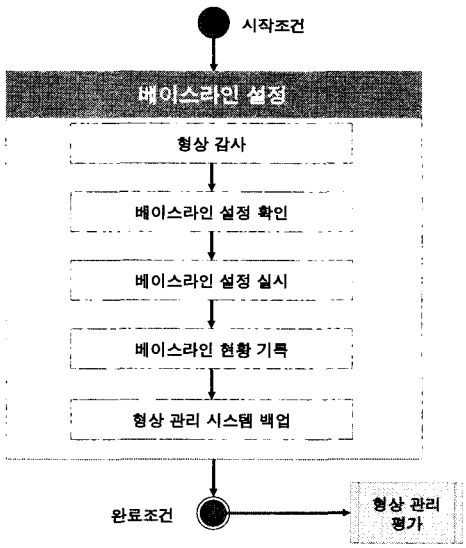
변경이 수행된 해당 이터레이션의 고객 인수 테스트를 통해 변경 완료 여부를 확인한다. 이 때 인수 테스트 케이스는 베이스라인 형상 관리 시스템으로부터 체크 아웃 하여 수정된 스토리에 맞게 수정한다. 인수 테스트가 완료되고 고객이 동의하였을 경우 변경이 완료된다. 이 후 베이스라인 설정 프로세스로 이관된다.

형상 관리 담당자는 변경 확인일은 변경 요청 목록에 기록한다.

5.2.3 베이스라인 설정

베이스라인 설정은 공식적인 절차를 통해서만 변경이 가능한 베이스라인의 설정을 고객과 확인하고 실제 실시를 하며, 형상 관리 시스템을 백업하는 활동을 포함한다.

베이스라인 설정 프로세스를 도식화 하면 다음과 같다.



(그림 7) 베이스라인 설정 프로세스

• 형상 감사

형상 감사는 베이스라인의 무결성을 보장하기 위한 활동이다. 형상 관리담당자에 의해 수행되며, 체크리스트를 이용하여 수행한다. 형상 감사에서 확인하는 항목은 다음과 같은 것이 있다 [11].

✓ 승인된 변경의 베이스라인 반영 여부
✓ 관련된 산출물의 갱신 여부
✓ 승인되지 않은 베이스라인 항목의 변경 여부

형상 감사에 의해 발견된 부적합 사항에 대해

서는 형상 관리 담당자가 시정을 위해 적절한 조치를 한다.

• 베이스라인 설정 확인

형상 감사에서 부적합 사항이 발견되지 않거나 모든 부적합 사항이 수정된 후, 형상 관리 담당자와 형상통제위원회인 고객과 개발자는 해당 릴리즈의 베이스라인 설정을 합의한다. 합의는 구두 보고 및 구두 동의를 통해 이루어진다.

• 베이스라인 설정 실시

형상 관리 담당자는 베이스라인 대상 형상 항목이 저장되어 있는 형상 관리 시스템에서 마지막 버전을 익스포트(Export)하여 베이스라인 형상 관리 시스템에 해당 릴리즈의 디렉토리를 만들고 임포트(Import) 한다.

만약 변경 요청에 의한 베이스라인 설정일 경우, 변경 확인 후 형상 감사 단계를 생략하고 베이스라인 설정 합의 후 베이스라인 대상 형상 항목을 베이스라인 형상 관리 시스템에 체크 인 한다.

• 베이스라인 현황 기록

베이스라인 현황 기록은 형상 관리 담당자에 의해 수행된다.

익스포트된 베이스라인의 경우 익스포트된 항목과 베이스라인 설정 이전의 최종 버전에 대하여 형상 관리 시스템에 기록한다.

변경 요청에 의해 수정된 베이스라인의 경우 수정 요청된 스토리와 수정 요청 일시, 수정 요청에 대한 간략한 설명을 기록한다.

• 형상 관리 시스템 백업

형상 관리 담당자는 형상 관리 시스템을 형상 관리 계획서에 명시된 계획에 따라 백업을 실시한다. 백업 시 반드시 다음의 사항을 시행하여야 한다.

- ✓ 백업은 같은 컴퓨터의 다른 물리적 하드디스크, 다른 컴퓨터, 또는 CD나 DVD에 실시한다.
- ✓ 형상 관리 시스템에서 익스포트를 이용하여 백업하지 않고, 로그도 같이 백업될 수 있도록 저장소 자체를 백업한다.

6. 결론 및 향후 연구 방향

XP에서 형상 관리가 적절히 수행되지 않는다면 XP의 실천사항인 지속적인 통합, 리팩토링을 수행하며 산출물의 무결성을 보장하는 것은 어렵다.

본 연구에서는 XP에서 개발 산출물의 무결성 확보를 위해 XP의 실천사항과 개발 특징, 개발 산출물 관리의 특성에 대하여 연구하고 CMMI의 형상 관리 프로세스를 적용하여, CMMI 기반의 XP를 위한 형상 관리 프로세스 구축 지침을 개발하였다

이를 통해 XP 방법을 이용하여 프로젝트를 수행하는 조직은 다음과 같은 장점을 얻을 수 있다.

1) 형상 관리 프로세스 구축 지침을 이용해 조직에 유연하게 조정하여 형상 관리 프로세스의 구현 가능
2) XP의 기민함을 유지하고 개발자에게 추가 작업을 요구하지 않으면서 지속적인 통합과 리팩토링에서의 소스 코드 충돌을 해결할 수 있는 소스 코드 버전 관리 절차 정의
3) 릴리즈별 베이스라인 설정을 통한 산출물의 무결성 확보

즉, XP의 기민한 특징을 잃지 않으면서도 효과적으로 형상 관리 프로세스를 구현하는 것이 가능하며, 이를 통해 소스 코드를 비롯한 개발 산출물의 무결성을 확보할 수 있다.

차후 연구에서는 본 XP에서의 형상 관리 프로세스 구축 지침을 확대하고 구조화한 프레임워크

로의 확장이 필요하며, XP에서의 형상 관리 프로세스가 적용 분야 또는 프로젝트 및 조직의 특성별로 어떠한 조정 지침이 요구되는지에 대한 방안의 연구가 필요하다.

참 고 문 헌

[1] Alistair Cockburn, "Agile Software Development", Addison-Wesley, 2002

[2] Kent Beck, "Extreme Programming Explained 2/E", Addison-Wesley, Nov., 2004

[3] Martin Fritzsche, Patrick Keil, "Agile Methods and CMMI: Compatibility or Conflict?", e-Informatica Software Engineering Journal, Volume 1, Issue 1, 2007

[4] Paulk, M.C., " Extreme programming from a CMM perspective", Software, IEEE, Volume: 18, Issue: 6, pp.19-26, 2001

[5] Mary Beth Chrissis, Mike Konrad, Sand Shrum, "CMMI Second Edition : Guidelines for Process Integration and Product Improvement", Addison-Wesley, 2006

[6] Mike Cohn 저, 한주영, 심우곤, 송인철 역, "사용자 스토리", 인사이트, 2006

[7] Ron Jeffries, Ann Anderson, Chet Handrickson, "Extreme Programming Installed", Addsion-Wesley, 2002

[8] Margaret K. Kulpa, Kent A. Johnson, "Interpreting the CMMI", Auerbach Publication, 2003

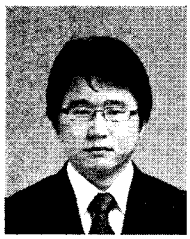
[9] http://www.nongnu.org/cvs

[10] http://subversion.tigris.org/

[11] 한혁수, "소프트웨어 공학의 소개", 홍릉과학출판사, 2007

[12] ANSI/IEEE Std 1042-1987 "IEEE Guide to Software Configuration Management"

◐ 저 자 소 개 ◑



한 동 준(Dong Joon Han)

2006 상명대학교 소프트웨어학과 졸업(학사)
2006 ~ 현재 상명대학교 일반대학원 컴퓨터과학과 (석사과정)
관심분야: 소프트웨어 프로세스, 소프트웨어 품질, CMMI
E-mail: lexia@smu.ac.kr

한 혁 수(Hyuk-Soo Han)

1985 서울대학교 계산통계학과 (학사)
1987서울대학교 계산통계학과 (석사)
1992 Univ. of South Florida 전산학과 (공학박사)
1993 ~ 현재 상명대학교 소프트웨어 대학 소프트웨어 학부 교수
2000 ~ 2003.2 시스템통합기술연구원장
2003. 1 ~ 2004. 2 한국소프트웨어진흥원 소프트웨어공학센터 소장
관심분야: 소프트웨어 프로세스, 소프트웨어 품질, 소프트웨어 아키텍쳐, 소프트웨어 사용성 평가 등
E-mail: hshan@smu.ac.kr