CMM과 프로세스 개선 사례

(주)LG-EDS 시스템 정학종·김도균·박남직

1. 서 론

1990년대가 시작되자, Newsweek지는 특집 기사에서 "Can we Trust Our Software?"라고 의문을 제기했고, Wall Street Journal은 Creating New Software was an Agonizing Task…라는 전면 타이틀로 주요 소프트웨어 개발 회사가 겪고 있는 고통을 언급하였다[1]. 실제 1994년 Standard Group의 조사에 따르면 조사 대상 8,380개 프로젝트에서 품질, 납기, 비용을 충족한 프로젝트 비율은 16.2%에 불과하며, 재작업 없이 완료한 프로젝트는 6%에 불과하다.

그리고 비록 개발은 완료되었으나 시스템이 전달되었을 때 소프트웨어의 결함으로 고객은 많은 실패를 겪게 된다. 그림 1에서 보듯이, 신규 소프트웨어 개발보다 기존 소프트웨어의 유지보수에 더 많은 투자가 이루어지고 있다는 것에서 소프트웨어 품질에 대한 문제점을 인식할 수 있다[2].

그리고 소프트웨어 개발 생산성 향상이 대부분의 소프트웨어 개발 조직의 주요 우선 과제

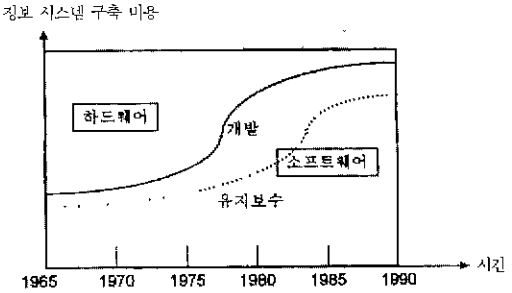


그림 1 정보시스템 구축시 비용구성률 추이

였음에도 불구하고, 컴퓨터 하드웨어의 성능이 3년마다 거의 2배로 향상되는 반면 소프트웨어의 생산성은 매년 4% 정도 향상에 그치고 있다. 이러한 문제가 개선되지 않는다면 결국 낮은 생산성과 낮은 제품 품질로 소프트웨어 개발 조직은 이익감소와 함께 사업기회를 상실하게 될 것이다[3].

한편 전통적으로 각 조직들은 생산성과 품질을 높이기 위하여 새로운 Tool이나 기술 도입에 의존하여 왔다. 1970년대의 구조적 개발 기법에서부터 시작하여 80년대, 90년대에 걸쳐 C와 Ada등의 구조적 언어의 도입, CASE 활용, 객체지향, 프로토타이핑 개발 방법 등을 적용하여 어느 정도의 소프트웨어의 생산성과 품질을 높일 수 있었지만 근본적인 해결책은 되지 못했다.

이러한 원인은 Tool이나 기술에 문제가 있었던 것이 아니라 접근 방법에 문제가 있다. 전통적으로 실패를 경험하면 새로운 도구나 기술을 도입하고 전문가를 고용하고는 모든 것이 해결된다는 환상에 빠졌다. 그리고 다시 실패를 경험하면 또 새로운 도구나 기술을 도입하게 된다.

따라서 이러한 여러 가지 문제점들을 근본적으로 해결하기 위해서는 지금부터 새로운 접근 방법이 모색되어야 한다. 1987년 미국방성 백서(DoD87)는, "지난 20여년 동안 새로운 소프트웨어 방법론과 기술 적용에도 불구하고 생산성과 품질 달성에 실패한 근본적인 문제는 소프트웨어 프로세스의 관리 실패에 있다."고 했으며, Watts Humphrey는 "소프트웨어 시스템의 품질은 그것을 개발하고 유지보수하기 위해

사용된 프로세스의 품질에 전적으로 달려 있다."고 하면서 프로세스의 중요성을 강조하였다.

새로운 접근 방법이란 바로 프로세스적인 접근 방법을 말한다. 프로세스적인 접근 방법이란 현재의 소프트웨어 문제점을 평가하고 결과를 향상시키기 위해 프로세스를 조정하며, 도구 선정을 위한 프로세스 모델을 사용하는 것부터 시작한다. 그리고 프로세스에 적절한 도구를 도입하고 프로세스 효과를 평가하여 지속적으로 프로세스를 개선하는 것이다[4].

실제 본 논문에서 소개하려는 CMM[1](Capability Maturity Model) 기반의 소프트웨어 프로세스의 개선에 따른 효과는 다음과 같은 것으로 보고되었다.

표 1 프로세스적인 접근방법의 효과[5]

구 분	범위	평균
프로세스 개선 기간	1~9년	3.5년
연간 개선된 생산성	9~67%	35%
연간 초기결함 발견 향상율	6~25%	22%
연간개발시간 감소율	15~23%	19%
연간 고객 인도 후 결함 발생률	10~94%	39%
사업상 가치(절감액/소프트웨어 프로세스 개선 비용)	4.0~8.8:1	5.0:1

프로세스를 개선하기 위해서는 계획적인 활동이 필요하다. Humphrey는 프로세스 개선을 위해 다음과 같은 단계가 필요하다고 하였다[6].

1. 현재의 프로세스 상태 이해
2. 추구하는 프로세스 모습 개발
3. 우선 순위를 위한 프로세스 개선 활동 내역 개발
4. 활동 달성을 위한 계획 수립
5. 계획을 실행할 수 있도록 자원 할당
6. 1단계부터 다시 시작

이를 바탕으로 CMM을 개발한 SEI(Software Engineering Institute)에서는 소프트웨어 프로세스 개선 활동을 위하여 조직 차원의 개선 모델인 IDEAL 모델을 제공하고 있다. IDEAL 모델은 그림 2와 같은 5단계로 이루어져 있으

1) CMM(Capability Maturity Model)은 카네기 멜론 대학의 Service Mark임.

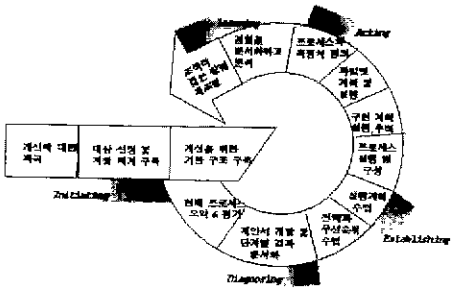


그림 2 IDEAL 모델[7]

며 각 단계의 첫자를 따서 IDEAL이라고 한다. 각 단계의 내용은 다음과 같다.

- Initiating : 성공적인 개선을 위한 기초 작업을 수행
- Diagnosing : 조직의 현재의 위치 파악
- Establishing : 조직의 목표를 달성하기 위한 계획 수립
- Acting : 계획에 따라 개선 활동 수행
- Learning : 개선 활동에 대한 경험을 습득하고, 신기술에 대한 적용 능력을 개선

CMM은 IDEAL 모델에서 Diagnosing을 위한 소프트웨어 프로세스 능력 심사 모델로서, 조직 내부 프로세스 개선을 위한 CBA-IPI (CMM Based Appraisal Internal Process Improvement)와 개발업체 선정을 위한 업체 능력 평가를 위한 CBA-SCE(CMM Based Appraisal Software Capability Evaluation) 방법이 있다. 본 논문에서는 CMM 소개와 함께 내부 프로세스 개선을 위해 CMM을 적용한 국내 한 SI업체의 사례를 살펴봄으로써 향후 국내 소프트웨어 프로세스 개선 방향에 대해 논의하고자 한다.

2. CMM 소개

2.1 CMM(Capability Maturity Model) 개요

CMM은 미국 카네기 멜론 대학의 소프트웨어 공학연구소(SEI : Software Engineering Institute)에서 국방성의 자금을 지원받아 개발한 모델이다. 처음에는 국방성의 요청에 따라 국방성이 발주하는 소프트웨어 프로젝트의 위

험을 줄이기 위해 입찰자들의 개발 능력을 평가하기 위해 개발되었으나, 이후 이 모델이 갖고 있는 성숙도에 대한 프레임웍이 국방 관련 프로젝트 뿐만이 아닌 상업적인 소프트웨어 개발 업체에서도 인정을 받으면서 널리 활용되기 시작하였다.

CMM을 이해하기 위해서는 먼저 핵심 개념인 프로세스(Process)와 능력(Capability), 성숙도(Maturity)에 대해 살펴볼 필요가 있다. SEI는 프로세스를 "원하는 결과를 만들기 위해 사람, 절차, 방법, 도구를 통합시키는 수단"이라고 정의하였으며, Fagan은 프로세스를 "명확한 순서로 입력물을 원하는 산출물로 전환하는 일련의 운용"이라고 정의하였다[8].

따라서 프로세스는 그림 3에서 보듯이, 입력물을 산출물로 변환하는데 사용하는 일련의 활

그림 3 프로세스 개념

동, 방법, 실행과 변형이라고 정의할 수 있다.

프로세스가 효과적이기 위해서는 프로세스의 실제 결과를 예측할 수 있어야 한다. 예를 들면, 프로세스를 따랐을 때 견적한 비용, 일정과 일치할 수 있어야 하고 또한 최종 제품의 품질은 고객의 요구를 충족시킬 수 있어야 한다. 예측과 실제 나타난 결과와의 부합 정도를 프로세스 능력(Process Capability)이라고 한다.

그러나 결과 예측은 프로세스 자체의 결함과 비용, 스케줄 등의 제약과 같은 여러 가지 변수에 의해 변동 가능성이 높아 예측하기 어렵고, 예측을 하여도 실제 결과와 상당한 격차가 있을 수 있다. 기대되는 결과를 예측하는 능력을 높이기 위해서는, 즉 프로세스 능력을 높이기 위해서는 프로세스가 명시적으로 정의되고, 관리되고, 측정되고, 통제될 수 있어야 한다. SEI는 특정 프로세스가 정의되고, 관리되고, 측정되고, 통제되는 정도를 프로세스 성숙도(Process Maturity)라고 한다.

Humphrey는 프로세스의 중요성에 대한 강조와 함께 87년 SEI Technical Report에 프로세스 능력을 일관적이고 체계적으로 성숙시킬 수 있는 프로세스 성숙도 프레임웍(Process Maturity Framework)을 발표하였다. 이 프레임웍은 Philip Crosby의 품질 성숙도 구조인 Five Evolutionary Stages in Adapting Quality Practices[9]를 소프트웨어 프로세스에 적용한 것으로 그림 4와 같다.

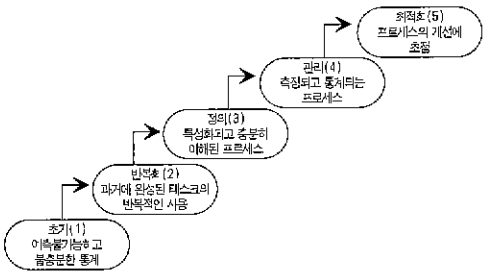


그림 4 Process Maturity Framework

CMM은 SEI가 바로 이 Process Matarity Framework에 심사방법을 제공하여 개발된 모델이다.

2.2 CMM 구조 및 단계 이해

CMM은 5개의 성숙도 단계를 가지고 있는데 각 단계와 단계별 특성은 그림 5와 같다.

CMM의 각 단계는 다음 단계로 진화되기 위해 반드시 달성해야 하는 핵심 프로세스가 있는데 CMM에서는 이를 KPA(Key Process Area)라고 한다. CMM의 각 단계를 이해하기 위해 간략하게 상위 단계로 성숙되어 가는 과정을 살펴보기로 한다.

성숙도 1단계(Initial;초기 단계)는 프로세스가 아닌 작업자의 능력에 의해 프로젝트의 성과가 좌우된다. 따라서 뛰어난 사람을 고용하면 높은 품질과 예외적인 성과가 가능하지만

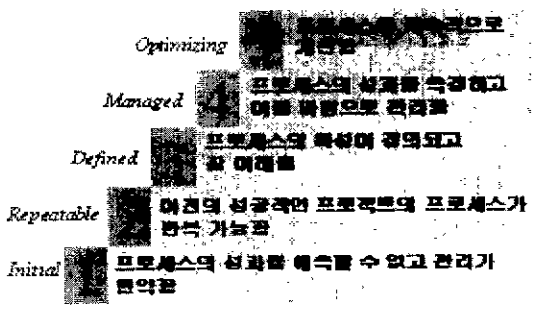


그림 5 CMM의 능력 성숙도 단계

일반적으로 일의 결과를 예측할 수 없다. 이 조직이 직면하고 있는 문제점은 기술적인 측면이 아니라 관리적인 측면에 있다. 소프트웨어 관리가 되지 않으며 소프트웨어 제품은 그림 6과 같이 무정형의(파악할 수 없는) 프로세스에서 만들어진다. 따라서 1단계에서는 핵심 프로세스가 없다.

그림 6 성숙도 1단계에서의 프로세스 관점[10]

1단계 조직에서도 성공한 프로젝트가 있을 수 있다. 성숙도 2단계(Repeatable; 반복된 단계)는 성공한 프로젝트의 사례를 반복하는 단계로서 초점은 프로젝트이다. 따라서 2단계 조직은 효과적으로 소프트웨어 프로젝트 관리를 하기 위한 강력한 니즈가 확립되며 프로젝트 관리와 관련된 프로세스들이 문서화되고 준수된다. 소프트웨어 제품은 그림 7과 같이 일련의 Black Box에서 만들어지지만, 프로젝트 관리 시스템이 만들어져서 미리 정의된 체크 포인트(마일스톤)에서 관리된다.

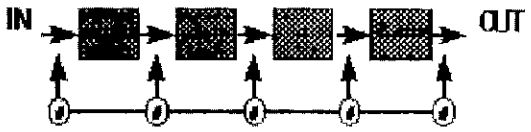


그림 7 성숙도 2단계에서의 프로세스 관점[10]

2단계는 프로젝트 관리와 관련된 6개의 핵심 프로세스를 가지고 있다.

- 요구사항 관리(Requirement Management)
- 소프트웨어 프로젝트 계획 수립(Software Project Planning)
- 소프트웨어 프로젝트 추적(Software Project Tracking and Oversight)
- 소프트웨어 외주관리(Software Subcontract Management)
- 소프트웨어 구성관리(Software Configuration Management)
- 소프트웨어 품질보증(Software Quality Assurance)

성숙도 3단계(Defined; 정의된 단계)는 조직 차원에서 프로세스를 정의하여 관리하는 단계이다. 따라서 소프트웨어 프로젝트 관리 기초 위에서 구축되며 프로세스를 통제하기 위해 반드시 프로세스가 정의, 문서화되며, 한 업무의 출력물이 다음 업무의 입력물로 자연스럽게 진행된다.

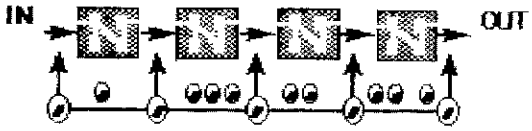


그림 8 성숙도 3단계에서의 프로세스 관점[10]

이 단계에서는 그림 8과 같이 잘 정의된 프로세스에 따라 관리되며 소프트웨어 제품의 생산은 전체 소프트웨어 프로세스에 걸쳐 가시화된다.

3단계는 조직과 관련된 7개의 핵심 프로세스를 가지고 있다.

- 조직 프로세스 지향(Organizational Process Focus)
- 조직 프로세스 정의(Organizational Process Definition)
- 교육 프로그램(Training Program)
- 통합 소프트웨어 관리(Integrated Software Management)
- 소프트웨어 제품 엔지니어링(Software Product Engineering)
- 그룹간 조정(Intergroup Coordination)
- 동료 검토(Peer Review)

프로세스가 정의되면 측정을 통한 프로세스 관리가 가능해진다. 성숙도 4단계(Managed; 관리된 단계)는 프로세스 변동의 이상 원인을 파악하기 위해 통계적 프로세스 관리 원칙을 적용한 단계로서 그림 9에서 보는 것처럼 제품과 프로세스가 정량적으로 관리된다. 경영층은 객관적 근거를 가지고 의사결정을 내릴 수 있

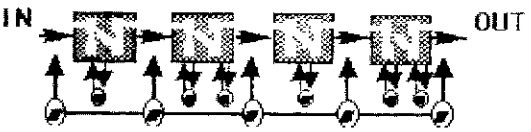


그림 9 성숙도 4단계에서의 프로세스 관점[10]

으며 정량화된 한도내에서 성과를 예측할 수 있다.

4단계는 정량적 관리 대상인 2개의 핵심 프로세스를 갖고 있다.

- 정량적인 프로세스 관리(Quantitative Process Management)
- 소프트웨어 품질 관리(Software Quality Management)

성숙도 5단계(Optimizing; 최적화 단계)는 정량적 관리를 통해 결함 발견/제거가 아닌 낮은 성과를 유발하는 만성적인 원인을 파악하여 사전 제거하는 단계로 그림 10에서 보는 것처럼 프로세스 자체가 변화된다.

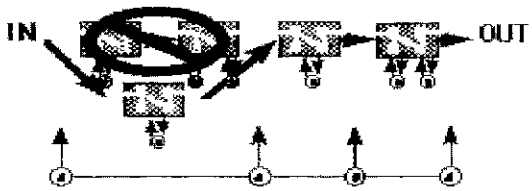


그림 10. 성숙도 5단계에서의 프로세스 관점[10]

5단계는 변화 관리와 관련된 3개의 핵심 프로세스를 가지고 있다.

- 결함 예방(Defect Prevention)
- 기술 변경 관리(Technical Change Management)
- 프로세스 변경 관리(Process Change Management)

2.3 CMM 심사 방법

앞에서 CMM 심사 방법으로 조직 내부 프로세스 개선을 위한 CBA-IPI와 개발 업체 선정을 위한 CBA-SCE 방법이 있다고 언급하였다. 본 논문은 CMM 적용을 통한 조직 개선 사례를 다루므로 CBA-IPI 방법에 대해 살펴보기로 한다.

IPI의 목적은 소프트웨어 프로세스 개선을 위해 조직의 현재 프로세스의 강, 약점에 대한 정확한 상황을 제공하고 개선을 위한 프로세스를 규명하는 것이다. 심사 대상은 조직을 대표하는 4개 정도의 프로젝트를 선정하여 실시하며, 사전 심사와 본 심사를 통해 능력을 결정한다.

심사는 SEI에서 공인을 받은 Lead Assessor의 주도하에 SEI에서 요구하는 경력을 충족하는 조직 대표 4~10명에 의해 수행된다. 피심사자는 대상 조직의 팀장과 프로젝트 리더와 함께 영역별 대표자가 된다[11].

심사 영역은 소프트웨어 개발 및 유지보수 프로세스와 주요 지원 프로세스이며 일반적으로 다음과 같은 프로세스를 포함한다.

- 요구사항 관리
- 설계
- 코딩&단위 테스트
- 통합/시스템 테스트
- 사용자 승인 테스트
- 품질 보증
- 구성 관리
- 사용자 문서 관리

심사는 다음과 같이 실시된다.

1) 사전 심사

- 설문심사: 현상 파악을 위한 성격으로 피심자자들이 자신의 프로세스에 대해 스스로 평가를 한다. 설문심사 결과는 등급 결정에 영향을 주지 않지만, 향후 본 심사를 위한 대상자 선정과 질문 내용 조정에 도움을 준다.
- 초기 문서 심사: 본 심사의 인터뷰시 사용할 질문서를 작성하고 프로세스가 정착되어 있는지를 확인하기 위한 데이타를 수집하고 CMM의 내용과 심사 조직의 문서들을 Mapping한다.

2) 본 심사

본 심사는 그림 11과 같이 실시된다. 기능별 대표자 인터뷰는 심사영역별로 그룹 인터뷰를 통해 심사한다.

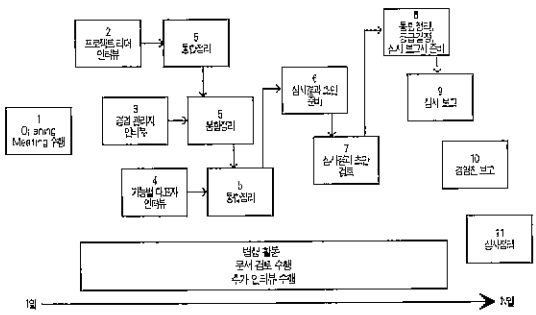


그림 11 본 심사 Activities

예를 들어 A라는 조직을 심사하기 위해 4개의 프로젝트를 선정하였을 경우, 4개 프로젝트의 요구사항 관리 담당자를 그룹으로 인터뷰하며 계속해서 다른 영역별로 그룹 인터뷰를 실시한다.

3. CMM 적용 사례

3.1 현 상태 파악

프로세스 개선은 현 상태를 파악하는 것으로부터 시작한다. 본 사례 조직은 프로세스 관리 능력 개선을 통해 부서의 실행 능력을 향상시키고 이를 통해 고객에게 실질적인 만족을 제공한다는 목표를 수립하고 이를 위해 프로세스를 개선하기로 하였다. 조직의 현재 프로세스 수준을 파악하기 위해 96년 CMM 전문가에 의해 CMM에 따른 내부 심사를 실시하였다. 심사 대상은 사업부별로 대표성이 있는 6개 프로젝트를 선정하였으며 심사 프로세스는 CMM 2단계를 대상으로 하되 해당이 없는 외주관리는 제외하였다. 각 프로세스별 요구하는 목적과 심사 결과는 다음과 같다.

1) 요구사항 관리

• 목적 : 프로젝트팀과 고객의 요구사항과의 사이에서 공통의 이해 기반을 형성하는 것으로, 여기에는 고객 요구사항의 문서화 및 통제와 함께 계획, 산출물, 활동을 요구사항과 일관성 있게 유지하는 것이 포함된다.

• 심사 결과

- 요구사항 접수, 배분, 처리에 일관성이 없음.
- 요구사항 변경에 대한 변경관리가 제대로 되지 않음.
- 요구사항 처리 현황 등이 주기적으로 보고되지 않아 관리가 안됨.
- 요구사항 추적이 어렵고 검증 활동이 미흡함.

2) 프로젝트 계획 수립

• 목적 : 소프트웨어 엔지니어링을 실행하고 소프트웨어 프로젝트를 관리하기 위한 합리적인 계획을 수립하는 것으로, 여기에는 실행되는 작업에 대한 Estimates를 개발하고 작업을 수행하기 위한 계획을 수립하는 것을 포함한다.

• 심사 결과

- 프로젝트 크기, 일정, 비용 산정시 주로 과거 경험에만 의존하고 공식적인 절차가 없음.
- 일정 개발시 교육 및 신기술 습득을 위한 시간이나 품질보증, 검증과 같은 항목을 포함하지 않아 전체 일정에 역으로 영향을 미침.

3) 프로젝트 추적

• 목적 : 프로젝트 성과가 계획과 크게 벗어난 경우 관리층이 효과적인 조치를 취할 수 있도록 실질적인 진척에 대한 충분한 가시성을 제공하는 것으로, 여기에는 문서화된 Estimates, 계획에 대한 성과와 결과를 추적, 검토하고 이에 따라 계획을 재조정하는 것이 포함된다.

• 심사 결과

- 대부분의 스케줄은 마스터 플랜만 있고 세부적인 스케줄이 없어 추적 관점에서 기술적, 관리적 문제가 있음.
- 고객의 주요 요구사항이 변경되거나 또는 계획이 실제대로 진척되지 않아도 스케줄이 변경되지 않음.
- 프로젝트 계획에 대한 진척을 추적할 만한 적절한 측정치가 존재하지 않음.

4) 구성관리

• 목적 : 소프트웨어 Life Cycle 전 과정을 통해 프로젝트 제품의 무결성을 설정하고 유지하기 위한 것으로, 여기에는 구성관리 항목/단위를 정의하고 변경을 체계적으로 관리하며 구성항목의 무결성과 추적성을 유지하는 것이 포함된다.

• 심사 결과

- 프로젝트 계획서 내에 별도의 구성관리 계획서가 부재
- 프로젝트 수행전 베이스라인에 대한 이해 부족
- 구성 관리를 위한 프로세스가 아닌 최종 산출물에 치우침.

5) 품질보증

• 목적 : 사용되고 있는 프로세스와 구축되고 있는 제품에 대해 적절한 가시성을 제공하기 위한 것으로, 여기에는 소프트웨어 제품과 활동이 절차 및 표준과 일치하는 지를 보증하기 위한 검토, 감사 활동과 프로젝트 관리자와 다

른 관리층에 결과를 제공하는 것이 포함된다.

• 심사 결과

—독립된 시각에 의한 품질보증 활동이 안됨.

—품질보증 활동에 대한 인식이 부족하며, 일정을 우선으로 하고 있어 품질보증 활동이 간과됨.

3.2 개선 활동

심사를 받은 조직은 결과를 바탕으로 개선 계획에 따라 표 2와 같은 실행 체계를 구축하고 이행하였다. 우선 프로세스별로 각각의 프로세스가 확립되고 지속할 것이라는 팀장의 방침과 스폰스쉽을 설정하였다. 또한 프로세스별로 담당자를 선정하고 실행될 수 있는 체계를 구축하였다. 이를 위해 프로세스별로 계획서/절차의 Guide를 개발하고 도구와 양식을 개발 적용하였다. 그리고 검증 체계(공식 검토 및 테스트 활동 등)를 강화하였으며, 각종 기록의 유지와 프로젝트 이력 관리가 되도록 하였다. 또한 각 프로세스별로 메트릭을 개발, 적용함으로써 현황이 주기적으로 팀장과 프로젝트 관리자에게 보고되어 관리될 수 있도록 하였다. 프로세스 개선 추진 본부에서는 정기적인 내부 심사를 통해 개선을 점검하였으며, 지속적으로 프로세스를 개선하였다.

3.3 공식 심사

공식심사는 97년 11월에 4개 프로젝트를 대상으로 5일간에 걸쳐 실시되었다. 대상 조직은 전체 개발 Life Cycle을 보여줄 수 있고 조직을 대표하는 프로젝트로 구성하였으며, Level 2의 5개 프로세스(외주관리 제외)와 3단계중 4개 프로세스(조직 프로세스 지향, 조직 프로세스 정의, 소프트웨어 제품 엔지니어링, 동료 검토)를 심사 대상 프로세스로 하였다. 본 심사 전에 설문조사와 문서 검토가 실시되었으며, 본 심사는 SEI의 공인 Lead Assessor의 주도하에 피심사 조직의 6명의 심사원에 의해 실시되었다. 인터뷰는 8명의 팀장과 프로젝트 관리자를 포함하여 전체 약 130명 정도의 조직에서 35명을 선발하여 실시되었다.

공식 심사 결과는 전반적으로 2단계를 충족하는 것으로 나타났다. 발견된 2단계 프로세스의 주요 강, 약점은 다음과 같다.

1) 요구사항 관리

• 강점

—변경된 요구사항에 대한 일관성 유지 및 영향 분석이 용이함(요구사항 추적표, 구성관리 현황판 사용).

—모든 요구사항의 접수/검토/계획수립/이행완료 사항을 Web을 이용한 Tool을 사용하여 추적 관리 가능하고 고객이 항상 이를 확인할 수 있음.

—요구사항에 대해 정기적 또는 필요시 고객/프로젝트 리더와 같이 검토되고 있음.

—요구사항을 유형별로 관리하고 있음.

표 2 CMM 2단계 실행 체계

프로세스 / 요소	요구사항 관리	프로젝트 계획수립	프로젝트 추적	구성관리	품질보증
Commitment	각각의 프로세스가 확립되고 지속할 것이라는 팀장의 방침과 Sponsorship 설정				
Ability	프로젝트관리자 요구사항관리담당자 구성관리담당자	프로젝트 관리자		구성관리 담당자	품질보증 담당자
Activity	요구사항 관리절차	프로젝트 계획서			품질보증 계획서
	관련 기록 유지				
Measurement	KPA 상태를 파악하고 개선할 수 있는 측정치 선정				
Verification	팀장	KPA 현황을 파악 할 수 있는 보고 체계와 피드백 체계 수립			
	프로젝트관리자	KPA 현황을 파악 할 수 있는 보고 체계와 피드백 체계 수립			
	품질보증담당자	정기적인 감사/검토/평가 실시 및 보고			전문가 검토

• 약점
—요구사항 관리를 위한 측정치가 표준화되어 있지 않음(예, 요구사항 누적 수, 요구사항 유형별 통계 등).

2) 프로젝트 계획 수립
• 강점
—프로젝트관리를 위하여 Work Breakdown Structure 분리시 Task를 포함하여 개인 수준까지 정의하여 사용하고 있음.
—프로젝트 계획은 조직의 프로젝트 관리 절차를 준수하여 소프트웨어 개발 계획서, 품질보증 계획서, 구성관리 계획서 등으로 상세히 작성하여 사용함.
—전사 차원의 마스터 플랜하에서 프로젝트를 수행하고 있어 프로젝트의 일관성이 있음.
• 약점
—과거/현재 프로젝트 데이타를 문서화하여 사용하고 있으나 재사용될 수 있도록 표준화가 되어 있지 않음.

3) 프로젝트 추적
• 강점
—프로젝트 진행시 고객과 함께 수행하며, 프로젝트의 성과나 결과의 검토를 위한 활동으로 단계말에 Workshop과 프로토타이핑을 수행함.
—프로젝트 관리의 통합화로 프로젝트 관리자가 프로젝트의 진행 상황을 체계적이고 전체적인 시각에서 파악하고 있으며, 시정조치하는 기준까지도 마련되어 있음.
• 약점
—프로젝트 규모에 대한 기준이 프로젝트별로 달라 데이타 수집과 Consensus(공동의 이해)가 어려움.
—프로젝트 관리 데이타들이 프로젝트별로 관리되고 있으나, 타 프로젝트에 활용할 수 있도록 관리되고 있지 않음.

4) 구성관리
• 강점
—구성관리를 위한 별도의 도구를 이용하여 구성항목을 관리하고 있으므로 구성항목 변경 및 구성관련 정보에 대한 팀원들의 정보 공유를 용이하게 하고 있음.
—구성관리 항목을 별도의 라이브러리 시스템으로 구축하여 물리적으로 구분하고 별도의 라이브러리 담당자를 두어 운영함으로써 구성항목의 무결성을 유지하고 있음.
• 약점
—일부 프로젝트에서 구성 항목의 변경에 대해 회람을 통해 공유하고 있으나, 일부 내용이 누락됨.

5) 품질보증
• 강점
—프로젝트에 대한 품질보증을 이행하고 조정 및 책임을 지는 그룹과 독립된 외부 전문가 그룹이 있음.
• 약점
—각종 품질보증 활동에서 수집된 데이타의 통계적 활용에 대한 실적이 없음.
—품질보증활동의 성과 측정 지표가 없음.

3.4 사후 활동

프로세스 개선을 위해 CMM 이행을 하였으나 프로세스 개선에 대한 정량적 목표가 없었고, 프로세스 개선이 비전, 사업 목표, 니즈와 명확한 연결을 갖지 못해 공식 심사 후 프로세스 개선에 따른 정량적 효과 측정이 어려웠다. 그럼에도 불구하고 CMM 이행을 통해 다음과 같은 효과를 거둘 수 있었다.

조직 차원에서는 기존 ISO/TickIT 품질시스템에 프로세스를 강화한 실질적인 Level-Up이 이루어졌으며, 프로젝트 차원에서는 팀원들의 인식 변화가 큰 성과였다. 즉, 프로젝트 관리 개념과 통제 능력 향상에 대한 관리자의 인식 변화와 수행 능력 향상을 위한 작업자의 소프트웨어 엔지니어링 개념 및 품질 보증과 검증 활동의 필요성에 대한 인식 변화가 이루어졌다. 그리고 추진 본부에서는 프로세스 심사 Framework 구축과 심사 및 진단 능력 향상을 가져왔다. 그러나 무엇보다도 큰 성과는 조직의 강, 약점을 인식하고 개선의 범위와 우선순위를 인식하였다는 것이다.

본 사례 조직은 이상과 같은 문제점을 보완하고 성과를 확산하기 위해 평가 후 다음과 같은 계획을 수립하여 이행하였다.

1) CMM 2단계 프로세스 표준화

공식 심사시 발견된 문제를 보완하고 2단계

프로세스 및 하위 프로세스를 추가 개발, 표준화하여 모든 조직에서 쉽게 활용하도록 함.

2) 전사 차원의 프로세스 정비

프로세스 개선 활동의 조정 역할을 수행할 수 있는 조직을 구성하여 지속적으로 전사 차원의 표준 소프트웨어 프로세스 자산(프로세스 표준, 프로젝트 수행 Tailoring Guide, 메트릭스 DB, 표준 문서/양식)을 정비하도록 함.

3) 내부인정제도 개발 및 내부평가 활성화

팀원들은 자주 진척에 대한 증거를 보지 못하면 쉽게 실망할 수 있다 따라서 동기 부여를 위해 CMM 단계내에서도 각 프로세스별로 인정할 수 있도록 그림 12와 같이 각 단계를 다시 1~3개의 단계로 구분하였다. 이는 프로세스의 우선 순위 부여를 통해 조직 또는 프로젝트에서 프로세스 개선을 위해 초점을 맞추어야 할 영역을 명확히 하는 한편, 현 능력 수준의 파악을 통한 개선 목표 수립을 용이하게 하기 위함이다.

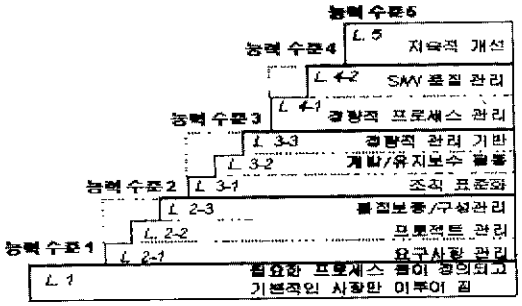


그림 12 발전된 프로세스 능력 심사모델[12]

4) 성과측정을 위한 메트릭스 프로그램 구축

전사적으로 주요 메트릭을 선정, 관리할 수 있도록 하고 이를 통해 프로젝트 이력을 관리함으로써 공유가 가능하도록 함.

그러나 이러한 활동도 결국 체계적인 개선 프로그램과 맞물려 돌아가지 않으면 성과가 없으므로 SEI의 IDEAL 모델을 실정에 맞게 커스터마이징한 소프트웨어 프로세스 개선 프로그램을 구축하여 이행하였다. 각각의 단계에 대한 주요 목표 및 산출물과 활용 내용은 다음과 같다.

● Initiating

—목적

- 초기 개선 기반 조직이 설립되며 각각의 책임과 역할이 정의됨
- 초기 SPI 활동에 대한 상위 수준의 소프트웨어 프로세스 개선계획 및 일정 개발

—주요 산출물

- 프로세스 개선 기반 조직, 의사소통 계획, 프로세스 개선 프로그램

—활용 내용

- 조직의 목표 수립시 프로세스 개선 계획도 함께 수립하도록 함.

● Diagnosing

—목적

- 조직의 현재 상태에 대한 기준선을 수립하기 위한 평가 실시
- 평가로부터 얻은 결과 및 권고사항 전달

—주요 산출물

- 발견사항(능력 Baseline)
- 권고사항

—활용 내용

- 내부인정 제도에 따라 내부심사 활용

● Establishing

—목적

- 개선 활동을 통해 해결하기로 결정한 이슈에 대해 우선 순위를 부여하고 해결안을 만들기 위한 전략을 개발
- 착수 단계에서 정의한 전반적 목표로부터 측정 가능한 목표를 도출

—주요 산출물

- 프로세스 개선 전략 활동 계획

—활용 내용: 성과 측정을 위한 메트릭스 프로그램을 수립하여 이행

● Acting

—목적

- 진단단계에서 발견된 개선사항을 처리하는 해결안이 개발되고 시험되어 조직에 전파

—주요 산출물

- 시범 적용 계획서

—활용 내용: 프로세스 실행팀 운영

● Learning

—목적

- 경험을 수집하여 프로세스 개선 프로그

램에 사용된 전략, 방법, 기반 조직에 대한 평가를 수행하고 프로세스를 재조정

—주요 산출물

• 경험 분석 및 프로세스 개선에 반영

• 다음 주기 상위 수준 목표

—활용 내용 : 전사프로세스 자산 변화 관리

이상과 같은 결과에 따라 각 팀의 프로세스 능력은 지속적으로 개선되고 있는 것으로 나타났다. 그림 13은 소프트웨어 프로세스 개선 프로그램을 이행중인 6개 프로젝트에 대한 내부 심사 결과로써 상당한 개선을 보여준다.

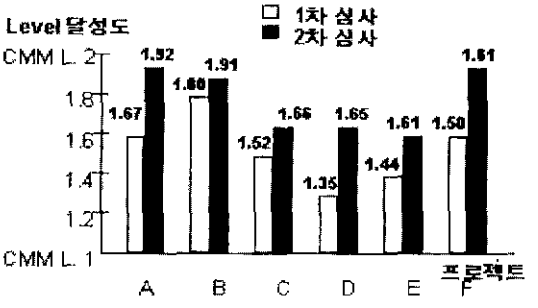


그림 13 프로젝트 내부 심사 결과

* 달성도는 CMM 2단계 충족을 위한 전체 활동중 수행된 활동의 비율로 계산함.
* 2단계 프로세스중 외주관리는 제외함.

표 3은 A, B 프로젝트에 대한 프로세스별 심사 결과이다.

표 3 프로세스별 심사 결과

프 로 세 스	A 프로젝트		B 프로젝트	
	1차	2차	1차	2차
요구사항 관리	70%	98%	68%	87%
프로젝트 계획수립	83%	88%	98%	99%
프로젝트 추적	66%	92%	84%	92%
품질보증	59%	94%	86%	87%
구성관리	55%	88%	65%	92%

* 각 프로세스별로 전체 활동중 수행된 활동의 비율임.

4. 결 론

한때 diet가 몸무게를 줄일 수 있는 유일한 방법인양 유행된 적이 있었다. 그러나 자료

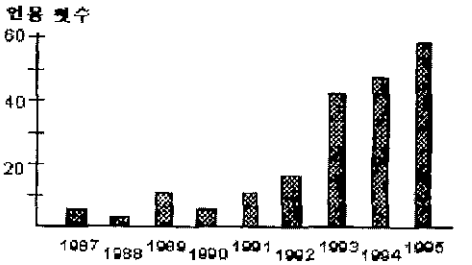


그림 14 잡지/저널에서 소프트웨어 프로세스 개선 기사로 인용된 횟수

[13]에 따르면 diet한 사람의 95%가 diet 후 1년 내에 뺀 몸무게를 다시 얻었으며, 덜먹고 운동하는 방식, 즉 프로세스를 변화시킨 사람은 60%가 몸무게를 줄일 수 있었다.

최근 소프트웨어 프로세스 개선에 대한 관심이 증가하고 있다는 것은 분명하다. 그림 14에서 보듯이 소프트웨어 프로세스 개선에 대한 기사가 계속해서 증가하고 있다[14].

본 논문에서 소개한 CMM도 처음에는 미국 방성의 필요에 의해 개발되었지만 지금은 전세계에서 다양한 형태로 채택되어 적용되고 있다. SEI의 Process Maturity Profile of the Software Community 1996 Update에 따르면, 1996년 3월까지 SEI로 보고된 560번의 평가(1995년 말까지 477개의 조직에서 수행됨) 중 17% 정도가 미국 이외의 지역이다[15].

CMM은 소프트웨어를 이해하고 개선하기 위해 유용한 도구이지만 CMM을 활용한 모든 조직이 프로세스를 개선한 것은 아니다. SEI의 자료에 따르면, 지난 9년 동안 SEI에서 실시한 515번의 CMM 평가에서, 평가를 받은 440개의 조직중 70%인 308개 조직은 시정조치를 이행하지 못하였으며, 14%만이 소프트웨어 프로세스 개선 프로그램을 이행한 후 재평가를 받았다. 재평가를 받은 조직중에서도 24%는 CMM 단계의 변화가 없었다.

향후 많은 조직에서 필요성에 의해 CMM을 사용하겠지만 여기에는 주의할 점이 있다. CMM은 모델이다. 모델이란 공동의 언어로써 시각을 나눌수 있게 하고 행동의 우선 순위를 결정하는 프레임웍을 제공할 수 있다. 반면, 모델은 현실 세계를 단순화시킨 것이므로 모델을 제대로 사용하기 위해서는 반드시 사업 목표에 맞

춰 해석하고 적용해야 하며 적절한 판단을 하여야 한다.

Humphrey는 성공적인 프로세스 변화를 위해서는 6가지 기본 원칙을 제시하고 있다[6].

첫째, 프로세스 변화는 반드시 경영진에서 시작되어야 한다. 경영진은 변화에 대한 동기부여와 함께 계속적으로 자원을 제공하여야 한다.

둘째, 모든 사람이 참여하여야 한다. 소프트웨어 엔지니어링은 팀 노력이므로 모든 사람이 참여하지 않는다면 프로세스 개선의 효과는 반감되고 심지어 개선에 장애가 될 것이다.

셋째, 효과적인 변화를 위해 목표와 현재 프로세스에 대해 이해하고 있어야 한다. Humphrey는 현재의 위치를 모른다면, 어떠한 지도도 도움이 되지 못한다(*If you dont know where you are, a map wont help*)[6]라고 하였다.

넷째, 변화는 지속되어야 한다. 소프트웨어 프로세스 개선은 한번에 끝나는 것이 아니며 살아있는 프로세스에 의해 계속해서 배우고 성장되어야 한다.

다섯째, 소프트웨어 프로세스 변화를 위해 의식적으로 노력하고 정기적으로 강화하여야 한다.

마지막으로 소프트웨어 프로세스 개선을 위해 많은 투자가 필요하다. 개선을 위해 계획을 수립하고 사람을 투입하고 관리하고 자금을 지원하여야 한다. 총 104번의 평가에서 2번 이상 평가를 받은 48개 조직의 경우, 1단계에서 2단계로 성숙하는데 약 25%의 조직이 3년 이상 소요되었으며, 약 50% 이상은 2년 이상 소요되었다. 또한 2단계에서 3단계로 성숙하는데 약 50% 이상의 조직이 2년 이상 소요되었다[16].

본 논문에서 CMM에 따른 프로세스 개선 사례를 소개하였다. 앞으로 CMM을 통해 프로세스 개선을 하려는 조직이 있다면, 본 사례가 시행착오 감소는 물론 체계적인 프로세스 개선 활동에 상당한 도움을 줄 수 있을 것이다.

참고문헌

[1] 유해영, 소프트웨어 공학, 휘중당, 1996.

[2] 이주헌, 실용 소프트웨어 공학론, 법영사, 1995.

[3] K. H. Möller and D. J. Paulish, *Software Metrics: A practitioners guide to improved product development*, IEEE Computer Society Press, 1993.

[4] Howard Rubin, "Software Process Maturity", Computer Channel Inc, 1993.

[5] James Herbsleb, Anita Carleton, James Rozum, Jane Siegel. and David Zubrow, "Benefits of CMM-Based Software Process Improvement: Initial Results", CMU/SEI-94-TR-13, 1994.

[6] Watts S. Humphrey, *Managing the Software Process*, Addison-Wesley Publishing Company, 1990.

[7] Bob McFeeley, "IDEAL: A Users Guide for Software Process Improvement", CMU/SEI-96-HB-001, SEI, February 1996.

[8] Fagan, Michael E., "Design, and Code Inspections and Process Control in the Development of Programs", IBM-TR-00.73, June 1976.

[9] Crosby, P. B. *Quality Is Free*, New York; McGraw-Hill, 1979.

[10] Mark C. Paulk, Bill Curtis, Mary Beth Chrissis and Charles V. Weber, "Capability Maturity Model for Software", SEI, February, 1993.

[11] Donna K. Dunaway and Steve Masters, "CMM-Based Appraisal for Internal Process Improvement Method Description", CMU/SEI-96-TR-007, SEI, April 1996.

[12] 정학종, 이민재, 조창현, "S/W 프로세스 심사 모델에 관한 연구", SQMS 98 제2회 소프트웨어 품질관리 심포지엄 논문집, 1998.

[13] R, Radice, *CMM Workshop*, Software Technology Transition, 1997.

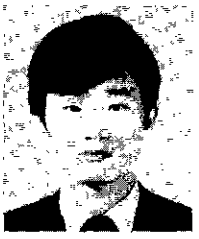
[14] Robert B. Grady, *Successful Software Process Improvement*, Hewlett-Packard

Company, 1997.

[15] Howard Rubin, *IT Metrics Strategies,* Computer Channel Inc, December 1996.

[16] Will Hayes and Dave Zubrow, "Moving On Up: Data and Experience Doing CMM-Based Process Improvement", CMU/SEI-96-TR-008, SEI, August 1995.

정 학 종



농수산부 통계관실, 금성전선 전산실에 근무하면서 응용 프로그램, 시스템 프로그램 및 통신 프로그램등에 대한 경험을 바탕으로, 1987년부터 EDS(미국)의 시스템 개발 방법론 개발에 참여하여 국내에 도입하여 전파하였다. 현재 (주)LG-EDS시스템의 품질경영 팀장으로 품질경영시스템 운영 및 소프트웨어 품질감사와 CMM에 의한 프로세스 평가 등을 담당하고 있다. 주요 관심분야로는 소프트웨어의 품질과 프로세스 평가, 정보시스템 성과 평가 등이다. SPICE 한국LTC에서 프로세스 심사원 교육을 담당하고 있으며, SPICE 심사원으로도 활동하고 있다.

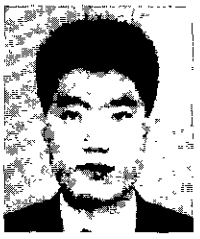
E-mail : hjjeong@lgeds.lg.co.kr

김 도 균

성균관대학교에서 컴퓨터감사 전공으로 석사학위를 취득하였으며, 현재 (주)LG-EDS시스템에서 CMM에 근거한 프로세스개선 활동과 프로젝트 품질보증업무를 담당하고 있다. 주요 관심분야로는 메트릭스와 소프트웨어 품질평가 등이며, SPICE 심사원 자격을 가지고 있다.

E-mail : dokyunkim @ lgeds.lg.co.kr

박 남 직



고려대학교에서 통계학을 전공, (주)LG-EDS 시스템에서 한전 판매 SI프로젝트의 품질보증활동과 S/W프로세스 평가를 수행하였으며, 현재 S/W프로세스 개선 활동과 프로젝트 품질보증업무를 담당하고 있다. 주요 관심분야로는 S/W프로세스 평가와 정보시스템 성과측정 등이며, SPICE 심사원 자격을 가지고 있다.

E-mail : njpark@lgeds.lg.co.kr