□ 기술해설 □

프로세스 모델 기반 개발 방법과 프로세스의 평가

한국소프트웨어품질연구소 양해술*
강원대학교 이용근** · 이하용***

목 차

1. 서 언

정보화 사회의 발전에 따라 소프트웨어의 품질과 생산성의 향상은 가장 중요한 과제중의 하나이다. 또한 정보화 사회의 진전에 따라 소프트웨어에 부과된 역할은 더욱 더 그 중요성이 증대되었으며 고도의 다양한 기능을 가지는 고품질의 소프트웨어를 효율적으로 개발·생산하는 것이 필요하게 되었다. 종래에는 주로 제품의 신뢰성, 안전성에 중점을 두고 연구가 이루어져 좋은 성과를 이루어 왔으며 이와 같은 소프트웨어의 품질 향상은 소프트웨어가 존재하는 한 피할 수 없는 과제이다.

또한, 소프트웨어 품질 관리의 경험에 의하면 제품 품질의 수준은 프로세스의 시작과 조직의 좋고 나쁨에 강하게 의존하는 것으로 인식되었고, 그 결과 소프트웨어 프로세스의 개선이 중요한 과제가 되어 왔다. 여기에서 소프트웨어 프로세스란 소프트웨어의 개발, 운용, 보수에 관한 작업과 그 연결 및 그것들에 영향을 미치는 각종 요인(사람, 기술 등)이라고 정의하였다. 그리고 프로세스를 평가하고 개선하기 위해서 프로세스가 가지는 기술적인 측면 뿐만 아니라 인간적 측면(인간 요소 : Human Factor)에 관한 문제점에 대해 해결하는 것도 커다란 비중을 차지하고 있다. 이와 같은 프로세스 모델에 기반을 둔 개발방법과 프로세스 평가의 지속적인 개선이 필요하다고 본다. 그리고 보다 좋은 소프트웨어 제품과 서비스를 개발하고 제공할 수 있는 프로세스는 어떻게 해야 하는가에 대해서도 연구함과 동시에 프로세스 자신의 기능과 성능의 레벨 및 프로세스를 가동하기 위해 필요한 조직의 구성과 인재의 종류 및 능력 등에 대하여 총합적으로 평가 가능한 프로세스 평가시스템의 연구와 실용화의 필요성이 강하게 지적되고 있다.

따라서 본 고에서는 먼저 프로세스 모델 기반 개발 방법에 대해 살펴보고, 소프트웨어 제품 평가와 프로세스 평가 그리고 프로세스 평

* 종신회원
** 정 회 원
*** 학생회원

가 모델의 종류에 대해 살펴보기로 한다. 또한 프로세스 품질과 제품 품질과의 관계, 최적 프로세스 평가 모델의 필요성에 대해 알아보기로 한다.

2. 프로세스모델 기반 개발 방법

오픈화, 다운사이징의 움직임에 호응하여 클라이언트/서버 시스템(CSS : Client/Server System) 개발의 요구가 증대하고 있다. 그러나 아직까지 클라이언트/서버 환경의 소프트웨어를 개발하기 위한 방법론에 대한 체계화된 표준은 없으며 여기에 대해서는 현재 지속적인 연구가 이루어지고 있다. 클라이언트/서버 시스템 구축의 목적은 크게 다음과 같이 2가지로 분류한다.

첫째, 업무의 흐름을 수정하고, 완전히 새로운 패러다임의 시스템을 실현한다.

둘째, 시스템을 가능한한 신속하게 낮은 비용으로 실현한다.

즉, 클라이언트/서버 시스템 개발의 프로세스는 종래의 시스템 개발의 프로세스와 전혀 다른 프로세스라는 것이 여러 가지 사례분석을 통하여 입증되었다. 또한 상기의 두가지 목적에 대응한 2개의 계층으로부터 만들어진 반복확대형의 개발 프로세스에 기초한 새로운 클라이언트/서버 시스템 개발 방법론을 개발하였다.

본 고에서는 먼저 방법론 개발까지의 경위와 구체적인 실시 기법에 대해 기술하기로 한다.

2.1 클라이언트/서버 시스템 개발의 실태

소프트웨어 개발 산업으로서 성립하고 있는 배경의 하나는 폭포수형 개발에 의한 개발공정의 분할이 널리 받아들여지고 있다는 것이다. 이에 따라 각 공정의 작업 내용과 필요한 요원의 기술 레벨에 관하여 개발 기관과 발주자간에 어느 정도 공통 인식이 확립되어 있다. 이와 같은 것은 종래의 메인프레임 소프트웨어 개발에서 생겨난 것이므로 클라이언트/서버 환경에서는 반드시 그렇지는 않는다. 지금까지의 예를 보더라도 그 원인은 클라이언트/서버 환경에서 요구되는 기술들이 종래와는 다르다는 것이며 필요한 기술들의 예를 들면 다음과 같다.

① PC나 WS의 하드웨어와 운영체제
② LAN 관련 기술
③ 호스트 컴퓨터와 클라이언트/서버 환경과의 접속
④ 클라이언트/서버 환경에서 활용가능한 개발 도구
⑤ 윈도우 시스템
⑥ 그래픽 유저 인터페이스 개발
⑦ 프로토타이핑 개발과 평가
⑧ 클라이언트/서버 환경에 적합한 모델링 기법
⑨ 데이터 보관장소와 프로세스 처리장소와의 최적 배치

또한, 클라이언트/서버 시스템 개발부문의 핵심은 지금까지 COBOL에서의 메인프레임 소프트웨어 개발을 중심으로 축적해온 기술력이 클라이언트/서버 시스템의 개발에서는 활용되지 못한다는 것이다. 워크스테이션(WS)이나 PC에 의한 소프트웨어 개발에서 어떤 점이 메인프레임의 소프트웨어 개발과정에서 사용해온 종래의 기법과 다른 것인가? 즉, 어떤 점에서 종래의 기법이 사용되지 않는지를 살펴보았더니 종래의 기법으로는 우수한 개발자가 실패할 확률이 높다는 것이다. 그 차이점을 구체적으로 서술하면 표 1과 같다.

이 차이점을 전부 습득하고 있는 기술자는

표 1 클라이언트/서버 소프트웨어 개발과 메인프레임 소프트웨어 개발의 차이점

구 분	클라이언트/서버 소프트웨어	메인프레임 소프트웨어
설계의 시점	· 사건에 대하여 프로세스가 발생하는 방식에서 설계	· 데이터베이스의 처리를 중심으로 프로세스를 설계
개발 프로세스	· 설계와 개발이 분리되어 있지 않음 · 명세서없는 개발도 있음	· 설계와 개발이 분리 · 명세서 작성이 원칙
필요한 지식	· 하드웨어에서 OS, DB, 네트워크, 개발지원 도구에 관한 폭넓은 지식	· 중간 소프트웨어가 완비되어 있지 않기 때문에 상세한 지식은 불필요

아직 많지 않으며 각각의 기술에 있어서도 그것을 어떤 체계로 활용하는가의 확립된 공통의식이 존재하지 않는다. 그 때문에 클라이언트/서버 환경에서 소프트웨어 개발은 소규모 개발 대상에 대하여 독자적인 개발 수법으로 실시하고 있는 실정이다.

2.1.1 클라이언트/서버 시스템의 개발자가 요구하는 개발지원

클라이언트/서버 시스템 개발부문에 어떤 지원이 있으면 핵심문제가 해결된다고 생각하는가? 설문조사 결과를 표 2에 분석하였다. 분석 결과로부터 프로젝트의 멤버는 클라이언트/서버 시스템 개발의 각 부분에서 하드웨어로부터 소프트웨어까지 다양한 사상을 거의 동시에 고려하고 있으며, 지원을 요구하고 있다는 점이다.

2.1.2 클라이언트/서버 시스템 개발 프로젝트 관리면에서의 특징

클라이언트/서버 시스템 개발 프로젝트의 실태를 문헌 베이스로 사례를 모아 분석한 결과 다음과 같은 일반적인 경향이 명확해짐을 알 수 있다.

- 프로젝트의 규모는 작고, 소수정예의 개발 요원이 담당하고 있다.
- 하드웨어로부터 어플리케이션까지 폭넓은 지식, 기술을 확보한 유능한 리더, 톱매니저나 사용자를 설득시키는 카리스마적인 리더가 존재하고 있다.

2.2 클라이언트/서버 시스템 개발의 BPR적 측면

클라이언트/서버 시스템 개발의 요건은 사용자의 만족도가 높은 시스템을 신속히 개발하는 것이다. Michael Hammer가 제창한 BPR(Business Process Engineering)의 사고방식에서 클라이언트/서버 시스템 개발 프로세스를 만족한다고 보면, 그것은 전문분화한 개발 팀이 순서적으로 스스로 분담업무를 완수하는 형태가 아닌, 한사람, 또는 하나의 팀이 일괄적으로 프로젝트를 실현하고, 사용자에 대하여 조기에 시스템을 제공하는 형태가 된다.

한편, 컴퓨터 시스템을 둘러싼 기술의 진보가 빠르지만 운영체제, 미들소프트웨어 등의 각 계층의 표준화는 곤란하다. 그리고 각각의 담당부서가 업무역할을 분담하여 시스템 개발을 하는 경우에서는 그 변화에 대응할 수 없다.

이와 같은 사고 방식으로부터 클라이언트/서버 시스템 개발이라고 하는 업무에 대해서는 프로세스의 근본적인 수정, 바꾸어 말하면 BPR이 절대적으로 필요하다고 하는 결론에 도달한다.

그러면 어떤 프로세스가 적절한가? 권장하는 개발방법은 2계층이며 각 계층이 반복확대형(나선형 모델)의 개발 프로세스이다. 다음에서 본 프로세스와 그것에 기초하고 있는 클라이언트/서버 시스템 개발 방법론에 대해 기술하기로 한다.

2.3 2계층 반복확대형 모델에 기초한 개발방법

2.3.1 2계층 프로세스

클라이언트/서버 시스템 구축에서 2종류의 목적이 있다는 것을 기술하였다.

첫째, 새로운 패러다임의 시스템을 구축하는 것보다 작업의 흐름을 변경한다.

둘째, 시스템을 신속히 낮은 비용으로 개발한다는 것이다.

이와 같은 이질적인 목적에 유연하게 대처하기 위해서는 하위의 2계층으로 분산시키면 좋

표 2 클라이언트/서버 시스템 소프트웨어 개발자가 요구하는 개발지원 환경

분 류	요 구
데이타베이스	· 유명한 관계데이타베이스를 전부 취급하는 간결한 방법으로 데이타베이스가 취급 · 자세한 데이타베이스의 제어, 처리도 실행
네트워크	· 유명한 네트워크 OS를 사용한 시스템을 쉽게 구축
유저인터페이스	· Motif, Windows 등의 화면을 쉽게 개발 · 가능하면 같은 방식으로 다양한 그래픽 유저 인터페이스가 취급
전체	· COBOL의 10배 이상의 생산성을 얻을수 있음

다.

상위의 계층은 가치있는 클라이언트/서버 시스템을 계획한다. 즉, 작업의 흐름을 수정한 계층으로서 여기에서는 고객만족도가 높은 클라이언트/서버 시스템을 제공하는 것을 목적으로 작업의 분석, 새 업무의 요구분석으로 시작하고 새로운 플랫홈에서의 시스템 구축 실현성의 확인까지를 행한다. 구체적으로는 클라이언트/서버 시스템 구축의 목적을 명확하게 하고 그 목적에 합당한 업무, 정보의 분산 방법을 검토하는 것이며, 이 경우 상당히 업무에 깊게 관련되기 때문에 사용자의 참가는 불가피하다.

하위의 계층은 클라이언트/서버 시스템을 신속하게 저가로 실현하는 부분으로서 여기서 중요한 것은 명세의 조기 확정과 불안 요인 특히 성능 문제의 조기 배제이다. 이것을 실현하기 위해서는 프로세스도 물론 중요하지만 이용하는 개발지원 도구의 선택 방법도 크게 영향을 미치므로 좋은 도구를 잘 사용하는 것이 성공의 열쇠가 된다.

2.3.2 반복 확대형 프로세스

클라이언트/서버 시스템의 두가지의 목적을 실현하기 위해 프로세스에 대해 생각해 보기로 한다. 먼저 기술한 2계층의 각각에 대해 유효한 프로세스란 가치있는 클라이언트/서버 시스템을 계획하는 계층이지만 여기에서는 클라이언트/서버 시스템 구축의 목적을 명확하게 하고 그 목적에 적합한 업무와 정보의 분산을 검토하지 않으면 안된다. 그러기 위해서는 어떠한 가치를 만들고 싶은가를 명확히 하고 그것을 실현할 때의 업무, 조직, 체제, 정보의 특성에 의한 다양한 제약 조건을 총합적으로 고려해야 하며 업무와 정보의 분산화기준을 설정하는 것이 필요하다. 그리고 그 기준으로 삼아 업무, 정보의 배치를 해 봄과 동시에 어느 레벨까지 클라이언트/서버 시스템의 구성을 설계하고 그 실현 가능성을 검토해 둔다.

이 일련의 프로세스는 사용자와 개발자가 협력하여 실행해야 하며 이 프로세스는 1회에서 마무리되는 것은 아니고 몇번 반복함으로써 검토 내용과 목적이 명확해지며 보다 실현성이 높은 계획이 된다.

다음에 클라이언트/서버 시스템을 조기에 적은 비용으로 개발하는 계층에 대해 생각해 보기로 한다. 이것은 2가지 관점으로 생각할 수 있으며 첫번째는 미확정된 부분을 조기에 확정한다고 하는 것과 두번째는 우선 필요한 최소의 기능만 실현하여 사용자에게 제공하고 후에 확장시켜 나가는 것이다.

전자의 프로세스는 우선 시스템의 명세를 사용자의 눈에 보이는 형태로 실현하고 사용자의 생각과의 차이를 명확히 하여 명세를 확정해간다는 것과 시스템을 구축할 때의 불안 요인, 예를 들면 신뢰성, 성능의 문제를 조기에 배제한다고 하는 것을 알 수 있다. 결국 사용자와 함께 시행 검토를 반복하면서 진행하는 방법이다.

후자에 대해서도 핵심이 되는 기능을 실현하면, 그것을 사용하면서 기능을 추가해 나가는 반복의 프로세스가 된다.

이와 같이 2계층은 유사한 프로세스를 채택해야 한다는 것을 알 수 있다. 클라이언트/서버 시스템 개발 방법론으로서는 2계층 반복확대형의 프로세스 모델을 채용하여 그림 1에 표현하였다.

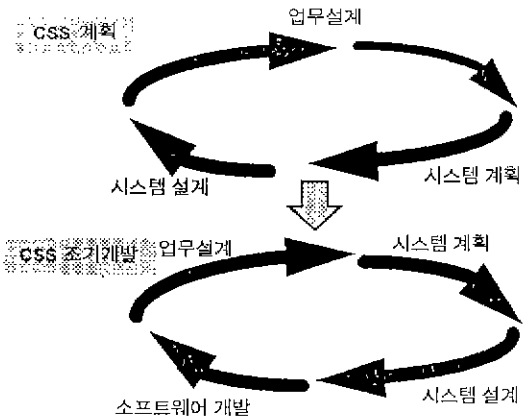


그림 1 2계층 반복 확대형 프로세스

상위 계층인 클라이언트/서버 시스템 계획은 가치있는 클라이언트/서버 시스템을 계획하는 프로세스이고 업무분석으로부터 클라이언트/서버 시스템의 구성 설계까지를 한다. 필요하면 성능 설계, 신뢰성 설계까지 행하고 시스템의 실현성을 확인한다.

하위 계층인 클라이언트/서버 시스템 조기 발달은 클라이언트/서버 시스템을 조기에 저가로 개발하는 프로세스이고 명세의 조기 확정과 성능의 조기 견적이 포인트라고 볼 수 있다.

3. 소프트웨어 제품 평가와 프로세스 평가 및 프로세스 평가 모델

3.1 소프트웨어 제품 평가와 프로세스 평가

소프트웨어 제품의 품질을 측정하고 평가하는 방식은 오래전부터 연구되어 왔고 특히 소프트웨어 제품의 품질 평가에 사용하는 품질 모델과 품질을 측정하는 척도로서의 품질 메트릭스가 부분적으로 개발되어 효과를 보고 있다. 이와 같은 활동을 기반으로 하여 소프트웨어 제품을 위한 품질평가 모델의 국제 표준화의 활동이 ISO/IEC JTC1 SC7/WG6에서 진행되고 있다.

한편 소프트웨어 제품을 개발하기 위해 프로세스의 기능, 구조 및 그 작업 내용을 정의하는 프로세스 모델에 대해서도 많은 연구가 진행되고 있으며 또한 SLCP(Software Life Cycle Process)라고 불리는 프로세스 모델 표준화의 검토도 진행되고 있다. 따라서 소프트웨어를 개발하기 위한 소프트웨어 프로세스의 전체적인 능력을 정량적으로 평가하기 위해 필요한 평가척도와 그 측정 평가방법으로 구성된 프로세스 메트릭스에 대해서는 명확하게는 정의되지 않고 있다. 소프트웨어 프로세스의 능력은 소프트웨어 제품의 품질을 평가한 결과를 기초로 간접적으로 평가되어 왔다. 소프트웨어 프로세스와 제품 평가의 개념은 그림 2와 같다.

1980년 후반부터 소프트웨어 프로세스가 가지는 총합적인 능력을 직접적으로 평가, 개선하도록 하는 경향이 강해지고 있으며 대표적인 소프트웨어 프로세스 평가모델로서는 다음의 4가지가 있다.

(1) CMM(Capability Maturity Model)

CMM은 미국 카네기대학 소프트웨어생산기술연구소(SEI)에서 1987년에 제창된 프로세스 성숙도 모델로 현재도 SEI에서 개선하기 위해 연구개발을 행하는 동시에 보급 활동을 계속하고 있다.

(2) SPICE(Software Process Improvement

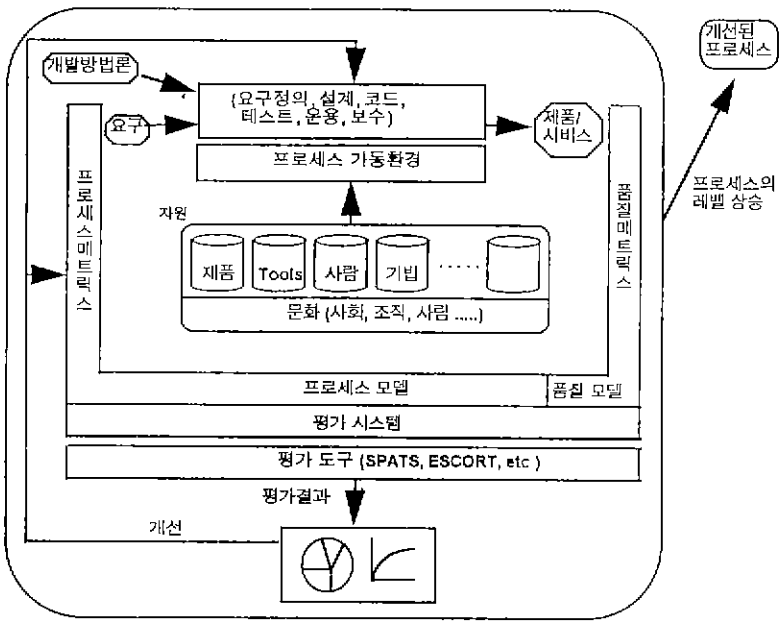


그림 2 소프트웨어 프로세스와 제품의 평가

and Capability dEtermination)

SPICE는 소프트웨어 프로세스의 개선과 능력의 결정을 목적으로 하는 프로세스 평가 모델이고, 동 모델을 기초로 한 국제 표준화의 활동이 ISO/IEC JTC1 SC7/WG10에서 미국, 유럽, 캐나다, 일본, 한국 등의 참가로 진행되고 있다.

(3) Bootstrap

Bootstrap은 유럽의 ESPRIT 프로젝트에서 연구되고 있는 소프트웨어 프로세스 평가방식이다.

(4) ISO 9000-3

ISO 9000-3은 하드웨어 및 시스템의 개발에 관련된 표준인 ISO 9001을 소프트웨어의 개발 프로젝트에 적용하기 위한 가이드로 개정이 추진되고 있다.

3.2 소프트웨어 프로세스 평가 모델

여기에서는 CMM과 SPICE의 2개의 소프트웨어 프로세스 평가 모델의 개요에 대해서 설명하기로 한다.

3.2.1 CMM(Capability Maturity Model)

조직, 작업 순서, 관리 방법, 기법, 도구, 환경 등을 총합적으로 포함한 것을 프로세스라고 정의하고, 프로세스를 실행하는 능력이 우수한 정도, 보다 좋은 제품과 서비스가 가능하다고 하는 사고방식을 기초로 하여 능력 성숙도 모델을 구축하였다.

(1) CMM의 평가척도

CMM은 표 3과 같이 5단계의 능력 레벨을 정의하고 있다. 또한, 이 능력 레벨 척도로서 프로세스의 성숙도를 계측·평가하는 방법을 포함시켜 제공하고 있다. 그림 3에 성숙도 레벨의 개념을 나타내고 있으며 어느 레벨에서 다음 레벨로 진행할때에는 다음 레벨에 도달하기 위한 개선 활동을 실행하여야 한다.

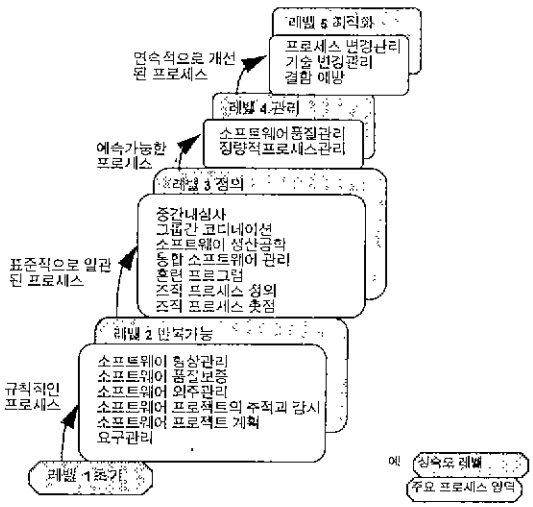


그림 3 성숙도 레벨의 개념

예를 들어 일본의 컴퓨터 메이커나 소프트웨어 개발기관 약 200사를 CMM으로 평가한 결과는 극히 소수의 기업이 제 2단계에, 대부분이 제 1단계에 있다고 보고하고 있다. 또한 미국기업의 경우도 마찬가지로 제 1단계가 80%, 제 2단계가 약 10%가 되고 있다. 제 3단계 이상은 극히 적다고 하는 결과로 되어 있다. 측정 방법이 그다지 엄밀하지 않으므로 평가 정밀도의 문제도 있지만 조직 전체의 능력을 평가하는 사고방식은 중요하다. 현재, CMM은 미국중심으로 널리 사용되는 프로세스의 평가와 개선에 도움이 됨과 동시에 이 결과에 기초한 CMM

표 3 CMM의 능력 레벨의 정의

초 기(레벨 1)	소프트웨어는 만들고 있지만 아무것도 관리하고 있지 않고, 데이터도 가지고 있지 않다.
반복가능(레벨 2)	같은 것을 반복적으로 행하고, 어느 정도 통계적 관리가 가능한 상태에 도달한 레벨
정 의(레벨 3)	프로세스의 작업이 정의되고 데이터에 의한 프로젝트 관리도 행해지고, 프로세스가 계속적으로 진보하는 기초가 정립된 상태
관 리(레벨 4)	프로세스의 데이터 수집을 자동화하고 데이터로 프로세스를 분석하고 수정한다. 실질적인 품질개선, 포괄적인 프로세스 개선이 가능한 상태의 레벨
최 적 화(레벨 5)	질적, 양적으로 큰 개선이 계속되는 상태에 도달하고 있는 레벨

자체의 개선도 진행하고 있다.

(2) 프로세스 카테고리의 커버 영역

CMM은 주요 프로세스 영역을 그림 4와 같이

① 관리(Management)

② 조직(Organization)

③ 기술(Engineering)

의 3가지의 광역적인 프로세스 카테고리로 분류하고 있다.

관리 프로세스 카테고리를 보면 레벨 2로 진행하기 위해서는 요구관리와 프로젝트의 계획, 추적과 감시, 외주관리 등의 기본적 관리 활동을 충실하게 하는 것이 중요하다는 것을 알 수 있다. 레벨 3에서는 통합 소프트웨어 관리, 그룹 간 코디네이트 등의 관리 활동이 중요하다. 레벨 4에서는 정량적 프로세스 관리, 레벨 5에서는 정상적으로 변화하는 환경에 대응할 수 있는 혁신적인 관리로까지 발전하는 형태로 되어 있다.

조직 프로세스 카테고리를 보면 레벨 3에서부터 시작하여 레벨 4에서는 정량적 프로세스 관리, 레벨 5에서는 계속적 프로세스 개선이 가능한 환경에서의 프로세스와 기술의 변경관리로 발전한다. 레벨 3, 4, 5를 통하여 조직이 성숙하기 위한 요건으로서 프로젝트의 포괄적인 책임의 명확화가 지적되고 있다.

기술 프로세스 카테고리는 요구분석, 설계, 코딩화 및 테스트와 같은 기술적인 활동을 포함하고 있다. 이것들은 모든 레벨에서 실행되지만, 레벨 3에서는 공학적 원리, 레벨 4에서는 통계적인 프로세스 관리 그리고 레벨 5에서는 계속적으로 측정 가능한 개선 등의 방향으로 진화하고 있다.

CMM과 SPICE에서는 프로세스 카테고리의 분류와 주요한 프로세스 영역과의 관계가 다르다.

3.2.2 SPICE

SPICE는 CMM의 발전 계보로서 연구가 진행되고 있다. 본 절에서는 SPICE에 대한 연구의 개요에 대해서 설명하기로 한다.

(1) SPICE의 모델

SPICE의 프로세스 모델은 소프트웨어 프로세스의 실행에 필요한 기본 작업을 기초로 계층적으로 구성된다.

이 프로세스 모델은 프로세스 전체를 다음 5가지 프로세스 카테고리, 즉 고객-공급자(CUS), 엔지니어링(ENG), 프로젝트(PRO), 지원(SUP), 조직(ORG)으로 분류하고 있다. 또한 각 프로

<table>
<tr><td colspan="2">프로세스 카테고리 / 레벨</td><td>관 리
소프트웨어 프로젝트의 계획, 관리 등</td><td>조 직
상급관리자 김모 등</td><td>기 술
요구분석, 설계, 코딩, 테스트 등</td></tr>
<tr><td rowspan="2">5</td><td rowspan="2">최적화</td><td colspan="2">기술변경관리</td><td rowspan="2">전위예방</td></tr>
<tr><td colspan="2">프로세스 변경관리</td></tr>
<tr><td>4</td><td>관 리</td><td colspan="2">정량적 프로세스 관리</td><td>소프트웨어 품질관리</td></tr>
<tr><td>3</td><td>정 의</td><td>통합 소프트웨어 관리
그룹간 코디네이션</td><td>조직프로세스 촛점
조직프로세스 정의
훈련 프로그램</td><td>소프트웨어생산공학
중간내 심사</td></tr>
<tr><td>2</td><td>반복가능</td><td>요구관리
소프트웨어 프로젝트계획
소프트웨어 프로젝트의 추적과 감시
소프트웨어 외주관리
소프트웨어 품질보증
소프트웨어 형상관리</td><td></td><td></td></tr>
<tr><td>1</td><td>초 기</td><td colspan="3">초기 경우의 프로세스</td></tr>
</table>

그림 4 주요 프로세스와 프로세스 카테고리의 대응

세스 카테고리는 복수개의 프로세스라고 불리는 작업군으로 구성된다. 그리고 각 프로세스는 복수개의 기본작업 항목(Base Practice)의 집합으로 구성된다.

표 4에 프로세스 카테고리의 특징과 그것을 구성하는 프로세스, 표 5에 프로세스를 구성하는 기본 작업 항목, 그림 5에 프로세스 모델의 구조를 나타내었다.

프로세스 카테고리, 프로세스 및 기본 작업 항목과 같은 3단계의 계층적 관계를 가지는 프

표 4 프로세스 카테고리와 프로세스

프로세스 카테고리	고객-공급자	공 학	프로젝트	지 원	조 직
특 징	· 고객에의 대응 · 소프트웨어의 개발이나 이행의 지원, 적절한 운용이나 사용 방법의 제공	· 시스템, 소프트웨어 제품, 사용자 문서의 명세화, 실행, 보수	· 프로젝트 확립 · 만족스런 제품이나 서비스를 작성하기 위해 자원의 조정과 관리	· 다른 프로세스 카테고리의 프로세스 실행 지원	· 조직의 비지니스 목표의 확립 · 조직의 비지니스 목표달성을 원조하는 프로세스, 제품, 자원의 개발
프로세스	1. 소프트웨어 제품/서비스의 획득 2. 계약의 체결 3. 고객의 요구 파악 4. 고객과의 검토 5. 소프트웨어의 도입 6. 소프트웨어의 운용 7. 고객 서비스 8. 고객 만족도	1. 시스템 요구 설계 개발 2. 소프트웨어 요구 정의개발 3. 소프트웨어 설계 개발 4. 소프트웨어 설계 실행 5. 소프트웨어 통합 테스트 6. 시스템 통합 테스트 7. 시스템 소프트웨어 보수	1. 라이프사이클의 설계 2. 프로젝트계획 3. 프로젝트팀의 발족 4. 요구정의관리 5. 품질관리 6. 위기관리 7. 자원과 스케줄의 관리 8. 청부업자관리	1. 문서작성 2. 구성관리 3. 품질보증 4. 문제해결 5. 중간내심사	1. 비지니스의 설계 2. 프로세스의 정의 3. 프로세스의 개선 4. 훈련의 실시 5. 재이용의 실현 6. 소프트웨어 공학 환경의 제공 7. 작업설비의 제공

표 5 프로세스와 기본 작업 항목

프로세스 카테고리	고객-공급자	공 학	프로젝트	지 원	조 직
프로세스	1. 소프트웨어 제품/서비스의 입수	1. 시스템 요구명세과 설계의 개발	1. 프로젝트 라이프사이클의 계획	1. 문서작성	1. 비지니스의 설계
기본작업항목	1. 필요성의 인식 2. 요구의 정의 3. 구입전략의 준비 4. 요구제안서의 준비 5. 소프트웨어 제품 공급자의 선정	1. 시스템 요구명세서의 작성 2. 시스템 구조의 기술 3. 요구명세의 할당 4. release 전략의 결정	1. 제품개발을 위한 옵션의 평가 2. 소프트웨어 라이프 사이클 모델의 선택 3. 활동 및 태스크의 기술 4. 태스크의 순서설정 5. 활동의 문서화	1. 문서형식 결정 2. 문서 작성 3. 문서 검사 4. 문서 배포 5. 문서 관리	1. 전략 비전의 확립 2. 비전의 전개 3. 품질문화의 확립 4. 통합 팀의 발족 5. 동기의 제공 6. 캐리이 계획의 정의
프로세스	2. 계약의 체결	2. 소프트웨어 요구명세의 개발	⋮	⋮	⋮
기본작업항목	1. 최종계약전의 검토 2. 계약 교섭 3. 독립된 대행자로의 창구의 확정 4. 외주업자관리	1. 소프트웨어 요구명세의 결정 2. 소프트웨어 요구명세의 분석 3. 운용환경에 대한 영향의 결정 4. 고객과의 요구명세의 평가 5. 반복을 위한 요구명세	⋮	⋮	⋮

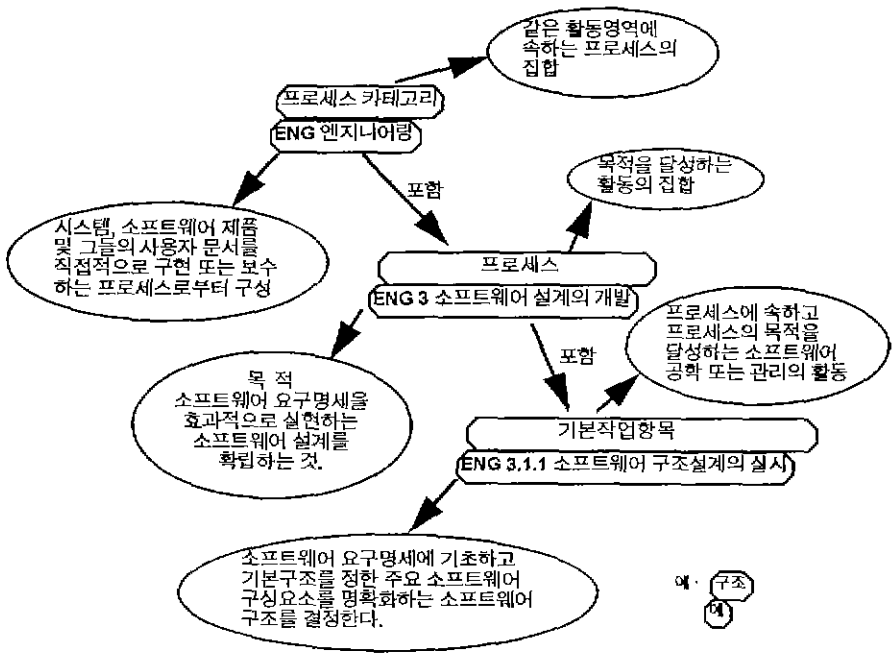


그림 5 SPICE의 프로세스 모델

로세스 모델은 일반적으로 다음과 같이 표현할 수 있다.

프로세스 전체 = {프로세스 카테고리 $-_i \mid i = 1, n$}

프로세스 카테고리 $-_i$ = {프로세스 $_{i,j} \mid j = 1, n_i$}

프로세스 $_{i,j}$ = {기본 프랙티스 $_{i,j,k} \mid k = 1, n_{ij}$}

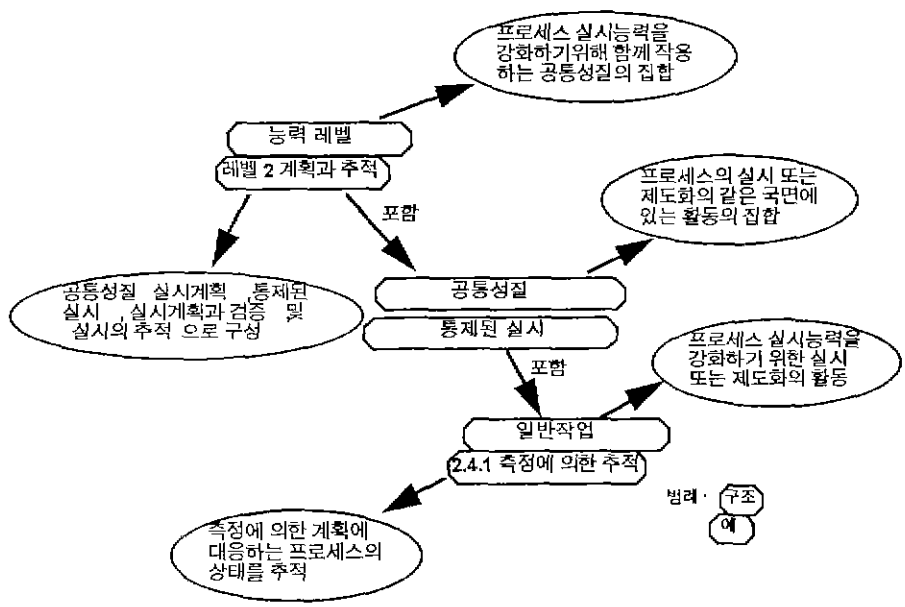


그림 6 SPICE의 프로세스 메트릭스

(2) SPICE의 프로세스 메트릭스

SPICE에서는 다음 6가지 능력 레벨을 가진다.

① Level 0 : 아무것도 하지 않는 레벨
② Level 1 : 비형식적으로 실행되고 있는 레벨
③ Level 2 : 계획하여 실행하고 있는 레벨
④ Level 3 : 적절하게 정의되어 있는 레벨
⑤ Level 4 : 정량적으로 관리되고 있는 레벨
⑥ Level 5 : 지속적으로 개선하고 있는 레벨

각 능력 레벨은 복수의 공통 성질, 그리고 각 공통 성질은 복수의 일반 작업(generic practice)으로 발전하며 이들의 관계는 표 6과 같고 프로세스 메트릭스의 구성을 그림 6에 나타내었다. 이 일반 작업이 능력레벨 판정의 체크리스트가 된다.

실제의 프로세스 평가에서는 프로세스 카테고리를 구성하는 프로세스 또는 기본 프렉티스와 일반 작업을 조합시킨 평가표를 작성한다. 그 다음에 실제의 프로젝트에 있어서 프로세스 성숙도 상황을 프로세스마다 혹은 기본 프렉티스마다 일반 작업의 내용과 대조하여 평가하고, 이 평점을 기입한다. 그리고 개발조직과 프로젝트의 특징을 감안하여 평가의 대상인 프로세스 전체 또는 프로세스 카테고리의 능력 레벨을 총합적으로 평가한다. 이상의 관계를 그림 7에 나타내었다.

(3) SPICE의 가이드라인

SPICE의 개념과 이용법을 명확히 하기 위해 현재 7가지의 가이드라인이 작성되어 있으며 각 가이드라인의 개요를 표 7에 정리하였다. WG10의 활동도 각 가이드라인 단위로 진행되고 있지만 현재의 시점에서는 BPG과 PAG가 기본적인 가이드라인이라고 생각하고 있다.

(4) SPICE의 시행 결과

SPICE 프로세스 평가 모델은 현재 국제적인 검토가 진행되고 있는 단계이고 참가 각국의 워킹 그룹에 의해 시행이 시작된 정도이다. 사용하기 쉽고, 문화적으로도 접근이 가능하며 동시에 효과적인 프로세스 평가 모델을 개발하는 데에는 연구, 시행, 개선을 반복할 필요가 있다.

표 6 프로세스 메트릭스의 관계

능 력 레 벨	공 통 성 질	일 반 작 업
#5 계속적으로 개선	· 조직능력의 개선	· 프로세스 효력의 목표의 확립 · 표준 프로세스의 계속적 개선
	· 프로세스 효력의 개선	· 원인의 분석으로 실시 · 정의된 프로세스의 계속적 개선
#4 정량적으로 관리	· 측정가능품질목표의 확립	· 품질목표의 확립
	· 객관적 실행관리	· 프로세스 능력의 결정 · 프로세스 효력의 사용
#3 적절히 정의	· 표준 프로세스의 정의	· 프로세스 표준화 · 프로세스 표준적합
	· 정의된 프로세스의 실행	· 적절한 프로세스의 사용 · 내부 프로세스의 실시 · 적절한 프로세스의 사용
#2 계획 추적	· 실행의 계획	· 자원배분 · 책임할당 · 프로세스의 문서화 · 도구의 공급 · 교육 · 프로세스의 계획
	· 훈련된 계획	· 계획 · 표준 · 절차의 설명 · 구관리의 실행
	· 실행의 검증	· 프로세스의 추종의 검증 · 작업제품의 감사
	· 실행의 추적	· 측정에 따른 추적 · 조정활동
#1 비형식적으로 실행	· 기본 작업항목의 실행	· 프로세스의 실행
#0 실행되지 않음	· 없음	· 없음

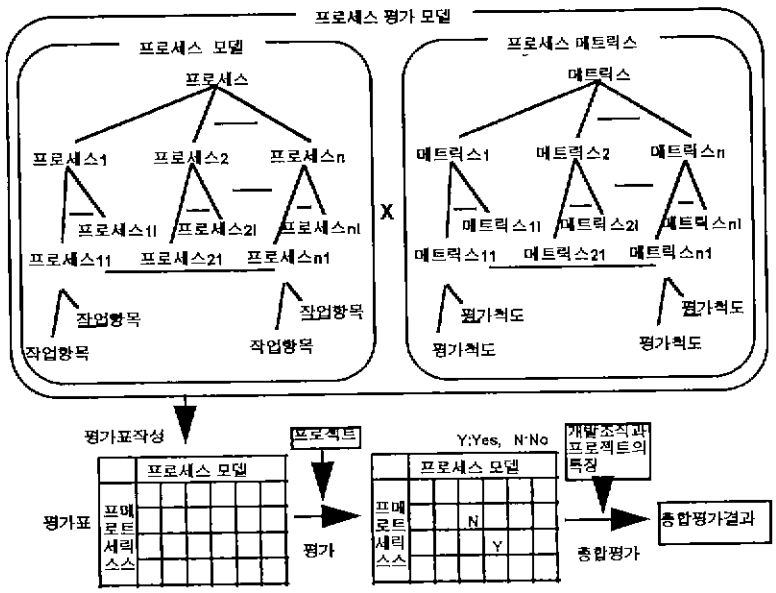


그림 7 프로세스 평가 시스템의 구조

4. 프로세스 품질과 제품 품질과의 관계

먼저 기술한 것과 같이 프로세스 성숙도에 의한 프로세스 평가의 기본적 사고방식은 "좋은 프로세스로부터 좋은 제품의 개발이 가능하다"고 하는 경험적 판정에 기초하고 있다. 한편, 개발을 해왔고 또는 개발을 의뢰하는 입장에서 보면 "좋은 제품을 만들기 위해서는 어떠한 프로세스가 필요한가?"라고 하는 의문이 생긴다. 프로세스 성숙도 모델에 의한 능력 레벨은 그 의문에 답변하는 것이지만 그것만으로는 충분하지 않다고 볼 수 있다. 프로세스 품질과 제품 품질의 관계에 대해 프로세스의 품질을 Q_{pc}(프로세스), 제품의 품질을 Q_{pd}(제품) 이라고 할 경우 이들의 관계는 형식적으로 다음식과 같이 표현할 수 있다.

$$Q_{pd}(\text{제품}) = Q_{pc}(\text{프로세스}(\text{시작, 계획})) \times Q_{rq}(\text{실제의 요구명세}) \times Q_{om}(\text{실제의 조직, 인재, 지식, 교육}) \times Q_{it}(\text{실제의 기술, 정보, 부품, 환경}) \times Q_{mt}(\text{동기})$$

위 식에서 알 수 있는 것과 같이 주어진 프로세스에 의해 개발되는 제품이 좋은 제품이기 때문에 프로세스가 가지는 능력 레벨 즉, 프로세스의 시작, 계획의 능력 레벨은 물론이고 이

표 7 SPICE의 가이드라인

프로덕트		개요
IG	Introductry Guide	도입 안내
PAG	Process Assessment Guide	assessment의 실시, 결과의 통합방법을 안내
PIG	Process Improvement Guide	assessment의 결과를 이용하여 프로세스를 개선하는 방법을 안내
PCDG	Process Capability Determination Guide	assessment 결과를 이용하여 개발조직의 능력평가를 행하는 방법을 안내
ATQG	Assessor Training & Qualification Guide	assessor 육성용의 훈련 프로그램의 개발, 제3자 assessor 자격을 위한 안내
BPG	Baseline Practices Guide	assessment의 기준으로서 바람직한 소프트웨어 활동을 규정
AI	Assessment Instrument	assessment를 행하고, 데이터를 추출하기 위해 필요한 도구의 요건, 예

외의 중요한 요인으로서 이 프로세스에 주어진 실제의 요구명세의 품질, 실제의 조직, 인재, 지식, 교육의 품질, 실제의 기술, 정보, 부분, 환경 등의 품질 및 개인, 팀 등의 프로젝트 구성 멤버가 가지는 동기의 품질이라고 할 수 이다. 즉, 제품의 품질은 위에 기술한 요인을 누적시켜 예측할 수 있다. 프로세스 능력은 제품의 품질을 결정하는 하나의 요인이지만 그 비중은 대단히 높다고 생각된다.

프로세스를 평가한 결과 얻어지는 프로세스의 성숙도 레벨과 실제로 개발한 제품의 품질과의 상관관계를 검토하는 것은 프로세스 평가 모델의 유효성을 결정한다는 점에서 중요한 연구과제의 하나이다.

5. 최적 프로세스 평가 모델의 필요성

현재, 국제적으로 검토가 진행되고 있는 SPICE에 의한 프로세스 평가 모델의 구성은 기본적으로는 폭포수형의 방법론을 사용하여 대규모의 소프트웨어 프로젝트를 수행하는 프로세스를 평가하는데에 적합하지 않다고 지적하는 사람이 많다. 그렇지만 실제의 소프트웨어 프로젝트에서는 나선형 개발, 추가형 개발, 객체지향 개발, 래피드프로토타이핑 또는 부품베이스의 개발 방법 등 여러 가지 프로세스 모델이 사용되어 왔다.

이것에 대해 SPICE의 개발자는 특히 폭포수형의 프로세스를 대상으로 하고 있는 것은 아니기 때문에 어떠한 프로세스의 평가에도 사용할 수 있다고 주장하고 있다. 개발하고자 하는 소프트웨어에 속해 있는 도메인, 규모, 품질, 신뢰성의 요구와 사용하는 기술 등에 의해 개발 방법이 다른 것은 당연하다고 생각한다.

위와 같은 주장은 개발대상의 소프트웨어의 속성, 규모 등을 고려한 최적의 개발 프로세스 모델의 구성, 바꾸어 말하면 프로세스 모델의 최적화가 제공되어 있다. 그러나 최적의 프로세스 평가 모델을 정의할 수 있는 최적화의 지침 등이 필요하게 된다.

이 방법에 의해 어느 도메인에 속하고 어느 범위의 규모를 가지고 유사한 시스템의 속성을 가지는 소프트웨어 개발 프로젝트에 최적인 프로세스를 구축하게 되며 정상적으로 능력을 개선시켜가는 것이 가능하다고 생각된다. 이상과 같은 과정을 표현하면 그림 8과 같다.

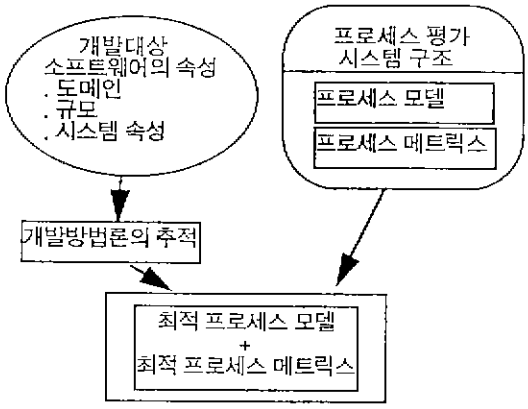


그림 8 최적의 프로세스 평가 모델

6. 결 언

지금까지 먼저 2계층 반복확대형 프로세스에 준하여 클라이언트/서버 시스템 개발 방법론에 대해서 기술하였다. 본 방법론은 앞으로 개발의 운용, 보수까지 포함한 프로세스로 구축해 갈 예정이다. 또한, 프로젝트 관리 방법, 품질관리 방법, 견적, 계약 방법에 대해서도 실 프로젝트에서의 적용사례를 분석하는 것에 의해 명확히 그 체계를 정립할 필요가 있다.

그리고 다음에 소프트웨어 제품평가와 프로세스 평가 모델에 대해 서술하였으며 프로세스 평가에 관한 앞으로의 연구과제를 열거하면 다음과 같다.

첫째, 프로세스 평가 시스템은 우선 프로세스의 자기 진단의 목적으로 이용하는 것이 바람직하다. 그리고 프로세스 평가 모델을 가능한한 많은 사람이 이해하고 활용하여 프로세스 개선에 유용하게 적용될 수 있는 노력이 필요하다.

둘째, 프로세스 평가의 결과를 축적하여 노우하우의 데이터베이스를 작성한다. 이것에 의해 프로세스 평가척도와 프로세스 평가 모델자체를 개선하는 것이 가능하다.

셋째, 프로세스 평가를 실시할 때에 부문 또는 프로젝트 책임자의 입장에서 보아 프로세스

카테고리 또는 프로세스로부터 평가하고 싶은가, 또는 어느 레벨 근처에 자신의 프로세스의 능력이 위치하고 있는가를 판단하여 그곳으로부터 평가를 시작하는 등의 연구를 함으로써 평가의 효율을 높이는 것이 가능하다.

넷째, 현재의 SPICE를 사용한 프로세스 평가는 상당히 시간이 걸린다고 지적되고 있다. 이 때문에 평가를 효율화하기 위한 평가 순서의 연구, 프로세스 평가 방법 자체의 간소화 및 프로세스 평가용 도구의 검토도 필요하다.

다섯째, 프로세스의 평가결과와 제품의 품질, 생산성의 평가결과와의 상관관계를 분석하는 것이 중요한 과제이다.

여섯째, 프로세스 평가모델을 표준화하기 위해서는 프로세스에 관계하는 다른 표준, 예를 들면 SLCP(Software Life Cycle Process) 등과의 내용의 연계도 필요하다.

이상으로 앞으로의 과제를 제시하였지만 프로세스 평가를 실시할 때 어느 정도까지 엄밀한 척도로 평가하는 것이 좋은가하는 문제가 있다. 예를 들면, 어느 작업을 실행하는 경우 작업 기술서를 개인 레벨까지 엄밀히 작성하지 않아도 그룹단위로 기술하여 나중에는 그룹장의 지시와 판단에 맡겨 작업을 진행시키는 방법이 좋다고 하는 생각도 있다. 실제 프로젝트 계획을 작성할 때에 그렇게 엄밀한 작업 계획이 가능할까하는 문제도 있다.

끝으로 보다 좋은 소프트웨어를 개발하기 위한 방편으로 소프트웨어 품질관리 활동과 더불어 프로세스 평가 시스템을 조화롭게 진행해나가는 것이 중요하다고 할 수 있다.

참 고 문 헌

[1] Curtis, B. et al., "L Process Modelling", CACM, Vol. 35, No. 9, pp. 75-90, 1992.

[2] Daskalanronakis, M. K., "Achieving Hifher SEL Levels", IEEE Software, Vol. 11, No. 4, pp. 17-24, 1994.

[3] Dion, R., "Process Improvement and Corporate Balance Sheet", IEEE Software, Vol. 10, No. 4, pp. 28-35, 1993.

[4] Grady, R. B. and Slack, T. V., "Key Lessons in Achieving Widespread Inspection Use", IEEE Software, Vol. 11, No. 4, pp. 46-57, 1994.

[5] Hammer, M., "Reengineering Work", Harvard Business Review, 1990. 8.

[6] Hasse, V., Messnarz, R., Koch, G., Kugler, H. J. and Decrinis, P., "Bootstrap : Fine-Tuning Process Assesment", IEEE Software, 1994. 7.

[7] Humphrey, W. S., "Managing the Software Process Improvement at Hughes Aircraft", IEEE Software, Vol. 8, No. 4, pp. 11-23, 1991.

[8] Mi, P. and Scacchi, W., "Process Integration in CASE Environments", IEEE Software, Vol. 9, No. 2, pp. 45-53, 1992.

[9] Osterweil, L., "Software Processes are Software Tool", Proc. 9th ICSE, pp. 2-13, 1987.

[10] Paulk, M. C., B., Chrissis, M B. and Weber, C. V., "Capability Maturity Model for Software", Version 1. 1, 1993. 2

[11] Perry, D. E. et al., "People, Organizations, and Process Improvement", IEEE Software, Vol. 11, No. 4, pp. 36-45, 1994.

[12] SPICE Process Improvement Guide, Document, 1994. 7.

[13] 飯塚悦功 : ソフトウエアの品質保證 ISO 9000-3 對譯と解說, 日本規格協會, 1992.

[14] 堀田 外, "ソフトウエアのプロセスメント手法の評價", 情報處理學會研究會報告, Vol. 92, No. 88-4, 1992. 11.

[15] 山本里枝子 外, ソフトウエアプロセスに基づくソフトウエア開發環境の檢討, 情報處理學會 'ソフトウエアプロセスシンポジウム' 論文集, pp. 1-10, 1994.

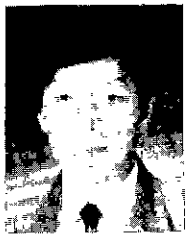
[16] 양해술, 이용근, 허태경, "소프트웨어 프로젝트 관리에서의 품질보증 시스템의 프로세스 기술방식", 한국정보처리학회, 정보처리 Vol. 1, No. 3, 1994. 9.

[17] 양해술, 김명옥, 박정호, "분산 개발환경의 현상과 전망", 한국정보처리학회, 정보처리 Vol. 2, No. 1, 1995. 3.

[18] 양해술, 이용근, 황인수, "소프트웨어 프로세스 리엔지니어링(SPR)의 개요와 접근 방법", 한국정보처리학회, 정보처리 Vol. 2, No. 3,

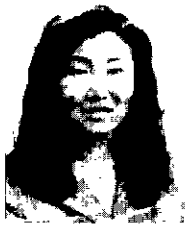
1995. 8.

양 해 술



1975 홍익대학교 공과대학 전기공학과 졸업(학사)
1978 성균관대학교 정보처리학과 정보처리 진공(석사)
1991 日本 오사카대학 기초공학부 정보공학과 소프트웨어공학 전공(공학박사)
1975~1979 육군중앙경리단 전자계산실 시스템 분석장교 근무
1986~1987 日本 오사카대학 객원연구원
1993~1994 한국정보과학회 학회지 편집부위원장
1980~1995 강원대학교 전자계산학과 교수
1994~1995 한국정보처리학회 논문지 편집위원장
1994~현재 한국산업표준원(IIS) 이사
1995~현재 한국소프트웨어품질연구소 소장
관심분야 : 소프트웨어 공학(특히, S/W 품질보증과 평가·SA/SD·OOA/OOD/OOP·CASE·SI), 소프트웨어 프로젝트관리

이 용 근



1988 강원대학교 자연과학대학 전자계산학과 졸업(이학사)
1994 강원대학교 전자계산학과 소프트웨어공학 전공(이학석사)
1989~1992 강원대학교 전자계산학과 조교
1994~1995 한림전문대학 전산정보처리학과 강사
1995. 3~현재 강원대학교 전자계산학과 박사과정
관심분야 : 소프트웨어공학(특히, 소프트웨어 품질보증과 품질평가·객체지향 프로그래밍·객체지향 분석과 설계 방법)

이 하 용

1993 강원대학교 자연과학대학 전자계산학과 졸업(이학사)
1995 강원대학교 전자계산학과 소프트웨어공학 진공(이학석사)
1995. 3~현재 강원대학교 전자계산학과 박사과정
관심분야 : 소프트웨어공학(특히, 소프트웨어 품질보증과 품질평가·객체지향 프로그래밍·객체지향 분석과 설계 방법)