소프트웨어 프로세스 개선과 메트릭 기법†

동국대학교 최은만*

1. 서 론

소프트웨어 프로세스와 메트릭 기법은 관련이 깊은 주제이다. 최근 각 기업마다 소프트웨어 프로세스를 평가받고 개선하려는 노력에는 여러가지 메트릭 기법과 관련된 질문들이 포함되어 있다. 우리 프로젝트에서는 과연 원하는 결과를 성취하였는가? 우리 팀의 생산성은 얼마나 되며 일정예측의 정확도는 어떤가? 개발한 소프트웨어의 결함률은 얼마이며 어떻게 품질을 향상시킬 수 있는가? 과연 지금의 개발 프로세스는 효과적인가? 등의 질문에 대한 응답이 필요하다.

최근 소프트웨어 업계의 화두가 되고 있는 소프트웨어 프로세스의 성숙도[1]는 여러가지 관리 기법과 프로세스의 정의, 주요 프로세스 영역 활동에 초점이 맞추어져 있지만 프로세스 개선이라는 측면에서 보면 메트릭 기법이 바탕을 이루고 있다. 특히 어느 정도 프로세스가 정의되고 관리가 이루어지는 성숙도를 보이는 기관에서는 더 높은 수준으로 나가기 위하여 메트릭에 의한 측정이 필수적이다. 측정에 의하여 프로세스와 프로덕트의 품질을 가시화할 수 있고 제어할 수 있기 때문이다.

소프트웨어 프로세스 개선이라는 주제를 메트릭이라는 관점으로 다루어 본다. CMM(Capability Maturity Model)에서 다루고 있는 메트릭은 어떤 것이며 과연 프로세스 개선 차원에 어떤 메트릭 기법을 도입할 수 있는지 설명한다. 또한 프로젝트 관리 측면에서 메트릭의 계획과 측정, 결과의 분석 과정을 다룬다.

† 본 연구는 정보통신부와 한국소프트웨어공학협회의 '한·카네기멜론 S/W 전문인력 교육 국내보급사업'의 지원으로 수행되었음

* 종신회원

2. CMM과 측정

CMM의 주요 주제는 지속적인 프로세스 개선이다. 소프트웨어 품질은 프로세스의 품질에 의하여 좌우되며 프로세스를 잘 정의하고 조직내의 문화로 정착되어야 한다는 것이 강조되어 있다. 또한 시니어 관리자의 프로세스 추진력, 개발 여건의 변화에도 프로세스가 무시되거나 지연되지 않는 안정성, 프로세스를 콘트롤할 수 있고 계속적으로 개선되어야 한다는 것이 CMM의 핵심이다[2].

CMM의 KPA(Key Process Area)를 해설한 [1]을 살펴보면 측정 메트릭은 여러가지 활동을 통합하는 역할을 하고 있다. 예를 들어 레벨 2(Repeatable)의 소프트웨어 프로젝트 계획 활동을 살펴보면 15가지 제시된 액티비티의 현황은 측정과 분석을 통하여 성패를 결정할 수 있는데 달성된 작업의 수, 지연된 작업, 추가 소요 예산 등을 추정된 값과 비교하여 판단할 수 있다.

소프트웨어 프로젝트에서 소요되는 기간과 비용, 소프트웨어의 규모와 결함 등의 예측값과 수행 후의 결과값이 일치할 확률을 나타낸 그래프는[3] 프로세스 성숙도 단계의 특징을 확연히 드러낸다. 개발 경험이 거의 없고 프로세스가 정의되지 않은 미숙한 단

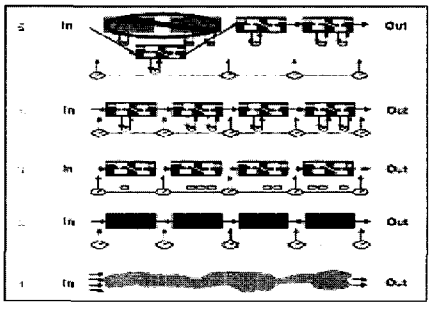

그림 1 CMM 단계별 프로세스 가시성[2]

계는 예측이 부정확하여 기간, 비용, 결함이 예측값을 초과하는 경우가 많다. 프로세스가 정의되어 어느 정도 관리가 이루어지는 단계에도 기간, 비용, 결함 등이 감소되지만 적중률이 떨어져 확률분포가 커진다. 그러나 높은 수준의 성숙도를 보이는 경우는 목표값에 대하여 정해진 범위 안의 오차로 적중할 확률이 높아지며 계속적인 개선에 의하여 최적화해 나갈 수 있다.

CMM에서 소프트웨어 메트릭이 담당하는 중요한 역할 중의 하나는 프로세스가 점점 가시화 된다는 점이다. 낮은 성숙도를 보이는 수준에서는 결과가 나올 때까지 작업 과정에서는 알 수가 없다. 그러나 그림 1에서 볼 수 있듯이 수준이 높아질수록 메트릭에 의하여 프로세스의 진행 상황을 드러낼 수 있고 완급과 정도를 조절할 수 있다. 레벨 5에서는 측정결과를 기초로 프로세스를 변경하여 개선해 나가는 높은 성숙도를 보인다.

메트릭이란 관점에서 CMM의 각 단계를 더 자세히 살펴보면 그림 2와 같은 특징을 보인다. 레벨 2는 주로 개별 프로젝트의 소프트웨어 규모, 비용, 기간, 요구 등에 대하여 계획대비 실행 결과값을 측정한다. 레벨 3(Defined)은 프로젝트 차원이 아니라 기관에서 통일된 메트릭으로 프로덕트와 작업결과에 대하여 측정하는 성숙도를 보인다. 레벨 4(Managed)는 프로세스별 작업효율을 측정할 수 있으며 레벨 5(Optimized)는 이를 이용하여 프로세스를 개선하는 능력을 보인다.

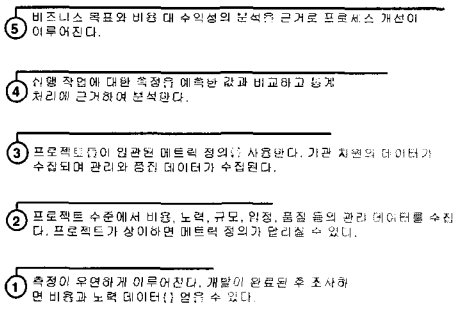

그림 2 단계별 측정활동의 특성

3. 프로세스 개선과 메트릭 기법

메트릭 기법은 일반적으로 측정 대상을 기초로 그루핑할 수 있다. 개발 프로세스의 기간, 비용, 효율을 측정하는 것과 개발한 소프트웨어, 즉 프로덕트 자체의 특성을 측정하는 메트릭이 있으며 소프트웨어 개발 인력, 팀, 동원되는 소프트웨어 및 하드웨어 등의 자원을 측정하는 메트릭으로 나누어 볼 수 있다. 이들 메트릭 중 하나의 측정값으로 표현할 수 있는 것이 직접 메트릭이며 여러개의 측정값의 비율 또는 함수로 표현되는 것이 간접 메트릭이다.

표 1 메트릭의 구분

구분	측정 대상	타입	메트릭의 예
프로덕트	원시코드	직접	크기
		간접	신뢰도
프로세스	테스트	직접	노력
		간접	비용
자 원	인력	직접	경력
		간접	생산성

메트릭 기법을 개발 조직의 어느 범위에 적용하느냐에 따라 CMM의 성숙도가 달라진다. 개발 또는 유지 보수하는 단위 프로젝트의 비용, 일정, 품질, 기능 등을 측정하여 프로젝트를 관리하려는 수준은 CMM 레벨 2에 해당된다. 레벨 3 이상에서는 메트릭이 프로세스 관리 차원에서 도입되어야 한다. 기관에서 수행되는 여러 프로젝트에 공통 메트릭을 적용하여 정의된 프로세스가 잘 수행되었고 과연 효율적인지 판단하고 개선하여야 한다. 즉 메트릭이 프로세스의 개선에 중요한 도구가 되는가가 CMM의 높은 성숙단계에 오르는 기준이 될 수 있다.

메트릭 기법을 프로세스 개선 차원으로 생각한다면 그림 3에 표현한 단계로 적용되어야 한다. 설계, 코딩, 테스트 등의 프로세스가 개발 프로젝트에서 수행되었다면 각 프로세스가 얼마나 잘 이루어졌는지 측정하기 위하여 예상 목표와 결과를 비교하여야 한다. 또한 이런 편차를 분석하고 해석하여 무엇을 개

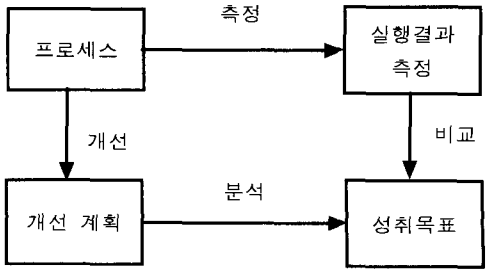

그림 3 측정을 통한 프로세스 개선

선할 것인지 파악한다. 계획한 개선을 실행한 후에는 다음 사이클에서 더 좋은 결과를 얻게 되었는지 확인한다.

위와 같은 측정 작업의 결과는 단일 프로젝트 차원만이 아니라 다양한 조직단위로 취합되어야 한다. 기업 수준의 프로세스를 담당하는 SEPG(Software Engineering Process Group)와 부서별, 또는 비즈니스별 집계가 이루어지고 결과를 분석하여 프로세스의 정의와 개선에 반영되어야 한다.

그림 3과 같은 메트릭 적용 프레임워크가 형성되었다면 프로세스 정의와 측정 작업 사이에는 깊은 관련을 맺게 된다. 프로세스 정의에서 작업 효율성, 즉 실행 목표를 설정하고 실행 결과가 목표에 도달하였는지 모니터링하여야 한다. 이런 성숙된 작업이 가능하려면 메트릭에는 항상 예측값이 중요하다.

표 2 CMM 레벨 2 메트릭[4]

KPA	번호	메트릭	범 주
프로젝트 계획	1	작성된 KLOC(Kilo Lines of Code), 재사용된 KLOC, 총 KLOC(또는 Function Point)	크기
	2	LM(Laber Month), LH(Laber Hour), 비용	비용
	3	경과된 달 수	일정
	4	제품 완결성(전체 소프트웨어 완결 비율(KLOC, FP 당)	획득 가치
품질보증	1	단계별 개발 프로세스를 위한 인스펙션 회수	미적용
형상관리	1	ECP(Engineering Change Proposal)의 수	안정성
	2	변경된 요구사항 개수/전체 요구사항 개수 * 100	
프로젝트 추적	1	작성된 KLOC(Kilo Lines of Code), 재사용된 KLOC, 총 KLOC(또는 Function Point),	크기
	2	LM(Laber Month), LH(Laber Hour), 비용	비용
	3	경과된 달 수(경과된 달 수/소요 개월) * 100	일정
	(중략)		

[4]에는 소프트웨어 프로젝트의 상태를 추적하고 모니터링하는데 사용될 수 있는 38 개의 메트릭이 제시되어 있다. 38 개의 메트릭은 8 개의 범주, 소프트웨어 제품 크기, 비용, 개발 일정, 프로젝트 기술 안정성, 프로젝트 상태, 품질, 획득 가치(Earned Value), 컴퓨터 자원으로 구분할 수 있다. 흥미로운 것은 이 38 개의 메트릭을 CMM의 KPA별로 연결시켰다는 점이다. 즉 메트릭을 성숙도 단계별로 나누고 각 KPA별로 사용할 만한 메트릭을 제안하였다. 예를 들어 CMM 레벨 2에서는 표 2와 같은 메트릭들을 제안하고 있다.

여기서 주의하여야 할 점은 성숙도 단계별로 사용되는 메트릭 기법의 종류가 달라지는 것이 아니라 예상되는 값과 실제값을 비교하고 있는지, 측정하는 메트릭값이 프로젝트 단위로 이루어지는지 아니면 프로세스 단위로 이루어지며 전사적으로 취합되고 있는지가 중요하다. 레벨 1에서도 [4]에서 제시된 레벨 3의 메트릭을 적용할 수 있다. 다만 레벨 1에서는 이미 완결된 프로젝트에 대하여 측정이 이루어지므로 예상값이 존재하지 않는다. 레벨 2 이상의 성숙도를 보이는 경우 프로세스를 추적하기 위하여 모든 메트릭에 예상값, 즉 목표치를 정하고 실향 결과값을 대비하여 분석한다.

표 3 CMM 레벨 2 메트릭 기법

범 주	메트릭	레벨 2 KPA	예측값	실행값
진척도	계획 및 실행 기간(간트 차트)	PP, TO, SM		
노력	계획 및 실행 투입 인력	PP, TO, SM		
비용	계획 및 집행 비용 비용과 일정의 변동	PP, TO, SM		
소프트웨어 품질 보증	부적합 개수	SQA		
검토 결과	작업 아이템의 상태	TO, SQA		
문제점 보고	문제점 보고에 대한 상태 보고기간에 해결된, 해결되지 않은 문제의 개수 문제점 보고 밀도 문제점 보고와 통과된 테스트 케이스 개수의 비교	RM, TO, SCM		
요구 안정성	변경된 요구 및 확정된 요구의 개수 요구 변경의 릴리스별 분포	PP, TO, SM, SCM		
크기 안정성	크기 변동 크기의 릴이스별 분포	SCM		
컴퓨터 자원의 이용	예측 및 실행된 컴퓨터 자원의 이용률	PP, TO, SM		

이러한 의미에서 [4]에 소개된 CMM 각 단계별 메트릭은 표 3과 같이 개선되어야 한다. PP(Project Planning), TO(Project Tracking and Oversight), SM(Subcontract Maintenance), SQA(Software Quality Assurance), SCM(Software Configuration Management), RM(Requirement Management) 각 KPA별로 중복 적용될 수 있는 메트릭이 많지만 레

벨 2에서는 규모, 노력, 일정, 자원, 요구 변경률, 형상 관리 및 품질 보증 관련 측정이 중요하다.

레벨 3은 메트릭 종류도 추가되지만 주목할 점은 메트릭 측정이 전사적으로 이루어지며 그 사용 방법이 달라진다는 것이다. 즉 과거의 데이터에 근거하여 예측이 이루어지며 각 프로젝트의 메트릭이 표준화되어 있고 여러 프로젝트의 데이터가 취합되어 분석된다. 즉 표 4에서와 같이 각 프로젝트 또는 프로덕트에 대한 메트릭 측정값들을 취합하여 전사적인 평균 또는 생산성 통계값이 추출되어야 한다.

레벨 4 이상의 성숙도를 보이는 기관의 메트릭 측정은 여기에 적합도를 판정하는 기준이 추가된다. 즉 각 메트릭값들에 대한 가이드라인, 즉 허용범위가 설정되어 궤도 수정 계획을 수립할 수 있다. 예측의 정확도가 높아지고 프로세스 수행결과, 생산성, 능력을 객관적으로 평가할 수 있는 메트릭 인프라를 가지고 있는 것이다.

레벨 5의 측정 메트릭은 프로세스 개선을 고려한 실험에 초점이 맞추어져 있다. 새 프로세스에 대한 가설을 세우고 이를 시뮬레이션하거나 실험하여 변화시킨 인자가 얼마나 영향을 주는지 알아내는 작업이 필요하다. 이러한 실험을 바탕으로 프로세스 개선에 의한 효과를 예측할 수 있다.

4. 측정 과정 및 문제점

메트릭 측정 과정과 기법은 [5]에 잘 정리되어 있다. 우선 [6]에 제안된 GQM(Goal-Question-Metric) 방법에 의하여 측정 목표를 세우고 그 목표와 관련된 질의를 정리하고 이를 만족시킬 만한 메트릭을 선택한다. 여기서 중요한 것은 메트릭 표시값, 즉 Indicator를 정확히 정의하고 이 값이 어떤 스케일에 속하는지 이해하는 것이다. 스케일에 따라 통계적 처리가 의미가 있는지 없는지 판단할 수 있다.

메트릭을 선택한 후에는 측정 과정과 기록 방법을 정의하여야 한다. 데이터를 취합하는 양식을 만들고 어떤 절차로 데이터 값을 찾아내며 사용할 수 있는 도구는 무엇인지 찾아낸다.

메트릭에 의한 측정이 이루어진 후에는 분석이 이루어져야 한다. 스케일에 따라 분석 방법이 달라지지만 중요한 것은 프로세스 개선을 위한 의사결정에 연결되어야 한다는 점이다. 분석된 결과를 보고하고 배포하는 범위 시간, 형식 또한 정해져야 한다.

측정에 대하여 항상 제기되는 문제점은 과거의 데이터가 데이터베이스로 잘 보관되어야 하며 측정 데이터를 쓸모있는 정보로 만들기 위하여 분석 방법을 개발하여야 한다는 점이다. 또한 분석 정보가 적시에 의사결정에 반영되어야 한다.

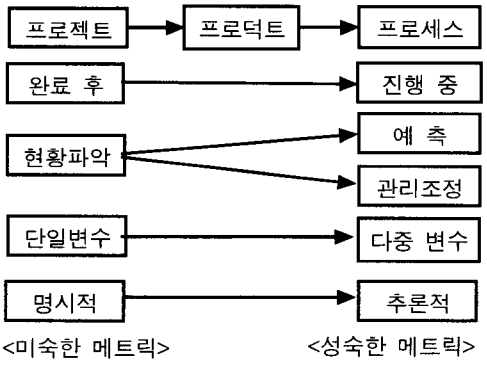

그림 4 메트릭의 개선

표 4 레벨 3의 메트릭 측정 예(Peer Review)

프로덕트 ID	Peer Review회수	발견한 결함수		평균규모 (페이지, KLOC)	총크기	리뷰당 검토인원	리뷰 시간		생산성		
		중요 결함	총계				리뷰준비 평균시간	리뷰관리 평균시간	시간 당 중요 결함발견	시간 당 결함발견	크기별 중요결함
RAD	1	0	28	13.00	13	6.0	0.67	0.67	0.00	3.11	0.00
SRD	7	0	73	6.86	48	4.4	0.68	0.87	0.00	1.48	0.00
PS	8	3	76	3.50	28	4.5	0.33	0.67	0.08	1.96	0.11
DA	13	2	188	13.23	172	4.5	0.57	1.20	0.02	1.91	0.01
PS&DA	1	0	11	12.00	12	5.0	0.23	1.00	0.00	1.65	0.00
FRS	1	0	15	6.00	6	6.0	0.62	0.75	0.00	1.76	0.00
SDD	6	0	215	23.50	141	5.3	1.00	2.27	0.00	1.72	0.00
Code	16	11	393	1.45	23.2	3.7	0.95	1.48	0.07	2.46	0.47
Proc	6	24	129	14.67	88	4.5	0.39	1.24	0.50	2.71	0.27
SCM	0	0	0	0	0	0.0	0.00	0.00	0.00	0.00	0.00
총계	59	40	1128			4.4	0.54	1.03	0.07	1.88	

5. 결 론

정확한 측정은 프로덕트 품질과 프로세스 개선에 중요한 요소이다. CMM을 모델로 하여 프로세스를 개선하려면 우선 여러 KPA에 걸쳐 퍼져있는 메트릭 기법들을 인지하고 이를 잘 정의하는 작업이 필요하다. 측정 메트릭 기법이 다양하고 복잡하지만 각 KPA에서 요구하는 작업의 특성을 잘 나타내는 메트릭 기법을 찾아내고 이를 프로젝트 차원뿐 아니라 기관 차원의 관리에 적용하려는 노력이 필요하다.

더 높은 수준의 프로세스로 평가받기 위하여 메트릭 측정에도 개선 계획이 필요하다. 확장된 KPA별로 측정 메트릭을 추가하여 단순히 적용하는 것으로는 부족하며 예측과 결과값의 비교, 기관 차원의 데이터 수집과 이를 분석하여 그림 4와 같은 메트릭 자체의 개선과 프로세스 개선에 연결되도록 하는 노력이 필요하다.

참고문헌

[1] M. Paulk et al. The *Capability Maturity Model: Guidelines dor Improving the Software Process,* Addison-Wesley, 1995.

[2] M. Paulk, SEEK 2003 *Lecture Note for Introduction to CMM,* ISRI, CMU, 2003.

[3] D. Perry, G. Votta, "People, Organizations, and Process Improvement," *IEEE Software,* 11(4), pp.36-45, 1994.

[4] J. Gaffney et al. *Software Measurement Guidebook,* International Thomson Computer Press. 1995.

[5] N. Fenton, S. Pfleeger, Software Metrics: A Rigorous and Practical Approach, International Thomson Computer Press, 1997.

[6] V. Basili, D. Weiss, "A Methodology for Collecting Valid Software Engineering Data," IEEE Trans. on Software Engineering, SE-10(6), pp. 728-738, 1984.

[7] A. Flora, R. Park, A. Carleton, *Practical Software Measurement: Measuring for Process management and Improvement,* Guidebook CMU/SEI-97-HB-003, April 1997.

[8] 안유환, 김길조, 오세원, 김종윤, 소프트웨어 프로세스의 계량적 관리, 정보과학회지, 17(1), pp. 35-43, 1999.

[9] 최은만, 소프트웨어 품질측정 및 데이터분석 기술연구, 한국전자통신연구원 위탁연구과제 최종보고서, 2001.

최 은 만

1982 동국대학교 전자계산학과 졸업 (학사)
1985 한국과학기술원 전자계산학과 졸업 (석사)
1993 Illinois University 전산학과 졸업 (박사)
1985 한국표준연구원 정보처리표준화 담당
1988 (주)데이콤 행정전산과 담당
1993~현재 동국대학교 컴퓨터멀티미디어공학과 부교수

관심분야 : 소프트웨어 메트릭, 소프으웨어 테스트, 소프트웨어 설계, 유지보수 및 프로그램 이해
E mail : emchoi@dgu.ac.kr