온라인 인터넷 서비스 환경에서 테스트 조직 개선을 위한 메트릭 개발

권효진°, 이진호, 최진영
고려대학교 소프트웨어공학과, 고려대학교 컴퓨터학과
e-mail : khjin9710@gmail.com° , {jhlee, choi}@formal.korea.ac.kr

A case study: Metrics for the improvement of the test organization in on-line web service

Hyo-Jin Kwon°, Jin-Ho Lee, Jin-Young Choi
Dept. of Software Engineering, Dept. of Computer Science, Korea University

요 약

소프트웨어 제품의 품질 확보 및 비용 감소를 위해서 테스트 활동이 중요시되고 있으며 품질을 중시하는 조직에서는 개발 프로세스 내의 독립된 조직에 의해 생산물에 대한 검토 활동이 수행되고 있다. 테스트 조직이 효과적·효율적으로 테스트 활동을 수행하기 위해서는 테스트 활동에 대한 측정 및 평가를 위한 메트릭을 필요로 한다. 하지만 현재 사용하고 있는 메트릭만으로는 테스트 조직의 활동을 평가하고 향상시키기에는 부족하다. 따라서 테스트 조직에 필요한 메트릭 연구가 필요 하며 이 메트릭을 통해 테스트 활동에 필요한 지표를 도출할 수 있다.

본 연구에서는 목표-질문-메트릭(GQM: Goal Question Metric)방법을 이용하여 온라인 인터넷 서비스 환경의 테스트 조직에서 사용하고 있는 메트릭을 검토하고 테스트 조직 향상에 필요한 메트릭을 도출한 사례를 제시한다.

1. 서 론

온라인 인터넷 서비스 환경에서 개발되는 소프트웨어 제품은 실시간으로 운영되고 있으므로 패키지 제품에 비해서 릴리즈 후 수정이 용이하다는 점이 있으나 이 역시 소프트웨어의 오류는 서비스사의 이미지 손실뿐만 아니라 금전상의 막대한 손실을 가지고 온다. 최근 온라인 인터넷 서비스 환경에서 제품을 개발하는 업체에서도 시스템 운영의 안정성 및 추가, 변경되는 기능의 품질 확보를 위한 검증 활동인 테스트가 수행되고 있다. 검증의 객관성 확보를 위해 제 3의 전문가에 의한 테스팅이 요구되며, 테스트 활동이 효과적·효율적으로 수행되기 위해 독립적인 테스트 조직을 구성하여 운영하고 있다.

시장 적시 출시(Time to Market)가 강조되는 온라인 인터넷 서비스 환경의 제품은 요구사항을 신속히 반영하여 제품을 출시해야 하기 때문에 소규모의 다수 프로젝트가 수행된다. 프로젝트 규모가 작은 만큼 테스트 조직에서 테스트를 수행하는 시간이 패키지 또는 여타의 제품에 비해서 상대적으로 짧은 시간에 이루어지게 된다. 테스트 프로세스도 패키지 개발의 테스트 프로세스에 비해 절차가 간소화되고 서비스 환경의 특성이 반영된 형태로 변화된다. 테스트 조직에서는 변화된 테스트 프로세스에서 테스트 활동이 올바르게 수행되고 있는지 평가하고 개선하기 위한 활동 측정지표를 필요로 한다.

온라인 인터넷 서비스 환경에서 독립된 테스트 조직으로 운영되는 A사는 테스트 활동을 측정하고 평가하기 위한 메트릭을 가지고 있다. 이 메트릭들은 결함의 정보를 바탕으로만 구성되었으므로 테스트 조직이 추구하고자 하는 목표와 연계를 맺지 못하고 품질을 평가하는 기준이 명확하지 못하다.

테스트 조직의 활동 측정지표를 도출하기에 앞서 테스트 조직 향상을 위한 방향 제시로 테스트 성숙도 평가 모델인 Test Maturity Model(TMM)을 적용하여 테스트 조직을 평가하였다. 테스트 조직의 목표에 맞는 메트릭을 도출하기 위해서 Goal Question Metric(GQM) 방법을 이용하여 테스트 활동에 필요한 메트릭을 도출하고 검증하였다. 도출된 메트릭을 통해 A사는 테스트 조직의 향상을 이룰 수 있을 것이다.

본 연구의 구성은 다음과 같다. 2장에서 테스트 성숙도 평가 모델인 TMM과 목표 지향 측정 방법인 GQM에 대해서 간략히 소개한다. 3장에서는 온라인 인터넷 서비스 A사에 GQM을 이용해서 개발된 메트릭 사례를 소개한다. 4장에서는 결론 및 향후 연구사항을 기술하도록 한다.

2. 관련연구

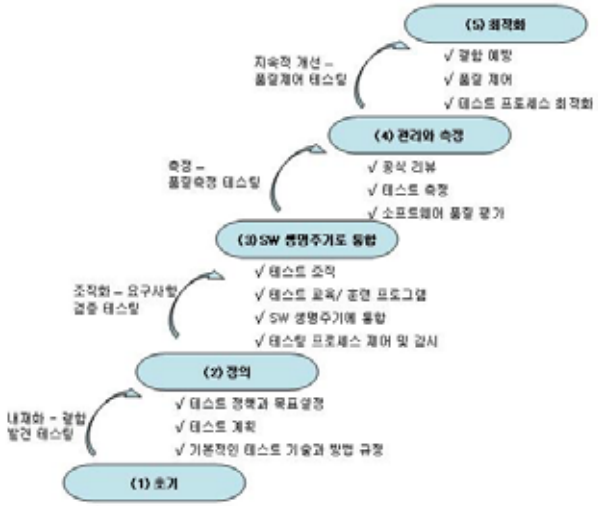
2.1 TMM (Test Maturity Model)

TMM 은 1996 년 일리노이 공대의 Burnstein 교수에 의해 개발된 것으로 테스트 관리에서 제안된 이슈의 중요성, 성숙도 각각의 레벨에서 권고된 활동, 조직의 테스트 프로세스를 평가하고 향상시키기 위한 평가 모델이다. TMM 모델은 CMM 에서 테스트 영역의 부족함을 보완하기 위해 만들어졌고 5 레벨의 단계적 프레임워크

로 이루어져 있다[1][2].

- Level1 (인식): 테스팅의 필요성 인식. 테스트 프로세스 미 정립
- Level2 (이해): 개별 프로젝트에서 테스트 프로세스가 사용됨
- Level3 (정의): 조직의 프로세스가 정의됨
- Level4 (정량화): 다양한 매트릭스에 의해 테스트 프로세스가 정량적으로 평가됨
- Level5 (최적화): 테스트 프로세스 개선, 결함 예방 중점

각 단계의 핵심 요소와 세부 목표들을 달성하기 위한 주요 활동은 그림 1 과 같이 나타낼 수 있다.



(그림 1) 테스트 성숙도 모델 레벨

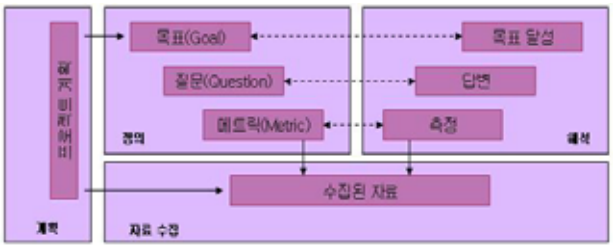
2.2 GQM (Goal Question Metric)

GQM 은 비즈니스 목표를 식별하여 이를 나눔으로써 시작하며, 잘 정의된 메트릭과 목표를 지원하는 지표에 대한 계획을 세움으로써 종료된다. GQM 접근방법은 3 단계로 정의되며 다음과 같다. 첫 번째 단계는 개념적인 단계이다. 이 단계의 요소로는 Object, Purpose, View Point, Focus 등이 포함되고 목표를 정의한다. 두 번째 단계는 운용단계로서 설정된 목표에 각각의 관점에서 합당한 질문을 하는 단계다. 세 번째 단계는 Metrics 로 답하는 단계이다[3].

품질 향상을 위한 GQM 방법을 내포한 목표 지향 측정은 계획, 정의, 데이터 수집, 해석 단계로 구성된다.

- 계획 단계: 측정을 위한 대상이 선정되고 계획됨
- 정의 단계: 측정을 위한 대상에 대해 GQM 접근
- 데이터 수집 단계: 데이터 수집을 통한 결과 획득
- 해석 단계: 메트릭과 데이터 수집에서 얻어진 결과를 통해 목표 달성을 위한 해결책 제시

이를 도식화하면 그림 2 와 같다[4].



(그림 2) GQM 방법의 4 단계

3. 테스트 조직 향상 (Test Organization Improvement)

온라인 인터넷 서비스 환경에서 소프트웨어 제품을 개발 및 유지보수 하는 영역은 요구사항을 신속히 반영하기 위해 개발 프로젝트 단위를 작게 해서 짧은 시간 내에 개발과 테스트를 수행하여 릴리즈 하고 있다. 수익과 관련된 심각한 결함이 아닌 경우, 기존에 수립된 베이스라인의 릴리즈 일자에 맞게 사이트에 반영된 후에 결함이 수정되는 경우도 적지 않다. 서비스 환경의 특성에 맞춰 테스트 프로세스는 경량화 되어 수행되고 있다.

3.1 TMM 에 기반을 둔 테스트 조직 평가

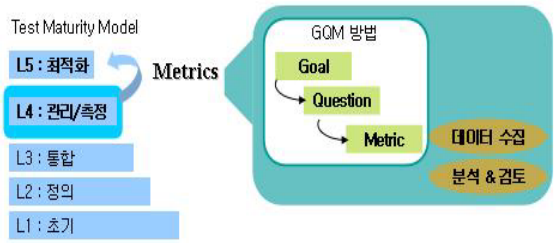
설문 및 인터뷰를 통하여 테스트 조직 내의 프로세스 및 활동을 조사하였다. 조사한 결과를 TMM 에서 정의하는 활동들과 매칭해 본 결과 레벨 3 에 준수하는 조직으로 판단된다. 테스트 조직의 향상을 위해서는 테스트 측정(레벨 4)이 올바르게 되어야 하며, 이를 바탕으로 결함예방 및 품질제어 활동(레벨 5)이 이루어져야 테스트 프로세스가 최적화 될 수 있다.

<표 1> 서비스 환경 A 사의 테스트 조직 TMM 평가

	A 사 테스트 프로세스	TMM
1	테스트가 독립된 단계로 정의됨	Level 2
2	프로젝트 초기 단계에서 테스트 계획화됨	Level 2
3	테스트 기술 및 도구 지원됨	Level 2
4	요구사항부터 테스트 프로세스가 구성되어 수행 됨 (SW Lifecycle 통합)	Level 3
5	별도의 테스트 조직 존재	Level 3
6	테스트 교육 훈련이 전문적 활동으로 수행	Level 3
7	리뷰활동이 공식적으로 수행됨 *(리뷰의 결함이 관리 측정 되지 않음)*	Level 4 (부족)
8	테스트 활동이 정량화되고 측정됨 *(품질평가 및 메트릭 보완 필요)*	Level 4 (부족)

레벨 4 는 관리와 측정이 수행되어야 하는 단계로 테스트 활동이 측정되고 정량화 되어한다. 현 테스트 프로세스에서 사용하고 있는 메트릭이 테스트 조직에 적합여부와 결함 예방 활동을 지원하기 위한 추가적인 메트

릭 고안을 필요로 한다. 관리와 측정에 필요한 메트릭 개발의 접근법은 그림 3 과 같다.



(그림 3) 테스트 조직 향상을 위한 TMM 과 GQM 관계

3.2 GQM 을 이용한 테스트 메트릭 도출

테스트 조직에서 필요로 하는 데이터가 올바르게 측정되고 정량화 하기 위해서 GQM 방법을 이용하여 <표 2>와 같은 테스트 활동에 필요한 메트릭을 도출하였다. 테스트 조직 내에서 추구하고자 하는 목표를 정의하고 그 목표를 달성하기 위해 필요한 항목을 질문 한다. 그 질문에 대한 답으로 메트릭과 산정식을 구한다.

도출된 메트릭이 현 테스트 프로세스 내에서 측정되면 조직의 목표에 만족한 메트릭으로 인정하고, 측정되지 않는 메트릭이라면 새롭게 도출된 메트릭을 테스트 시스템 및 프로세스 내에 적용시켜 측정지표로 사용한다.

<표 2> GQM 을 이용한 Test Metric

Goal	Question	Metric
Product Quality (제품의 품질 확보)	Defect density (결함이 얼마나 밀집하고 수정되는가?)	제품 결함 밀도 = 결함수 / SIZE
		명세 결함 밀도 = 결함수 / Page
	Bug Type (어떤 유형의 버그가 발생하는가?)	버그 유형 분포
	Phase Intoruduced (버그가 어느 단계에서 많이 생산되는가?)	버그 유입 분포
	Bug Severity (얼마나 심각한 버그가 도출되는가?)	버그 심각도
Development Quality (개발 품질 향상)	Bug Severity (얼마나 심각한 버그가 도출되는가?)	버그 심각도
	Bug Fix (버그 수정에 들이는 시간이 얼마나 되는가?)	버그 수정 비용 = Min(1개 버그 수정 시간)
	Corrective Bug Fix (버그가 한번에 올바르게 수정되는가?)	버그 수정 빈도 = 재수정되는 결함 수
Test Egineer Ability (테스터 능력 평가)	Test Case Reuse (테스트 케이스 재사용이 얼마나 되는가?)	테스트 케이스 재 사용률
	Execution Results of Test Cases (테스트 케이스 적중률이 얼마나 되는가?)	테스트 적중률 (pass 비율, Fail비율, N/A 비율)
	Defect Quality (발견한 결함이 정말 결함인가?)	결함 품질 (dropped ones, defects counts)
	Test Estimation (테스트 scope산정이 얼마나 적중하는가?)	테스트 일정 산정 = (추정공정 - 실제공정)
Defect Cost (결함 비용 감소)	Test Execution Time (테스트 수행 시간이 얼마나 되는가?)	테스트 수행 비용 = Min(단계별 테스트 수행 시간)
	Bug Fix (버그 수정에 들이는 시간이 얼마나 되는가?)	버그 수정 비용 = Min(1개 버그 수정 시간)
Test Effectiveness (테스트 효과성 추구)	Test Effectiveness (테스트가 얼마나 효과적으로 이루어졌는가?)	테스트 효과성 = 테스트 기간 내의 결함 수 / (테스트 기간 내 결함 수 + 릴리즈 후 결함 수)

위의 도출된 12 개의 메트릭 중 8 개의 메트릭은 기존에 사용된 것으로 조직의 목표에 만족한 메트릭으로 검토되었다. 5 개의 메트릭은 새로 도출된 것으로, 프로세스 내에서 측정이 적합한지에 대한 검증이 필요하다.

3.3. 테스트 메트릭 검증

GQM 을 통해 도출된 메트릭은 데이터 수집을 통하여 테스트 조직 내에 필요한 데이터가 측정되고 적합한지를 검증한다. 검증 작업 후, 검증된 메트릭은 테스트 조직을 평가하는 지표로 사용할 수 있다. 본 연구에서는 GQM 방법을 이용해서 도출한 메트릭 요소 중 제품 결함밀도가 현 테스트 조직에서 사용하기에 적합한지를 검증하기 위해 데이터 수집 및 분석을 시도 하였다.

새롭게 도출된 메트릭(Defect Density, Corrective Bug Fix, Test Case Reuse, Test Estimation, Test Execution Time) 중 결함밀도는 품질을 평가하는 척도로 테스트 조직에서 품질 보증 및 향상을 위해 필요한 메트릭 이다. 결함밀도는 생산물에 얼마나 많은 결함이 존재하는지를 평가하기 위한 지표로, 개발된 생산물 내에서 발견되는 결함의 수에 따라 측정된다. 제품의 생산 단위(Size)는 결함 예방 및 테스트 활동을 추정하기 위한 기본 단위로 사용될 수 있는 만큼, 단위의 선정이 중요하다.

3.3.1 제품의 생산 단위(Size) 선정

A 개발사에서는 개발 생산품의 생산량 측정 단위로 널리 사용하는 LOC(Line of Code)를 사용하려 하였으나, 다음과 같은 이유로 LOC 에서 File 로 변경하였다.

첫 번째, LOC 방식은 4 세대 언어 특성 때문에 오브젝트 라이브러리, GUI 등을 사용하는 개발에서는 적합하지 않다. 두 번째, 프로젝트 내의 테스트가 평균 2 주 내에 수행되어 릴리즈되는 도메인 특성상 테스트 수행(Execution of Feature Test) 외의 부분(문서 및 관리 작업)에 0.5 일 이상 투여하기 어렵다. 툴에 대한 투자 비용도 쉽지 않아 적은 시간 내에 수작업으로 수행된 LOC 데이터 수집은 이루어 질 수 없었다. 20 개의 프로젝트에서 1 개의 프로젝트만이 LOC 데이터 수집이 가능하였다. 세 번째, 특정 기간 동안의 코드 변경 라인 산정 접근방식을 시도해보았지만, 산정 규모가 커지고 수행한 프로젝트와 생산된 코드의 상관관계를 구해야 하며 데이터 산정을 위한 별도의 작업 비용이 0.5~1 일 소요되는 단점이 발생하였다.

메트릭에서 사용하는 지표는 위의 3 가지 문제점을 해결할 수 있어야 하며 프로세스 내에서 자연스럽게 고비용을 들이지 않고 취득 할 수 있는 것이어야 한다. 연구 대상의 조직에서 수행하는 개발 프로세스를 살펴보면 다음과 같다. 무결성 확보를 위해서 제품의 소스에 대한 형상관리가 형상관리 툴(Rational ClearCase)을 사용하고 최종적인 소스 형상 검증은 독립적인 테스트 조직에 의해 수행된다. 제품 개발 프로세스 정의에서는 개발 기간에 작업한 산출물을 형상관리 툴에서 추출하여 테스트 환경에 적용 할 수 있도록 파일 리스트를 테스트 조직에 전달하도록 되어 있다. 개발 가이드 규칙상 새로운

기능은 새 파일(Stored Procedure, file)을 생성하도록 되어있다. 따라서 테스트 조직은 프로세스 내에서 파일 리스트를 받거나 형상관리 툴에서 해당 프로젝트 내에 개발된 산출물만을 검색하여 프로젝트 내의 추가되고 변경된 파일을 추출하여 개발 생산량의 데이터를 수집할 수 있다. 그 수집에 투여되는 시간도 평균 0.5 시간이였다. 이 File 을 이용한 코드 생산량의 측정법은 정밀한 FP(Function Point)방법은 아니지만, LOC 에 비하여 기능 단위의 사이즈로 측정 가능하다.

3.3.2 제품 결함밀도 측정

2007 년도에 5 개월 동안 정규 개발 프로세스에 의해 수행된 20 개의 프로젝트를 바탕으로 파일과 결함을 수집한 결과는 <표 3>과 같다.

<표 3> 제품 개발의 크기와 결함 관계

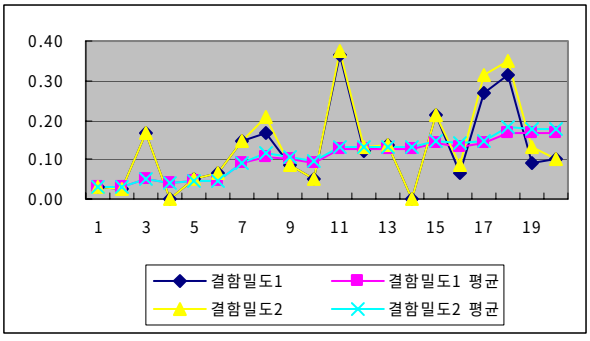
Projects	Files			Defects	Post Defects
	Total	New	Change		
1	31	0	31	1	0
2	37	4	33	1	0
3	12	0	12	2	0
4	17	4	13	0	0
5	19	3	16	1	0
6	31	5	26	2	0
7	108	39	69	16	0
8	72	14	58	12	3
9	127	17	110	11	0
10	122	86	36	6	0
11	88	41	47	32	1
12	105	73	32	13	1
13	52	15	37	7	0
14	5	4	1	0	0
15	154	20	134	33	0
16	91	4	87	6	2
17	48	43	5	13	2
18	219	199	20	69	8
19	45	26	19	4	2
20	20	1	19	2	0
Sum	**1403**	**598**	**805**	**231**	**19**

1) 결함 밀도 측정 Case 1

산정식: Product Density = count of defect / count of Files

- 결함밀도 1: 0.17
- 결함밀도 2 (릴리즈 후 결함 포함): 0.18

그림 4 는 20 개의 프로젝트에서 Case 1 의 산정식에 따른 결함밀도와 평균치를 보여준다.



(그림 4) 산정식 1 의 제품 결함밀도 분포

릴리즈 후 발생한 결함과 파일의 상관관계를 살펴보면, <표 4>와 같이 약 68.42%가 변경된 파일에서 릴리즈 후 결함이 발생하고 있다. 따라서 변경된 파일에 보정 값(0.5)을 추가하여 결함밀도를 구할 필요가 있다.

<표 4> 릴리즈 후 결함 수정 File 의 상관관계

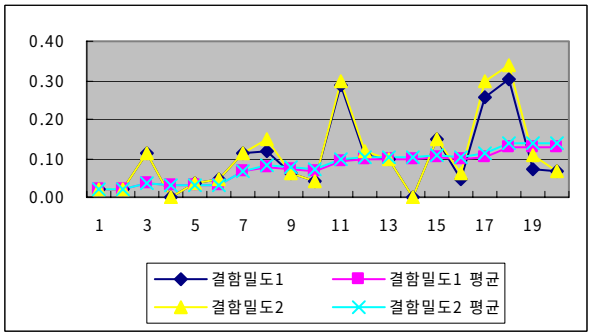
Projects	Post Defects	Files	
		New	Change
8	3	-	3
11	1	-	1
12	1	1	-
16	2	1	1
17	2	2	-
18	8	2	6
19	2	-	2
Sum	19	6	13

2) 결함밀도 측정 Case 2

산정식: Product Density = count of defect / count of Files (New + Change *a), a=1.5

- 결함밀도 1: 0.13
- 결함밀도 2 (릴리즈 후 결함 포함): 0.14

그림 5 는 20 개의 프로젝트에서 Case2 의 산정식에 따른 결함밀도와 평균치를 나타낸다.



(그림 5) 산정식 2 의 제품 결함밀도 분포

3.3.3 제품 결함밀도 측정 결과 분석

20 개의 프로젝트 내에서 제품 결함밀도 측정 결과 변경된 파일에서 결함이 약 2 배 많이 발생됨에 따라 변경된 파일에 가중치를 두어 제품의 결함밀도를 구하였다. 그 값은 0.14(case2 의 제품 결함밀도 2) 수치를 보였다. 이 측정된 결함밀도를 바탕으로 제품의 품질 목표의 결함 밀도 수치를 0.15 로 정의할 수 있다.

이 값이 품질 목표로 정의되면 프로젝트에서 개발 제품의 규모에 따라 발견해야 하는 결함 수를 예측할 수 있다. 예를 들어, 개발된 산출물이 테스트 조직에 인도될 때 추가된 파일이 10 이고 변경된 파일이 20 이면, 결함 수가 약 6 개 발생할 수 있다고 추정 가능하다. 더불어 테스트 기간 동안 6 개 이상의 결함을 발견해야 한다는 목표를 세울 수 있다.

결함밀도가 현 목표 값 보다 상회하는 수치를 보일

경우 버그 유형의 메트릭과 연계하여 테스트가 수행되기 이전의 리뷰 활동을 강화시켜 테스트 동안 발견하는 결함 수를 줄일 수 있다. 결함밀도가 현 목표 값을 계속 유지 한다면 목표 값을 내려서 테스트 케이스 작성 활동을 강화 시켜 테스트의 능력과 품질을 향상 시킬 수 있다.

4. 결론 및 향후 연구 방향

온라인 인터넷 서비스 환경의 테스트 조직을 향상시키기 위하여 테스트 활동을 측정하고 평가하는 메트릭에 대하여 연구하고 제시하였다. TMM 모델을 적용한 A 사의 테스트 조직에서는 레벨 4, 레벨 5 로 향상되기 위해서 테스트 활동에 대한 예측 및 결함예방 활동을 필요로 한다. 예측과 결함 예방 활동의 수행은 품질과 테스트 활동에 대한 정량적인 산정이 요구된다. 정량적인 산정을 위해서 GQM 방법을 이용하여 기존의 메트릭에 대한 검토와 조직의 목표를 달성하기 위한 새로운 메트릭을 도출하였다. 도출된 메트릭을 지표로 사용하기에 앞서 분석하고 활용에 대해서도 언급하였다. 도출된 메트릭을 기반으로 제품과 테스트 활동의 능력을 평가하고 결과 값을 개선 프로세스에 반영시킨다면 정량적인 데이터 기반으로 관리되는 테스트 조직으로 발전할 수 있을 것이다.

이 연구에서 도출된 메트릭은 특정 도메인에 국한되어 개발되었으므로 모든 테스트 조직에 적용 될 수 없는 한계가 있다. 하지만 이 연구의 접근방식을 TMM의 세부항목 평가를 위한 산정에 GQM 방법을 이용하여 표준화된 평가 메트릭을 도출하면 전문 테스트 조직을 평가하는데 유용하게 사용될 수 있을 것이다.

[1] Ilene Burnstein, Taratip Suwannasart, C.R. Carlson “Developing a Test Maturity Model: Part I”, Crosstalk, August 1996

[2] Ilene Burnstein, Taratip Suwannasart, C.R. Carlson “Developing a Test Maturity Model: Part II”, Crosstalk, August 1996

[3] VR Basili, G Caldiera, HD Rombach, “The Goal Question Metric Approach” - Encyclopedia of Software Engineering, 1994

[4] Rini Van Solingen, Egon Berghout, The Goal / Question / Metric Method, McGrawHill, 1999

[5] William Lewis, “Software Testing and Continuous Quality Improvement” 2 edition, 2005

[6] 양주미, 한혁수, “ CMMI 기반의 프로세스 및 제품품질보증 활동 평가를 위한 메트릭에 관한 연구”, 2006 한국 소프트웨어공학 학술지