컴포넌트의 품질 측정을 위한 메트릭에 대한 연구

장연세

수원과학대학 인터넷정보과 겸임교수 / 쿠도커뮤니케이션(주) 기술연구소장

A Metric for Evaluation of Component Quality

Jang, Yeun Sae

Adjunct Professor of Suwon Science College, Director of CUDO Tech. Institute

E-mail : tonyjang@ssc.ac.kr, tonyjang@cudo.co.kr

요 약

최근 국내 SI 사업체들은 소프트웨어의 가치를 향상시키기 위해 컴포넌트를 적극 도입하고 있다. 그러나 컴포넌트 시장을 활성화 시키기 위해서는 다양한 범주의 고객이 요구하는 양질의 컴포넌트를 풍부하게 갖추고, 고객이 시스템을 직접 개발하는 것보다 적은 비용과 시간을 소모하면서도 시스템을 구축할 수는 환경을 조성해야 한다. 그러한 환경을 구축하기 위한 요소중 고객의 구매 결정과 직결되는 가장 중요한 항목은 컴포넌트의 기술적 가치와 비즈니스적 가치의 평가와 이를 위한 시험 기준이다. 객관적이고 공정한 시험 기준이 마련되지 않는 상태에서 품질 평가가 이루어 지지 못한다면, 잠재적 고객이 구매 또는 사용하고자 하는 컴포넌트가 적절한 가치를 갖고 있는 것인지 판단할 수 있는 근거가 없고 시장 형성 초기부터 불량 컴포넌트 제품이 공급됨으로 인해 신뢰성이 저하되어 컴포넌트를 구매하는 대신 다른 대안을 선택하려 할 것이다. 객관적이고, 유력한 품질 평가 시스템을 구축하기 위해서는 품질 시험 평가를 위한 기준 마련이 선결 조건이다. 품질 시험 평가 가이드라인은 고품질의 소프트웨어 컴포넌트의 생산을 가능하게 하여, 궁극적으로 소프트웨어의 신뢰도를 향상 시키는 가장 유력한 방안이 될 것이다. 이를 통해 컴포넌트 소프트웨어의 유통 촉진 및 시장 성장을 견인, 생산/개발-유통-사용에 이르는 전체 컴포넌트 산업의 완결된 서비스를 제공 할 수 있을 것이다. 본 연구에서는 이러한 기준 마련을 위한 메트릭을 제공한다.

1. 서론

소프트웨어 개발 업계에서 프로젝트의 성공이란 수행 속도가 더 빠르고 성능이 뛰어나면서 안정성이 보장되며, 총 비용이 저렴한 소프트웨어를 만드는 것이다. 소프트웨어는 시장의 요구에 부응하여 현재의 요구 사항 혹은 장래의 요구 사항에 대한 서비스를 지원하면서도 수행 시 더 적은 오류를 포함하고 생산 및 운영 비용이 저렴해야 한다. 이러한 성공 요소들의 관점에서 컴포넌트 기반의 개발(Component Based Development : CBD)은 학계와 업계의 관심을 동시에 받고있다.[1][2]

본 연구는 한국소프트웨어컨소시엄의 연구 과제의 연구비 지원에 의하여 연구되었음.

컴포넌트에 기반한 소프트웨어 개발 기술은 소프트웨어 위기를 극복할만한 개발 생산성을 높일 수 있는 방법으로 주목받고 있다. 객체지향 패러다임을 기반으로 하는 컴포넌트의Plug-and-Play 기술의 발전으로 대규모 응용 소프트웨어를 쉽게 조립 완성할 수 있게 하는 기 술이 바로 컴포넌트 기술이다. 또한 인터넷으로 연결된 작업 환경은 컴포넌트를 획득하여 이를 쉽게 이용할 수 있게한다.[2]

컴포넌트를 이용하여 생산성을 증가시키려면 컴포넌트의 구성 자체의 기능과 호환성도 중요하지만 무엇보다 컴포넌트의 신뢰성, 즉 품질이 보장되어야 한다. 일반적으로 응용 소프트웨어는 완성된

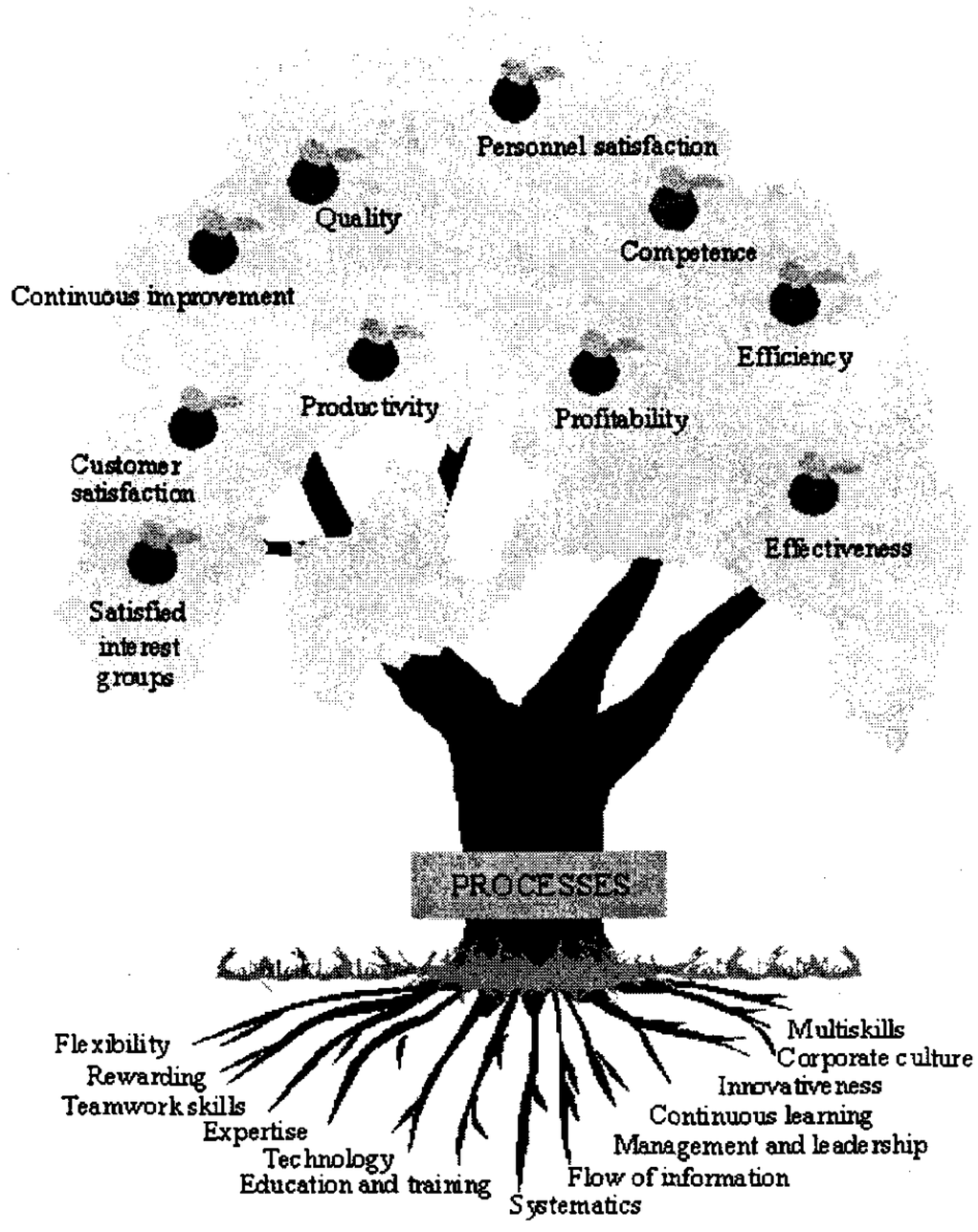
후 사용하기 위한 기능이나 성능, 유지 보수성등을 검사하고 평가한다. 그러나 컴포넌트는 제3자에 의하여 조립되어 새로운 소프트웨어를 만드는데 사용되므로 또 다른 차원의 품질 평가가 필요하다. 컴포넌트의 재사용 또는 조립에서 일어날 수 있는 오류를 철저히 예방할 수 있는 품질 평가 방법이 필요하다. 예를 들어 미국 NASA의 Ariane 5호 로켓의 폭발은 Value Range가 다른 Ariane 4호 로켓에 사용된 소프트웨어를 시험하지 않고 재사용했기 때문이었다. 이렇듯 컴포넌트 품질 평가는 개발하는 과정에서 주의를 기울여야 할 부분, 완성 전에 검토하여야 할 사항들과 완성 후 컴포넌트가 컴포넌트로서의 역할 즉 설계, 구현, 조립 등의 전 과정에 걸쳐 전반적인 품질 평가가 필요하다. 그러나 기존의 표준들은 ISO/IEC-12207과 같이 패키지 소프트웨어 평가에 대한 정의와 관점만이 투영된 상태이고 컴포넌트에 대한 평가 방법이나 지원 도구에 대한 연구는 아직 완료되지 않은 상태이다.[5]

2. 관련 연구

국내 주요 SI업체의 경우 ISO 9000에 근거한 품질 관리시스템은 어느 정도 갖추어져 있다고 볼 수 있다. 공정과 제품을 중심으로 하는, 품질구현 기법을 적용하는 방향으로 추진하는 전략은 기본이 되는 품질관리 시스템을 ISO 9000을중심으로 정비한 후, ISO12207을 적용하여 공정을 설정하고, CMM과 SPICE 등의 모형으로 프로세스를 개선해 나가야한다. 물론 이 과정 전반에 걸쳐서 ISO 9126을 기반으로 하는 제품 모형도 병행 적용해 나가야 한다.[4][7]

최근 논쟁이 되고 있는 컴포넌트 제품과 컴포넌트 제조 공정중 어떤 부문에 역점을 두어야 할 것인가에 대해서는 많은 학자들과 컨설턴트들이 제품 보다는 공정개선을 통한 품질증진에 무게를 두고 있다. 일반 제조업에서는 제품의 품질 관리에서 출발하여, 이로 인한 성과를 충분히 얻은 후에 한 차원 높은 품질구현을 위해서 공정모형과, 생산과 직접 관련이 없는 경영 전반에 걸친 활동(품질 경영)에 까지 그 적용영역을 확장해 왔다. 소프트웨어 산업의 경우는 공정의 안정 없이 뛰어난 1인의 기술자의 능력만으로도 품질이 높은 제품을 생산할 수 있지만, 문제는 이러한 성공의 반복을 보장할 수 없다는 것이다. 또한 현재의 기술을 비교해 보면 제품 보다는 공정중심의 모형이 보다 진보되어 있고, 실제 적용 경험도 많이 가지고 있다. 이러한 이유들로 인해 공정모형에 보다 많은 관심을 가지는 것이 바람직하다.[11][13]

다음의 [그림1]은 공정의 중요성을 잘 나타내 주고 있다. 즉, 품질, 이해관계자 그룹의 만족, 생산성, 이익추구, 효율성, 효과성, 능력 등의 열매를 얻기 위해서는 먼저 그 근간이 되는 유연성, 보상, 팀워크 기술, 전문지식, 기술, 교육훈련, 시스템화, 정보흐름, 경영과 리더십, 혁신, 기업문화, 복합기술 등의 요소들이 훌륭하게 뿌리를 내릴 수 있도록 하는 기반을 형성하는 노력이 선행 되어야 한다. 이렇게 해서 형성된 건실한 , 뿌리를 토대로 하여, 공정이라는 나무를 소프트웨어 공학기술과 품질 증진 프로그램들을 통하여 충실하게 육성하면 품질을 비롯한 각종 바람직한 결과물을 얻을 수 있을 것이다.[6][8]

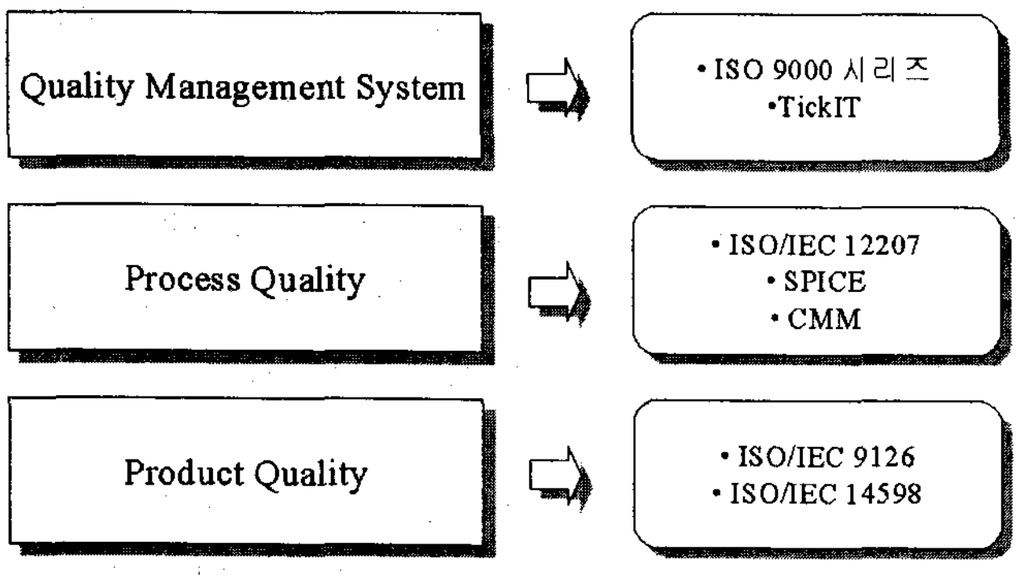


<그림 1> 프로세스 나무

2.1 소프트웨어 품질 관련 국제 규격

소프트웨어의 품질을 확보하기 위한 선진 각국들의 노력에 힘입어, 품질 증진을 위한 많은 이론과 구현 기법들이 개발되었고, 현재도 세계 각국에서 수많은 연구자와 실무자들이 많은 노력을 기울이고 있다. 이러한 노력들은 결국 국제 표준이라는 목표를 지향하여 자신들이 개발하여 사용하고 있는 기법을 세계의 모든 국가에서 표준으로 채택되기를 원하고 있다. 따라서 우리나라와 같이 소프트웨어 품질구현을 위한 기반환경이 조성되어 있지 않은 국가에서는, 품질을 확보하기 위한 구체적인 방법론으로 이들 국제 표준들을 이해하고 적용하는 것으로부터 출발하여야 한다.

현재 제정 되었거나 표준으로 제정되기 위한 작업이 가시적으로 진행된 위치에 있는 관련 표준들을 분류해 보면, 다음 [그림2]와 같이 품질관리 시스템(quality management system), 공정 품질(process quality), 제품 품질(product quality) 등의 세 분야로 나누어볼 수 있다.[3][7][8]



<그림 2> 소프트웨어 품질 국제 표준

3. 품질 측정 메트릭

품질 측정은 대부분 실험자의 주관과 실험 환경에 따라 다른 결과를 발생할 수 있다. 따라서 본 연구에서는 평가 결과의 객관성을 확보하기 위해 정상적 평가 항목을 정량적 평가 방법에 이입하였다.[9]

3.1 시험 평가 메트릭스

ISO/IEC-9126에 근간하여 품질 특성을 기본적으로 서로 다른 시각에서 6가지 특성으로 구분하였고, 각 특성은 3~5가지의 하부특성으로 상세화 한다. 이에 따라 시험 평가를 시행하면 산술적 평가 결과를 얻게된다 이를 정리하면 다음 [표 1]와 같다.

<표 1> 컴포넌트의 품질특성

품질 특성	하부 특성
기능성 (functionality)	적합성(suitability) 정확성(accuracy) 컴포넌트간 상호운용성 (interoperability) 표준 부합성(compliance) 보안성(security)
신뢰성 (reliability)	성숙성(maturity) 결함 허용성(fault tolerance) 복구성(recoverability)
사용성 (usability)	이해성(understandability) 학습성(learnability) 운용성(operability)
효율성 (efficiency)	자원활용도(resource utilization)
유지보수성 (maintainability)	분석성(analysability) 변경성(changeability) 안정성(stability)
이식성 (portability)	적응성(adaptability) 설치성(installability) 부합성(coformance) 치환성(replaceability)

1) 기능성(functionality): 컴포넌트가 명시된 조건하에서 사용될 때, 명시적 또는 묵시적으로 요구되는 기능을 제공해주는 컴포넌트의 기능을 의미한다. 이 기능성의 하부 특성들은 컴포넌트가 무엇(what)을 하는가를 나타내며, 기타의 나머지 특성들은 언제(when), 어떻게(how) 수행되는가에 관한 특성을 나타낸다.

- 적합성(suitability): 컴포넌트 명세서와 구현 사항의 일치성 검토를 통한 기능적 적합성 평가.

[표 A-1] 적합성

외부 메트릭	측정 공식	해 석	척 도	측정 형태
[1] 기능적 이행능력	명세서 대비 구현 기능 구현율. X=A/B A=테스트 중에 실행이 확인된 기능 수.	0≤X≤1 1.0에 가까울 수록 좋음.	X=절대치	A=개수 B=개수 X=개수/개수
[2] 기능적 안정성	시험 운영이후 수정이 요규되는 비율. X=1-(A/B) A=시험 평가 기간 중 오류가 발견된 기능의 수	0≤X≤1 1.0에 가까울 수록 좋음.	X=절대치	A=개수 B=크기 X=개수/크기

- 정확성(accuracy): 정확한 결과, 정확한 효과를 나타내는 컴포넌트의 능력.
- 컴포넌트간 상호운용성(interoperability): 컴포넌트가 명시된 하나 또는 그 이상의 시스템들과 상호운영 될 수 있는 능력으로써, 치환성(replaceability)과의 혼동을 피하기 위해서 호환성(compatibility)이라는 용어대신 사용되었다.
- 표준 부합성(compliance): 컴포넌트가 관련 CBD 플랫폼상의 규격등에 부합되는 능력.
- 보안성(security): 컴포넌트에 대한 부정 접근을 방지하고, 제한된 정보에 대해 비권한자의 접근을 차단하거나, 또는 공격자에게 이득을 주거나 정당한 사용자에게 서비스를 거부하도록 하기 위해서 정보나 프로그램을 변경하는 것을 방지하는 능력

2) 신뢰성(reliability): 명시된 조건하에서 컴포넌트의 성능이 일정 수준으로 유지되는 능력으로, 컴포넌트는 마모나 노화현상이 발생하지 않으므로 신뢰성의 한계는 요구사항, 설계 또는 구현에서 발생하는 결함에 기인한다. 이러한 결함은 사용시간 보다는 사용방법과 선택된 옵션에 좌우된다.

- 성숙성(maturity): 컴포넌트의 결함(fault)으로 인한 오류(failure)를 회피하는 능력.
- 결함 허용성(fault tolerance): 명시된 인터페이스의 위반 또는 컴포넌트의 오류 발생시에, 컴포넌트가 명시된 수준으로 성능을 유지하는 능력.
- 복구성(recoverability): 고장시에 컴포넌트가 직접 피해를 입은 데이터를 복구하고 성능 수준을 회복하는 능력.
- 가용성(availability): 명시된 사용 조건하에서, 컴포넌트가 정해진 시점에서 요구되는 기능을 수행하기 위한 상태로 이행하는 능력.

3) 사용성(usability): 컴포넌트가 명시된 조건하에서 사용될 때, 사용자가 이해하고, 배우고, 사용하고, 선호할 수 있도록 하는 능력으로써 기능성(functionality), 신뢰성(reliability), 효율성(efficiency)의 몇몇 특성들도 사용성에 영향을 미치지만, 본 가이드라인의 목적을 위해서는 사용성으로 분류하지 않는다.

- 이해성(understandability): 사용자가 컴포넌트의 기능을 수행하는데 있어서 사용 방법과 사용 조건을 이해할 수 있게 하는 컴포넌트의 능력
- 학습성(learnability): 사용자에게 컴포넌트 활용법을 학습하도록 하는 능력.
- 운용성(operability): 사용자가 운영하고 통제하도록 하는 능력.

4) 효율성(efficiency): 명시된 조건하에서 사용된 자원의 양에 비해, 상대적으로 컴포넌트가 요구되는 성능(required performance)을 제공해 줄 수 있는 능력.

- 자원 활용도(resource utilization): 명시된 조건하에서 컴포넌트가 그 기능을 수행할 때, 적절한 시간동안 적절한 자원을 사용하는 능력.

5) 유지보수성(maintainability): 컴포넌트가 수정(modification)을 수용하는 능력으로써, 명시된 기능, 필요성, 환경의 변화에 대한 컴포넌트의 수정(correction), 개선(improvement), 적응(adaption) 등을 포함한다.

- 분석성(analysability): 컴포넌트의 고장원인 또는 결함을 진단하거나, 수정되어야 할 부분을 찾아내는 능력.
- 변경성(changeability): 컴포넌트가 명세서의 절차에 따라 수정을 가능케하는 능력으로써 코딩(coding), 설계(design), 문서화(documenting)의 변경을 포함한다..
- 안정성(stability): 컴포넌트의 수정에 의해 야기되는, 예기치 못한 영향을 최소화 하는 능력.

6) 이식성(portability): 컴포넌트가 한 환경으로부터 다른 환경으로 이식될 수 있는 능력으로써, 이 환경에는 조직, 하드웨어, 컴포넌트가 포함된다.

- 적응성(adaptability): 컴포넌트가 이식 목적에 해당되는 작업이나 수단이외의 것은 이용하지 않고, 명시된 다른 환경으로 이식될 수 있는 능력으로써, 내부수용능력(예를 들면, 스크린 필드, 테이블, 거래량, 레포트 포맷)을 측정하는 능력을 포함한다.
- 설치성(installability): 컴포넌트가 명시된 환경에서 설치되는 능력.
- 부합성(conformance): 컴포넌트를 개발하기 위한 개발 플랫폼의 규격이나 명세에 부합하는 능력
- 치환성(replaceability): 컴포넌트가 명시된 다른 컴포넌트와 치환되어, 그 환경에서 사용되는 능력.

4. 결론

컴포넌트 품질 평가 가이드라인 개발을 통해 얻을 수 있는 산업적 효과는 다음과 같은 측면이 있다 :

- 제품 품질 향상 및 신뢰도 증가
- 컴포넌트 유통을 위한 평가 표준 항목의 구축
- 유통활성화 ; 유통을 위한 최소한의 평가 기능
- 기능 평가에 대한 보완 작업 후 공식적인 품질 평가안으로 발전

정상성 평가는 평가자의 주관에 따라 각기 다른 결과를 얻을 수 있다. 이는 컴포넌트를 개발한 당사자 뿐만 아니라 제 3자가 사용하는 컴포넌트에 대한 평가 항목이나 방법으로는 부적절 하다. 따라서 본 연구에서는 이를 정량적으로 평가할 수 있는 새로운 매트릭을 개발하였다. 또한 패키지 위주의 평가 기준들을 컴포넌트에 적합하도록 재정립하였다. 그러나 실제로 적용된 사례가 없기 때문에 새로이 마련된 평가 기준을 완벽하게 적용하기는 미흡할 수 있다. 따라서 향후 연구 과제에서는 1차 적용된 결과를 토대로

- 평가 보고서 개발
- 실제 시험을 하는 항목과의 관련 사항 보완
- 평가 관련 양식이나 Template 개발
- 메트릭스 측정표
- 시험 계획서
- 시험 결과서등을 보완해야 할 것이다.

[참고문헌]

[1] 장연세, "참조 컴포넌트 아키텍처 모델과 UML 명세화에 대한 연구",한국OA학회, 제6권, 제3호, 2001년

[2] 장연세, 임승린, "UML을 이용한 컴포넌트 명세화에 대한 연구", 수원과학대학기술연구소, 2002년

[3] 장연세, "컴포넌트 시험 평가 가이드라인 개발에 관한 연구",(사)한국소프트웨어컴포넌트컨소시엄, 2001년

[4] 장연세,금영욱, "CORBA와 DCOM의 통합", 한국정보과학회, 1999년 7월호

[5] 장연세, 금영욱,"CORBA 3 프로그래밍 바이블, 장연세", 도서출판 그린, 2000년5월

[6] 김 수동, "ISO 9001 Compliant 객체지향 소프트웨어컴포넌트 품질시스템," 정보과학회지, 제13권 제9호, 1995

[7] 김 종윤·정호원, "소프트웨어의 계량적 품질 평가- ISO/IEC 9126 시리즈 및ISO/IEC 14598 시리즈," 소프트웨어과학회지, 제10권 제2호, 1997

[8] 문 대원 외, 소프트웨어 프로세스 평가지침(안), 한국전산원, 1996

[9] 전산감리 세칙(안), 한국전산원, 1996

[10] 신 동익, "소프트웨어 품질의 국제 표준화 동향," 제1회 소프트웨어 품질관리심포지움, 한국소프트웨어산업협회/한국정보처리학회/한국소프트웨어지원센터, 1997

[11] 안 유환 외, 소프트웨어 생산기술 표준화 연구, 과학기술처, 1995

[12] 이 양규·나관식, "경쟁력 향상을 위한 소프트웨어 품질정책 연구," 제1회 학술세미나 발표논문집, 서원대학교 경영과학연구소,1997

[13] 이주헌 외, 소프트웨어 공정능력평가 조정지침 개발, 시스템공학연구소, 1997

[14] 전 진옥 외, 클라이언트-서버 정보 시스템을 위한 객체 지향 개발 방법론에 관한 연구, 정보통신부, 1997