▨연구논문

설계단계에서의 소프트웨어 품질측정과 평가기법 개발

장영숙
신성대학 품질경영과
권영식
동국대학교 산업시스템공학부

Development of Measurement and Evaluation Technique for Software Quality in Design Phase

Young-Sook Jang
Dept. of Quality Management, Shinsung College
Young-Sik Kwon
School of Industrial and Systems Engineering, Dongguk University

Abstract

It is booming to use computer and information owing to the information society. As software applications have grown, so too has the importance of software quality. Therefore, software quality measurement and evaluation are necessary to satisfy the users who are using computers. The ISO/IEC 9126 defines six quality characteristics and twenty-one subcharacteristics including forty inter quality characteristics. We developed software measurement and evaluation technique using the check list in accordance with the ISO/IEC 9126 in design phase.

We applied idea and concept of Total Quality Management in order to evaluate and measure the quality of software. Namely, it is the concept such as "you should do it right the first time." or "you have to build quality in the process." We executed the quantitative evaluation of software, also had shown the distinctive result in order that users and developers can find the errors easily. We evaluated the quality of academic softwares in order to verify developing technique at S college. As a result of the evaluation, this college has found it necessary to maintain the software as soon as possible because academic systems are not adequate for users at this present time.

1. 서론

1.1 문제 제기 및 연구 필요성

세계 각국은 다가오는 지식 정보사회에서 경제 패권을 잡기 위한 핵심 요소가 정보화임을 인식하고 다양한 추진 전략을 마련하여 일을 추진하고 있다. 농업혁명과 산업혁명이후 정보화의 커다란 흐름이 다가오고 있는 현실이다[4]. 또한 고도 정보화 사회라고 일컫는 오늘날 컴퓨터는 거의 모든 분야에서 사용되고 있으며 그에 따라 엄청난 소프트웨어가 개발되어져 오고 있다. 이에 따라 소프트웨어에 대한 인식과 중요성이 대두되고 있으나 아직 소프트웨어 품질에 대한 우리의 관심은 부족한 실정이며, 저질 소프트웨어로 인한 유지보수 업무의 부담과 비용을 가중시키는 것이 현실이다[2, 27].

사용자의 품질 요구수준이 높아짐에 따라 개발과 더불어 소프트웨어 품질에 대한 중요성을 심각하게 고려해야 할 시기에 이르렀다고 본다. 1991년 7월 미국의 전화불통사건이 일어났는데 그 원인은 2백만 개 이상의 명령어로 구성된 프로그램에서 3줄의 잘못된 명령어가 있었기 때문이라고 밝혀졌다[22]. 국내에서도 사용자 요구에 의해서 프로그래머가 개발한 소프트웨어가 실제로 사용하다 보면 불만이 이만저만이 아니다.

그러나 소프트웨어 개발부서(기관)의 현실은 이러한 이상적인 기대를 반영하지 못하고 있는 것이 현실이다. 소프트웨어 오류는 사용자가 요구한 항목들이 충실하게 반영되지 않아 일어나는 현상이므로 사용자의 요구사항이 빠짐없이 반영되었는지를 확인할 수 있는 방법이 강구되어야 할 것이다. 이와 같이 소프트웨어를 대상으로 품질을 측정하고 품질평가 활동을 하지 않으면 안 되는 국면에 있다[9, 21].

1.2 연구 방법 및 범위

연구방법은 국제표준화 규격인 ISO/IEC 9126[16]에서 제시하고 있는 소프트웨어 품질특성 값을 중심으로 내용을 전개하였다. 개발자 혹은 유지보수자 관점 중심의 내부특성 값에 대한 품질평가 점검표를 개발하여 품질측정을 실시하고, 그 결과 값으로 사용자 관점 중심의 품질부특성 및 주특성 값을 도출해 냄으로서 소프트웨어를 정량적으로 최종 평가하고자 한다.

개발된 기법을 실 업무에 적용해 봄으로써 나름대로 연구 내용의 타당성과 검증을 시도해 보았고, 또한 소프트웨어 품질의 특수성, 즉 말 그대로 'Soft'면을 극복하고 Yourdon[25]이 말한 "측정할 수 없는 것은 관리될 수 없다(You can't manage what you can't measure)"는 원칙에 입각하여 가능하면 정량화 하여 눈으로 볼 수 있는 가시적(可視的)인 소프트웨어 품질평가를 위해 노력했다[3].

연구범위는 일반적으로 소프트웨어 개발부서(조직)가 사용자 부서에서 전산화 요구를 의뢰 받아 개발하는 학사업무의 응용소프트웨어(application software)를 대상으로 한다. 소프트웨어 개발 주기 상에서 유지보수 비용에 지대한 영향을 미치는 설계단계의 산출물에 대해서 소프트웨어 품질 측정방법과 평가기법을 제시하고자 한다.

2. 소프트웨어 품질

2.1 소프트웨어 품질

우리는 흔히 「소프트웨어(software) = 프로그램(program)」이라고 인식하고 있는 것이 보통이지만 그러한 개념들에 많은 변화가 있었고, 이제는 더 이상 「소프트웨어 = 프로그램」이라는 제한된 정의는 사용되지 않고 있다. Mills[20]는 소프트웨어를 사용자 요구사항 문서, 기능명세 문서, 설계 문서, 프로그램, 프로그램 문서, 테스트 문서 및 사용자 매뉴얼의 7가지 형태의 집합이라고 했다. ISO/IEC 9126[16]에서는 소프트웨어를 "컴퓨터 프로그램과 절차, 규칙, 그리고 컴퓨터 시스템의 운용에 관련된 문서와 자료"로 정의하고 있으며, IEEE[14]에서는 "컴퓨터 프로그램과 절차, 규칙, 그리고 컴퓨터 시스템의 운용에 가능한 관련된 문서와 자료"라고 ISO/IEC 9126과 비슷한 개념으로 정의를 내리고 있다.

소프트웨어 품질(software quality)을 IEEE[14]에서는 "소프트웨어가 요구하는 속성들(desired combination of attributes)을 갖고 있는 정도"라고 정의하고 있으며, 여기서 요구하는 속성들이란 신뢰성, 유지보수성, 가용성 등을 말한다. 미 국방성[23]은 "소프트웨어의 명세된 최종 아이템 사용을 수행하는데 가능한 소프트웨어의 속성 정도"라고 정의하였고, Schulmeyer[21]은 "전체 소프트웨어 제품의 사용 적합성"이라고 소프트웨어 품질을 정의하고 있다. 본 논문에서는 국제표준에 맞춰 ISO 8402[17]에서 정의한 "명시적(明示的) 이거나 묵시적(默示的)인 요구를 만족시키는 소프트웨어 제품의 능력과 관련된 소프트웨어 제품의 특성 전체"라는 정의에 따른다.

2.2 소프트웨어 품질특성

소프트웨어 품질을 최초로 정량적으로 평가해 보고자 했던 Boehm[9]은 품질특성(quality characteristics)을 초기운용(as-is utility), 유지보수(maintainability), 이식성(portability)의 세 가지로 분류하여, 일곱 개의 중간구조와 열 다섯 개의 기초 구조를 제안하였다. 소프트웨어 품질 견해를 제품운용(product operations), 제품개정(product revision)및 제품전이(product transition)의 관점으로 본 McCall[19]은 11가지, Evans [12]는 12개의 품질특성을 선택한 후, 소프트웨어 개발 수명주기를 중심으로 성능(performance), 설계(design), 개선(adaptation)의 3개 범주로 구분하였다.

국제표준화기구 ISO(International Organization for Standardization)에서는 소프트웨어 품질보증 규격으로 제시한 ISO 9000-3[18]에 기반을 둔 ISO/IEC 9126[16]에서는 소프트웨어 품질평가에 사용할 수 있는 틀을 제공하고 있으며, 품질특성으로 기능성(functionality), 신뢰성(reliability), 사용성(usability), 효율성(efficiency), 유지보수성(maintainability) 및 이식성(portability)의 6개 품질주특성을 제시하고 있다. <표 1>은 이상에서 언급된 각 학자 및 기구(기관)에서 제시한 소프트웨어 품질특성에 관한 비교를 나타낸 것이다.

<표 1> 각 학자들의 소프트웨어 품질특성 비교

학자(기구)	수	소프트웨어 품질특성
Boehm	15	Device Independence, Self Containedness, Completeness, Robustness, Integrity, Consistency, Accountability, Device Efficiency, Accessability, Communicativeness, Self Descriptiveness, Structureless, Conciseness, Legibility, Augmentability
McCall	11	Correctness, Efficiency, Flexibility, Integrity, Interoperability, Maintainability, Portability, Reliability, Reusability, Testability, Usability
Evans	12	Correctness, Efficiency, Expandability, Flexibility, Integrity Interoperability, Maintainability, Portability, Reliability, Reusability, Usability, Verifiability
DoD	11	Efficiency, Flexibility, Integrity, Interoperability, Maintainability, Portability, Reliability, Responsibility, Reusability, Testability, Usability
Deutsch[11]	15	Correctness, Efficiency, Expandability, Flexibility, Integrity Interoperability, Maintainability, Manageability, Portability, Reliability, Reusability, Safety, Survivability, Usability, Verifiability
ISO/IEC	6	Functionality, Reliability, Usability, Efficiency, Maintainability, Portability

2.3 외부특성과 내부특성

ISO/IEC 14598[15]에서는 품질모형으로 외부품질(external quality)과 내부품질(internal quality)로 구분하고 있다. 외부품질이란 특정 조건하에서 사용된 소프트웨어 제품이 "명시된 그리고 묵시적 요구(stated and implied needs)"를 만족시키는 품질정도를 의미한다. 이에 반하여 내부품질은 "명시된 그리고 묵시적 요구"를 만족시킬 수 있는지를 결정하는 소프트웨어 제품의 자체 속성의 집합을 의미한다.

따라서 외부품질은 <표 2>와 같이 ISO/IEC 9126에서 언급하는 기능성, 신뢰성, 사용성, 효율성, 보수성 및 이식성 6개 항목의 품질주특성 및 합목적성으로 부터 치환성까지 21개 항목의 품질부특성 집합이라고 볼 수 있다. 여기서 기능성은 5개, 신뢰성 3개, 사용성 3개, 효율성 2개, 보수성 4개 및 이식성 4개 항목의 품질부특성으로 구성되어 있다[1, 5, 16].

반면에 내부품질은 ISO/IEC 9126에서 이야기하는 완전성부터 전달성까지 40개 항목의 내부특성 집합이며, <표 3>은 내부특성에 대해서 정의를 한 것이다[1, 5, 16]. 평가 관점 측면에서 보면 외부특성은 사용자 또는 관리자 입장에서, 내부특성은 개발자 혹은 유지보수자에 의해서 평가되는 항목이라 말할 수 있다.

<표 2> 소프트웨어 품질주특성과 품질부특성

품질주특성 품질부특성	정 의
기능성(functionality)	**일련의 기능 존재와 규정된 기능 특성과 관련된 속성의 집합**
합목적성(suitability)	지정된 업무를 수행하는데 필요한 기능이 존재하고 그 기능이 적절한 정도
정확성(accuracy)	소프트웨어가 올바른 결과 또는 합의된 결과를 출력할 수 있는 정도
상호운용성(interoperability)	소프트웨어가 지정된 시스템과 연결되어 상호 작용할 수 있는 정도
표준적합성(compliance)	소프트웨어가 관련 표준, 지침, 규정 등을 따르는 정도
보안성(security)	프로그램과 자료에 대해 인가되지 않은 접근을 방지할 수 있는 정도
신뢰성(reliability)	**명시된 기간동안 명시된 조건에서 성능 수준을 유지하는 소프트웨어 능력과 관련된 속성 집합**
성숙성(maturity)	소프트웨어 결함으로 인해 발생하는 고장의 빈도
장애허용성(fault tolerance)	소프트웨어 결함이나 인터페이스 상의 문제가 있더라도 지정된 수준의 성능을 유지하는 정도
회복성(recoverability)	고장에 의해 영향을 받는 데이터를 복구하고 성능을 회복할 수 있는 정도
사용성(usability)	**사용을 위한 노력과 그러한 사용에 대한 개개인의 심사와 관련된 속성의 집합**
이해성(understandability)	사용자가 소프트웨어의 개념과 응용성을 인식하는 데 필요한 노력의 정도
습득성(learnability)	사용자가 연산 제어, 입출력 등과 같은 적용 법을 배우는데 드는 노력의 정도
운용성(operability)	운용자가 소프트웨어를 적재, 초기화, 운용 및 통제, 종료하는데 드는 노력의 정도
효율성(efficiency)	**규정된 조건에서 소프트웨어의 성능 수준과 사용된 자원의 양 사이에 관련된 속성의 집합**
시간효율성(time behavior)	소프트웨어 기능 수행에 따른 응답 시간, 처리 시간
자원효율성(resource behavior)	소프트웨어의 기능 수행에 따른 자원 소요량 및 자원 사용 기간 등의 수준
보수성(maintainability)	**규정된 수정을 수행하기 위하여 필요한 노력과 관련된 속성의 집합**
분석성(analyzability)	소프트웨어의 고장 원인이나 취약점을 진단하고 변경 부분을 식별하는데 드는 노력의 정도
변경성(changeability)	소프트웨어의 환경 변화로 인해 드는 또는 결함 제거 및 변경에 따라 드는 노력의 정도
안정성(stability)	소프트웨어의 변경으로 인해 발생하는 예상치 못한 위험의 정도
시험성(testability)	변경된 소프트웨어를 검증하는데 드는 노력의 정도
이식성(portability	**다른 환경으로 이전되는 소프트웨어 능력과 관련된 속성의 집합**
환경적응성(adaptability)	소프트웨어를 다른 환경에 적응시키는데 있어 고려된 사항 외에 다른 사항이 추가되지 않는 정도
설치성(installability)	소프트웨어를 지정된 환경에 설치하는데 필요한 노력의 정도
규격적합성(conformance)	소프트웨어가 이식성과 관련된 표준이나 관례를 따르는 정도
치환성(replaceability)	다른 소프트웨어 환경에서 그를 대신해서도 사용될 수 있는 가능성 및 그에 따른 노력의 정도

<표 3> 내부특성(일부분)

코드	내부특성	정 의
01	완전성(完全性) completeness	계획 또는 요구된 기능이 충분히 실현되었는지를 평가
02	추적가능성(追跡可能性) traceability	개발 공정에 따라 요구 명세로부터 실현된 것(기능)으로의 관련을 추적할 수 있는 성질
03	일관성(一貫性) consistency	설계기법이나 표기법, 용어, 기호 등이 통일되었는지를 평가
04	자기기술성(自己記述性) self descriptiveness	기능 및 기능과 기능의 연관을 설명하는 성질
05	무모순성(無矛盾性) coherence	시스템과 프로그램이 가지는 동일 기능에 모순이 없는 성질. 프로그램과 도큐먼트 사이에 모순이 없는 성질.
06	계산정확성(計算正確性) Accuracy	계산결과 및 출력이 요구하는 정도를 달성하는 성질
:	:	:
35	확장성(擴張性) Expandability	사양의 추가, 변경에 대하여, 용이하게 대응 가능하도록 준비되어 있는 성질
36	제품관리성(製品管理性) product management	소프트웨어의 개개 부품, 도큐먼트에 대한 구성, 이들의 정합성과 이력 등 제품의 관리가 쉽도록 하는 성질
37	소프트웨어 독립성 s/w independence	소프트웨어가 특정의 소프트웨어 환경(os compiler, 언어, utility 등)에 의존하지 않고 동작 가능한 성질
38	기계독립성(機械獨立性) machine independence	시스템이 특정의 하드웨어 환경(시스템구성, 기종, 장치, 단말 등)에 의존하는지를 평가
39	데이터독립성(獨立性) data independence	시스템이 특정 데이터 환경(데이터,DB,DBMS 등)에 의존하는지를 평가
40	전달성(傳達性) communicativeness	시스템의 입출력 형식이나 내용이 어느 정도 사용이 용이하게 통일되어 있는지를 평가

3. 측정 및 평가기법

3.1 품질특성 선정

소프트웨어 품질 측정 및 평가를 위해서는 먼저 대상이 되는 품질특성을 선정해야 하는데, 본 논문에서는 ISO/IEC 9126[16]에서 제시하고 있는 소프트웨어 개발 주기별 내부특성간의 관계 <표 4>를 참조하여 선정하기로 한다. 따라서 설계단계에서는 완전 성, 추적가능성, 일관성, 자기기술성, 무모순성, 통신절차공통성, 엑세스가능성, 엑세스제어성, 엑세스감사성, 견고성, 무결성, 단순성, 계측성, 통일성, 표현성, 계층성, 설명성, 주목성, 간결성, 선택성으로 21개 항목이 해당된다[1, 5].

<표 4> 소프트웨어 개발 주기별 내부특성 관계

구분 / 코드 내부특성	개발 주기			
	요구분석 단계	설계단계	코딩단계	테스팅 및 유지보수 단계
01 완전성	○	○	○	
02 추적가능성	○	○	○	
03 일관성	○	○	○	
04 자기기술성	○	○	○	○
05 무모순성	○	○	○	○
06 계산정확성	○		○	○
07 데이터공통성	○			○
08 통신절차공통성	○	○	○	
09 엑세스가능성	○	○	○	
10 엑세스제어성	○	○	○	
11 엑세스감사성	○	○	○	
12 견고성	○	○	○	○
13 무결성	○	○	○	○
14 모듈성	○	○	○	
15 단순성		○	○	○
16 계측성		○	○	○
17 자기포함성			○	○
18 통일성		○	○	○
19 표현성		○	○	○
20 계층성	○	○	○	
21 설명성	○	○	○	○
22 비유성	○		○	○
23 완비성			○	○
24 주목성	○	○	○	○
25 적시성	○			○
26 적량성(기억성)	○		○	○
27 간결성		○	○	○
28 선택성	○	○	○	○
29 유도성			○	○
30 환경적합성	○		○	○
31 성력성			○	○
32 환경적합성			○	○
33 실행효율성			○	○
34 자원효율성			○	○
35 확장성	○		○	○
36 제품관리성			○	○
37 S/W독립성			○	○
38 기계독립성			○	○
39 데이터독립성			○	○
40 전달성			○	○

요구분석 단계에서는 완전성을 포함하여 23개 항목, 코딩 단계는 완전성을 포함한 38개 항목, 테스팅 및 유지보수 단계에서는 자기기술성을 포함한 31개 항목이 해당된다. 특히 자기기술성, 무모순성, 견고성, 무결성, 설명성, 주목성, 및 선택성 7개 항목은 소프트웨어 개발 모든 단계에 해당되므로 중요한 항목이라 하겠다.

3.2 평가 점검표 설계 및 내부특성 값 산출

평가 점검표(check list)는 내부특성을 평가할 수 있도록 평가항목을 구분 기술하며, 이때 정량적, 정성적으로 평가할 수 있도록 항목은 상세하게 기술해야 한다. 평가 등급은 평가자의 주관성을 가능한 한 배제하기 위하여 5단계 평가등급(우수/양호/보통/미약/불량)으로 하고, 평가항목에 따라 평가 결과를 평가등급 난에 표시(∨)를 하면 된다. 만약 평가항목에 해당이 없으면 「해당 없음」 난에 표시하고, 이는 평가점수에 들어가지 않는다. 평가등급별로 획득한 점수를 해당 소계 난에 기록하고, 등급별 점수를 합산하여 총점 난에 기록하면 된다. <표 5>는 이러한 내용을 반영한 평가 점검표 양식이 되겠다[6].

<표 5> 평가 점검표(check list) 양식

내부특성 :		해당□에 ∨표시를 한다					
		우수	양호	보통	미약	불량	해당 없음
No.	평 가 항 목	10	8	6	4	2	
:	:	□	□	□	□	□	□
소계							
총점							

평가된 총점은 식 (3.1)의거 내부특성 값을 산출하면 된다[3].

$$T = \sum_{i=1}^{n} P_i / 10 \cdot n \times 100 \qquad (3.1)$$

T : 내부특성 총점 백분율(%)

P : 평가점수

n : 평가항목 수(단, 「해당 없음」 항목은 제외)

$(i = 1, 2, 3, \cdots, n)$

3.3 품질부특성, 주특성 및 최종 소프트웨어 값 산출

외부특성인 품질부특성과 품질주특성 값 산출은 ISO/IEC 9126[16]에서 제시하고 있는 <표 6>과 같이 내부특성과의 관계를 이용하여 산출하면 된다. 관계 정도는 표에서 와 같이 강한관계, 보통관계 및 약한관계 3종류로 구분하고 있다[5].

합목적성은 완전성, 추적가능성 및 자기기술성과는 강한관계를 일관성 및 무모순성은 중간관계를 맺고 있다. 이러한 품질부특성과 내부특성간의 관련 강도는 내부특성에 대한 측정결과로부터 품질부특성을 산출하는 과정에서 가중치를 부여할 때 적용된다. 강한 관련성을 갖는 특성들간에는 큰 가중치 값이 적용되며, 관련성이 약한 경우는 작은 가중치 값을 부여하여 평가하면 된다. 물론 가중치 합($\sum W_i$)이 1이 되야 한다.

<표 6> 내부특성과 품질부특성 및 주특성간의 관계

품질주특성	기능성					신뢰성			사용성			효율성		보수성				이식성			
품질부특성 / 코드 내부특성	합목적성	정확성	상호운용성	표준적합성	보안성	성숙성	장애허용성	회복성	이해성	습득성	운용성	시간효율성	자원효율성	분석성	변경성	안정성	시험성	환경적응성	설치성	규격적합성	치환성
01 완전성	●	◐				●	○	○													
02 추적가능성	●					●	○	○						●	●	◐	○				
03 일관성	◐	◐				●	◐	●						●	●	◐	◐				
04 자기기술성	●	●				●		●						●	●		◐		◐		
05 무모순성	◐	●																	◐		
06 계산정확성		●	◐																		
:	:	:	:	:	:	:	:	:	:	:	:	:	:	:	:	:	:	:	:	:	:
35 확장성														◐	◐	○	○				
36 제품관리성														○	●						
37 S/W독립성														◐	◐		◐	●	●	◐	●
38 기계 독립성														◐	◐		◐	●	●	◐	●
39 데이타 독립성																		●	●	●	●
40 전달성																		○	○		○

● 강한관계 ◐ 보통관계 ○ 약한관계

식 (3.1)에서 산출된 내부특성 값을 가지고 품질부특성 값을 아래 (3.2)식에 의해서 산출할 수 있다.

$$S = \sum_{i=1}^{n} (T_i \times W_i) \quad (3.2)$$

S : 품질부특성 값
T_i : 내부특성 값
W_i : 내부특성 항목 가중치 ($\sum_{i=1}^{n} W_i = 1$)
n : 내부특성 항목 수 ($i = 1, 2, 3, \cdots, n$)

또한 식 (3.2)에서 얻어진 품질부특성 값을 이용하여 식 (3.3)에 따라 품질주특성 값을 얻을 수 있다.

$$M = \sum_{i=1}^{n} S_i / n \quad (3.3)$$

M : 품질주특성 값
S_i : 품질부특성 값
n : 품질부특성 항목 수 ($i = 1, 2, 3, \cdots, n$)

마지막으로 식 (3.3)에서 산출된 품질주특성 값을 이용하여 식 (3.4)에 따라 최종적으로 소프트웨어 값을 산출해 낼 수 있다.

$$F = \sum_{i=1}^{n} M_i / n \quad (3.4)$$

F : 최종 소프트웨어 값
M_i : 품질주특성 값
n : 품질주특성 항목 수 ($i = 1, 2, 3, 4, 5, 6$)

본 연구에서는 품질부특성 값에서 품질주특성 값, 품질주특성 값에서 최종 소프트웨어 값을 산출할 때 가중치 적용을 않기로 하며, 향후 연구 과제로 남겨 놓는다.

3.4 결과분석 및 시정조치

<표 7>은 식 (3.4)에 의해 산출된 최종 소프트웨어 품질 값으로 소프트웨어 개발 시 등급에 따라 프로젝트 관리방법을 제시한 것으로, 원리는 제조 현장에서 흔히 사용되는 공정능력 지수를 소프트웨어 개발 프로젝트 관리지수로 응용해 본 것이다[3].

표에서 보면 1, 2등급은 현상 유지를 하면서 프로그램 개발을 진행해도 좋으나, 3등급 이하는 사용자 요구 사항이 충분히 반영되지 못한 상태에서 진행되므로 개발 후 사용자 이용 시 고객의 불만과 동시에 엄청난 유지보수 비용이 발생할 것으로 판단된다. 따라서 이때는 개발을 계속 진행하기보다는 개발 시 관련되는 문서를 토대로 다시 한 번 사용자 요구 사항이 충실히 반영되었는지를 점검해야 할 것이다. 이러한 결함(error)의 조기 발견과 시정조치를 위해서 프로그램 개발 관련 당사자 즉, 사용자, 개발자 및 관리자가 눈으로 쉽게 파악할 수 있도록 가시화 하여 나타낼 필요가 있다. 가시화를 위해서는 품질관리 활동에서 많이 쓰이는 QC 7가지 도구[7] 즉, 그래프(막대, 띠, 꺽은선 등), 파레토도(pareto diagram), 특성요인도(cause and effect diagram), 산점도(scatter diagram), 층별(stratification), 체크시이트(check sheet) 및 히스토그램(histogram)과 같은 그림을 이용하면 효과적이다. 또한 품질평가 값이 어떤 항목에 치우쳐 있는가를 쉽게 파악할 수 있는 방사형 그림(radar chart)도 이용하면 쉽게 소프트웨어 품질지표를 알 수 있으며 조속한 결함(error) 발견이 용이하다[8, 26].

<표 7> 소프트웨어 품질 등급에 따른 개발 프로젝트 관리방법

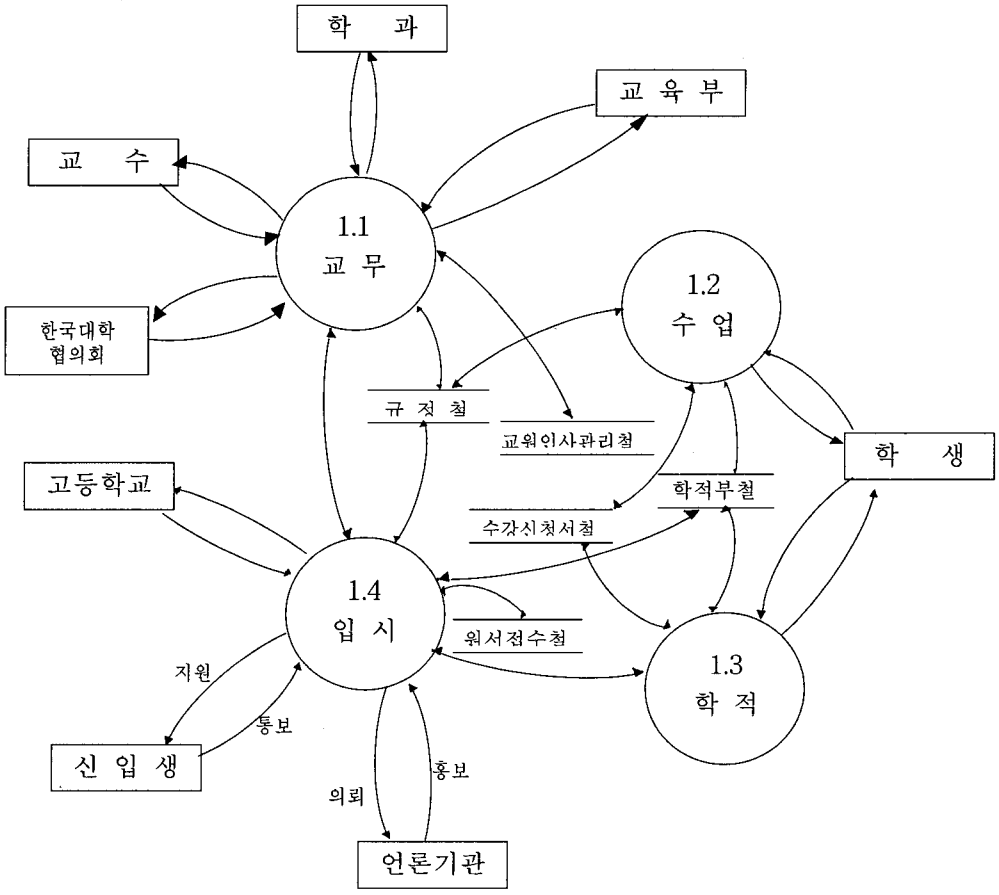
최종 값	등급	결 과	관 리 방 법
100~90	1등급	충족도가 매우 좋음	관리 간소화나 개발비용 절감방법 고려
89~80	2등급	충족도가 높음	이상적인 상태이므로 유지해감
79~70	3등급	일부항목 수정요망	충족도가 충분한 사항은 개선을 요함
69~60	4등급	충족하지 않음	오류가 많이 발생. 관리개선을 요함
60미만	5등급	전면적인 개선요망	긴급대책 강구

이와 같이 평가과정에서 나타난 문제점이나 오류(error)를 신속·정확하게 feedback 시켜 줌으로써 조기에 시정할 수 있어 결과적으로 유지보수비를 절감할 수 있고, 또한 경제적인 고품질의 소프트웨어를 획득할 수 있다고 본다.

4. 적용 사례

4.1 개요

개발된 기법을 S대학의 주요 학사업무중 교무, 수업, 학적 및 입시업무 4종에 대해 적용해 보았다. 학사업무의 개략적인 내용을 <그림 1>과 같이 자료흐름도(DFD: Data Flow Diagram)로 나타내었다[24].



< 그림 1 > 학사업무 자료흐름도

설계단계의 품질평가 대상으로는 입출력 설계서(I-P-O), 조회화면 설계서, 데이터베이스 설계서, 상세설계 작업일정 계획표, 상세설계 조직 구성도, 서브시스템 명세서, 데이터 용량 산출서, 프로그램 및 모듈설계, 단위 프로그램 설계서, 프로그램 일람표, 시스템 분석 산출물 등이 있다.

4.2 내부특성 값 산출

<표 8> 평가 점검표는 개발된 <표 5> 평가 점검표 양식에 따라 내부특성 중 모듈성을 교무업무에 대해서 측정 및 평가한 내용이다.

<표 8> 평가 점검표

내부특성 : 14 모듈성(modulity)		해당□에 ∨표시를 한다					
		우수	양호	보통	미약	불량	해당 없음
No.	평 가 항 목	10	8	6	4	2	
01	· 시스템이 모듈형식으로 설계 되었는가?	□	□	□	☑	□	□
02	· 각 모듈에 의해 호출되는 모듈수는 허용된 범위(예:7)를 초과하지 않는가?	□	□	☑	□	□	□
03	· 각 모듈은 허용된 크기 내에서 설계되었는가? 허용범위가 명시되지 않았다면 코드의 합리적 문장수(예:100미만)를 초과하지 않는가?	□	□	☑	□	□	□
04	· 각 모듈은 모듈 계층도의 상위 수준으로 부터만 호출되는가?	□	☑	□	□	□	□
05	· 각 모듈은 하나의 기능만을 수행하는가?	☑	□	□	□	□	□
06	· 호출된 각 모듈은 그 기능을 수행한 후 원래의 상태로 환원되는가?	□	□	☑	□	□	□
소 계		10	8	18	4	0	
총 점		40					

평가 내용을 살펴보면 6개 평가 항목 중 미약(4점) 1개, 보통(6점) 3개, 양호(8점) 1개 및 우수(10점) 1개 항목으로써 모듈성의 총점은 식 (3.1)에 따라 다음과 같이 산출하면 된다.

$$T_{(모듈성)} = \{(4+6+6+8+10+6)/10 \times 6\} \times 100$$

$$= 66.7\ \%$$

위와 같은 요령으로 4종 업무에 대하여 내부특성 값을 정리하면 <표 9>와 같다.

<표 9> 설계단계 업무별 평가 결과

코드	내부특성	교무	수업	학적	입시
01	완전성	83.5	84.5	86.9	80.4
02	추적가능성	82.1	86.2	88.2	80.5
03	일관성	76.4	78.0	82.1	72.4
04	자기기술성	72.1	73.4	75.7	70.8
05	무모순성	82.9	89.9	82.5	83.2
08	통신절차공통성	79.4	88.6	87.7	79.8
09	엑세스가능성	77.7	78.3	76.5	74.5
10	엑세스제어성	76.4	72.4	77.4	78.9
11	엑세스감사성	67.1	65.3	75.6	65.9
12	견고성	72.3	72.2	74.3	79.0
13	무결성	69.6	63.4	70.0	67.6
14	모듈성	66.7	89.5	78.3	72.4
15	단순성	74.6	65.4	74.6	73.6
16	계측성	69.2	64.2	72.7	63.4
18	통일성	87.3	87.5	86.5	82.2
19	표현성	82.7	83.2	82.0	77.4
20	계층성	85.6	90.4	84.2	79.8
21	설명성	80.4	85.3	86.9	79.8
24	주목성	84.2	79.4	86.3	70.4
27	간결성	79.8	88.8	85.3	72.6
28	선택성	73.7	75.4	77.7	68.7

4.3 품질부특성, 주특성 및 최종 소프트웨어 값 산출

품질부특성 값은 식 (3.2)에 따라 <표 9>의 평가 결과 값을 이용하여 산출할 수 있으며, 교무업무중 합목적성 값을 계산해 보면,

$$S_{(\text{합목적성})} = (83.5*0.25)+(82.1*0.25)+(76.4*0.125)+(72.1*0.25)+(82.9*0.125)$$
$$= 79.34$$

이다. 여기서 가중치 부여는 완전성, 추적가능성 및 자기기술성은 강한 관계이므로 0.25, 일관성과 무모순성은 보통관계이므로 0.125를 부여하였다. 같은 요령으로 정확성 79.26, 상호운용성 79.40, 표준적합성 79.72 및 보안성 73.22 값을 얻을 수 있으며, 이를 정리하면 <표 10>과 같다.

<표 10> 교무업무 기능성 품질부특성 값

품질부특성	코드	내부특성	평가치	가중치	결과치	산출과정
합목적성	01	완전성	83.5	0.25	79.34	83.5*0.25+82.1*0.25+ 76.4*0.125+72.1*0.25+ 82.9*0.125
	02	추적가능성	82.1	0.25		
	03	일관성	76.4	0.125		
	04	자기기술성	72.1	0.25		
	05	무모순성	82.9	0.125		
정확성	01	완전성	83.5	0.125	79.26	83.5*0.125+82.1*0.25 76.4*0.125+72.1*0.25 82.9*0.25
	02	추적가능성	82.1	0.25		
	03	일관성	76.4	0.125		
	04	자기기술성	72.1	0.25		
	05	무모순성	82.9	0.25		
상호운용성	08	통신절차공통성	79.4	1.0	79.40	79.4*1.0
표준적합성	08	통신절차공통성	79.4	0.6	79.72	79.4*0.6+77.7*0.4
	09	엑세스가능성	77.7	0.4		
보안성	08	통신절차공통성	79.4	0.1	73.22	79.4*0.1+77.7*0.1+ 76.4*0.3+67.1*0.3+ 72.3*0.2
	09	엑세스가능성	77.7	0.1		
	10	엑세스제어성	76.4	0.3		
	11	엑세스감사성	67.1	0.3		
	12	견고성	72.3	0.2		

여기서 가중치 부여는 ISO/IEC 9126에서 제시하고 있는 내부특성과 외부특성인 품질부특성, 품질주특성과의 관련정도 <표 6>에 따라 평가자의 주관에 따라 상호 비교하여 적용했음을 밝혀 둔다[3]. ISO에서도 현재 이 부분에 대해 연구 중(Draft International Standard 상태)에 있고, 필자도 이 부분에 대하여 실 업무와 연계하여 어떻게 합리적으로 부여할 것인가에 대한 설문지를 설계 중에 있으며 계속 연구 중에 있다.

품질주특성 값은 식 (3.3)에 따라 산출하면 되고, 예로 교무업무 기능성 값은 다음과 같다.

$$M_{(기능성)} = (79.34 + 79.26 + 79.40 + 79.72 + 73.22) / 5$$
$$= 77.99$$

최종적인 소프트웨어 결과 값은 식 (3.4)에 따라 산출하며 예로 교무업무 결과 값은 다음과 같다.

$$F_{(교무)} = (77.99 + 74.04 + 82.01 + 73.85 + 78.19) / 5$$
$$= 77.21$$

같은 요령으로 업무별 품질주특성 값과 소프트웨어 최종 값(평균)을 집계하면 <표 11>과 같이 정리된다.

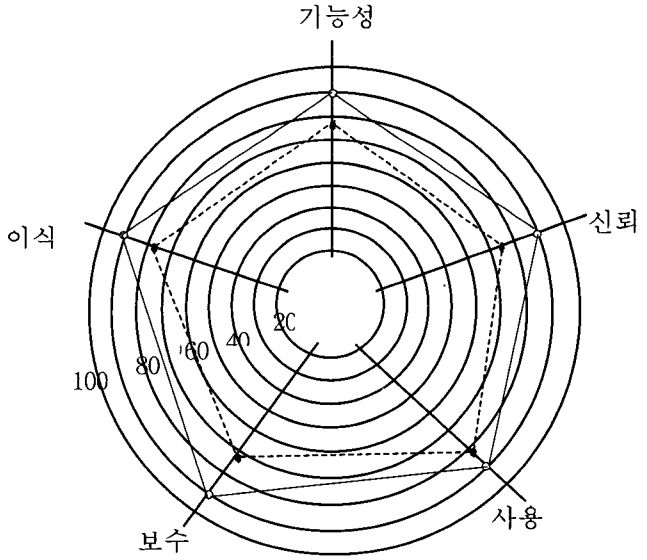
여기서 효율성 값이 없는 것은 <표 6>에서 보듯이 설계단계에서는 효율성과 내부특성간에는 관련성을 갖고 있지 않기 때문이다.

<표 11> 업무별 품질주특성 결과 값

구분	교무	수업	학적	입시
기능성	77.99	82.04	82.82	77.45
신뢰성	74.04	76.48	78.30	73.64
사용성	82.01	84.31	83.22	75.83
보수성	73.85	77.13	79.08	72.13
이식성	78.19	86.66	84.83	78.35
최종 값	77.21	81.32	81.65	75.48

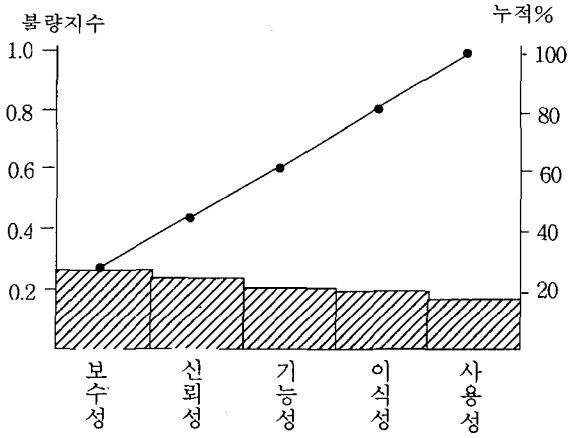
4.4 평가결과 분석/가시화 및 프로젝트 관리방법

평가결과를 사용자, 개발자 및 관리자가 쉽게 파악할 수 있도록 QC 7가지 도구 및 Radar Chart 등을 이용하여 가시화 한 것으로 <그림 2>는 교무업무에 대한 품질주특성 값을 Radar Chart 나타냈다. 교무업무의 품질주특성 값이 사용자 충족도가 높은 실선으로 표시한 1등급이상이 되려면 평가 값인 점선과의 차이만큼 품질개선(QI: Quality Improvement) 활동을 하지 않으면 안 된다.



< 그림 2 > 교무업무 품질주특성 값의 Radar Chart

<그림 3>은 품질주특성 값을 불량지수로 환산하여 파레토도로 나타낸 것이다. 그림을 보면 품질개선 활동 우선 순위를 보수성, 신뢰성, 기능성, 이식성 및 사용성 순으로 진행하는 것이 경제적임을 알 수 있다.



<그림 3> 교무업무 품질주특성 값 불량률 파레토도

<표 7>에서 보면 4종 업무중 수업과 학적 업무는 80점 대 이상으로 2등급이며, 교무와 입시 업무는 80점 미만의 3등급으로써 사용자의 불만과 현재 유지보수비가 증가하고 있는 실정이다. 따라서 교무 및 입시업무는 충족도가 불충분한 사항이 많아 관리개선이 시급한 소프트웨어로 결론을 내릴 수 있으며 유지보수가 조속히 이루어져야 한다.

5. 요약 및 결론

본 연구를 통하여 소프트웨어 개발 주기 상에서 설계단계에서의 산출물에 대한 소프트웨어 품질측정 및 평가기법을 제시하였다. 이를 위해서 국제표준화 규격인 ISO/IEC 9126에서 언급되고 있는 소프트웨어 품질특성을 중심으로 평가항목을 선정하고, 그에 따른 평가 점검표를 개발하여 내부특성 값을 도출하였다. 내부특성 값을 기초로 하여 외부특성 값인 품질부특성 값과 품질주특성 값을 산출해 내고 최종적으로 소프트웨어 전체를 평가하는 내용이다. 개발된 기법을 실 업무에 적용하여 나름대로 검증도 시도해 보았다.

본 연구의 주 내용을 요약하고 결론을 내리면 다음과 같다.

첫째, 소프트웨어 품질측정과 평가에 종합적 품질경영(TQM : Total Quality Management) 개념을 도입하여 적용하였다. 산업공학 관점에서 흔히 말하는 "제품을

처음부터 잘 만들어라(do it right the first time)" 또는 "제작 공정에서 품질을 주입하라(build quality into process)"라는 원칙에 따라 소프트웨어를 개발하고 평가함으로써 저렴하고 질 높은 품질을 획득할 수 있을 것으로 본다[10, 13].

둘째, 품질평가를 정량적으로 측정하고 결과를 가시적으로 표현하여 소프트웨어 개발 구성원들이 쉽게 문제점을 파악할 수 있도록 하였다. 이를 위해서 QC 7가지 도구 및 Radar Chart 등을 이용하여 품질특성 값이 어느 정도 인지를 빨리 인지하여 결함에 대한 시정을 조기에 조치함으로써 유지보수 비용을 줄일 수 있다.

셋째, 적용 사례를 통하여 S대학의 학사업무 프로그램은 사용자(고객) 만족을 위해서 유지보수가 필요하다고 판단된다. 수업과 학적업무는 2등급 이상으로 어느 정도 양호한 편이나, 교무와 입시업무는 3등급으로 사용자 충족 도에 불충분한 사항으로 개선이 요구된다고 볼 수 있다.

앞으로 연구 과제로는 ISO/IEC 9126에서 제시하고 있는 내부특성을 모두 측정하고 평가하는 것은 비효율적이므로 측정대상을 합리적으로 선정하는 방법과 또한 내부특성 값에서 품질부특성 값, 품질부특성 값에서 품질주특성 값, 품질주특성 값에서 최종 소프트웨어를 평가하는데 있어서 보다 합리적인 가중치 부여방법이 연구되어야 할 것으로 본다. 또한 본 연구의 내용이 응용 소프트웨어 중심과 적용대상이 대학의 학사업무로 한정했기 때문에 이를 일반화하는데는 다소 무리가 있을 것으로 생각되나 추가 연구가 계속 이루어지면 여러 분야에 적용 가능할 것으로 확신합니다.

참고문헌

[1] 시스템공학연구소(1997), 「소프트웨어 품질평가 기술개발(Ⅰ)」, 정보통신부.

[2] 이주헌(1991),「프로젝트 관리론-소프트웨어 개발 · 관리 사례 중심으로-」법영사.

[3] 장영숙, 권영식(1999), "설계단계에서의 소프트웨어 품질평가 기법개발," 대한품질경영학회, 1999년 춘계학술대회, pp. 83-92.

[4] 정보통신부(1998), 「정보화 촉진에 관한 동향과 시책에 관한 내용」. http://www.nca.or.kr/NCA98/news/98ap-0.htm.

[5] 한국전산원(1998), 「정보시스템 품질관리 감리지침에 관한 연구」.

[6] 한국표준협회(1994), 「기업체질 강화를 위한 TQC 추진 체크리스트」.

[7] 한국표준협회(1988), 「QC 7가지 도구」.

[8] Adrian Burr and Mal Owen(1996), *Statistical Methods for Software Quality,* International Thomson Computer Press.

[9] Boehm, B. W.(1981), *Software Engineering Economics,* Prentice Hall.

[10] Crosby P. B.(1979), *Quality is Free,* A Mentor Book.

[11] Deutsch, M. S. & R.R. Willis(1988), *Software Quality engineering,* Prentice-Hall, Englewood Cliffs.

[12] Evans, M. W. & J. J. Marciniak(1987), *Software Quality Assurance and Management,* John Willey & Sons.
[13] Feigenbaum A. V.(1991), *Total Quality Control,* 3rd ed., McGraw-Hill.
[14] IEEE-STD-729(1983), "Software Engineering Technical Committee of the IEEE Computer society - IEEE Standard Glossary of Software Engineering Terminology," New York.
[15] ISO/IEC 14598(1995), "Information Technology - Software Product Evaluation."
[16] ISO/IEC 9126(1997), "Information Technology - Software quality characteristics and metrics -
Part 1 : Quality characteristics and sub-characteristics
Part 2 : External Metrics
Part 3 : Internal Metrics."
[17] ISO 8402(1986), "Quality - Vocabulary."
[18] ISO 9000-3(1991), "Quality management and quality assurance standards, Part 3 : Guidelines for the application of ISO 9001 to the development, supply and maintenance of software."
[19] McCall J. A. et al.(1978), "A Framework for the Measurement of Software Quality : The Proceedings of the ACM Software Quality Assurance."
[20] Mills H. D.(1980) "Principles of Software Engineering," IBM System Journal, Vol. 19, No. 4.
[21] Schulmeyer G. G. and Mcmanus J. I.(1987), *Handbook of Software Quality Assurance,* Van Nostrand Reinhold Company Inc., N.Y.
[22] The Editors of Business Week(1994), "The Quality Imperative," McGraw-Hill Inc.
[23] U.S. Department of Defense(1985), "DOD-STD-2168-Defense system Software Assurance Program -," Washington, DC.
[24] Weinberg G. M.(1980), *Structured Analysis,* Yourdon Press.
[25] Yourdon E.(1979), *Managing the System Life Cycle,* Yourdon Press.
[26] 菅野 孝男(1996), *實務者のための ソフトウェア 外注管理,* コンピュータ エージ社.
[27] 森口繁一(1990), *ソフトウェア 品質管理 ガイトブック,* 日本規格協會.