퍼지 및 지능시스템학회 논문지 2002, Vol. 12, No. 1, pp. 67-72

프로그램 모듈의 품질평가 함수 산출에 관한 연구

A Study on the quality estimate function of the program module

김혜경* · 최완규** · 이성주***
Hye-Kyoung Kim, Wan-Kyoo Choi and Sung-Joo Lee

* 안성여자기능대학 인터넷미디어과
** 광주대학교 컴퓨터전자통신공학부
*** 조선대학교 전자계산학과

요 약

고수준의 정보 서비스를 제공하기 위해서는 소프트웨어의 고품질화를 추구하여야 한다. 기존에 개발된 품질측정방법들은 모듈내에 포함되어 있는 요인항목들을 서로 다른 관점에서 개별적으로 측정하고있어 통합적인 평가방법이 필요하다. 따라서 본 논문에서는 다수의 측정 방법들을 모두 수용할 수 있는 모델을 제안한다. 제안된 모델은 비율척도들을 선택하고, 러프논리를 이용하여 그들의 중요도를 산출한다. 다음으로 모듈의 품질도를 측정하기 위해서 퍼지적분을 이용하여 척도들의 중요도와 측정값을 종합한다. 마지막으로 척도들과 산출된 품질도의 상관관계를 분석하고, 통계적 방법으로 제안된 모델의 타당성을 보인다.

Abstract

In order to offer a service of highly qualified information, software of good quality should be designed and developed. We need the united estimation method because the existing approaches for the quality measurements measure the attributes at the different viewpoint individually. Therefor, this paper propose the model that handles many methods of measurements. Our model selects the ratio scales and calculates the relative significance of them by using the rough logic. As a next step, the proposed method integrates the significance of scales and the measured value of them based upon the fuzzy integral. Finally, the correlations are analyzed between the existing scales with our measurements and our model is validated through statistical technique.

Key Words : 러프 집합, 품질, 복잡도, 중요도, 퍼지적분

1. 서 론

고수준의 정보 서비스들이 주로 소프트웨어의 품질과 신뢰성에 절대적으로 의존하고 있는 현실에 맞춰 소프트웨어의 대외 경쟁력을 확보하기 위해서는 소프트웨어의 고품질화를 추구하는 것은 필수적인 일이다. 이미 소프트웨어의 품질과 관련된 메트릭스와 소프트웨어의 품질을 정량적으로 측정할 수 있는 소프트웨어 복잡성 측정 모델들이 다수 제안되어 있으며 품질 평가를 위한 도구로 개발되어 활용되고 있다[4]. 그러나 지금까지의 품질 평가 방법들은 하나의 모듈내에 포함되어 있는 요인 항목들을 서로 다른 관점에서 개별적으로 측정하고 있으므로 단순히 정성적인 차원의 체크리스트 수준을 벗어나지 못한 실정이며, 측정치만을 가지고 여러 특성들을 종합한 품질 평가와 다른 소프트웨어들간의 정량적 비교가 불가능하였다[6,7]. 이러한 점을 극복하기 위해서는 효율적이고 타당성 있는 통합적인 평가방법을 개발하여 정확한 품질 평가를 수행함으로써 고품질의 소프트웨어 개발을 추진할 수 있도록 하는 것이 바람직하다. 본 논문에서는 기존에 개발된 품질 측정방법들을 하나의 모듈에 적용하여 평가함으로써 하나의 시스템 내에서 다수의 측정 방법들이 소스 프로그램의 정적인 분석을 통해 측정하고자 하는 관점들을 함께 수용할 수 있는 모델을 제안한다. 2장에서는 기존 품질척도가 안고있는 문제점들을 분석하고 3장에서는 본 논문에서 제시하고자 하는 품질 평가모델에 대해 다루었다. 4장에서는 실제로 사용되고 있는 소스 프로그램을 대상으로 하여 실험하고, 측정 결과 값이 적합한지를 검증하였다. 마지막으로 5장에서는 품질평가 함수의 기대효과와 앞으로 연구할 방향 등을 제시하였다.

접수일자 : 2001년 4월 23일
완료일자 : 2001년 11월 26일

2. 관련 연구

2.1 기존 품질척도의 문제점

기존의 품질 척도에 관한 연구들을 살펴보면, 다음과 같은 문제점들을 공통적으로 내포하고 있다.

첫째, 크기, 제어구조, 데이터구조를 기반으로 하는 메트릭들은 소프트웨어 품질 측정을 완전하게 측정 불가능하다. 예로 들면, Halstead의 software science는 여러 모

듈로 이루어진 프로그램의 전체 복잡도는 증가하기 마련인데, 이들 모듈 사이의 상호작용을 설명해주지 못하며[9,18], 따라서 이를 보완해주는 척도가 필요하다. 또한 Mccabe의 순환수는 제어구조 상의 복잡도를 측정하는 방법으로는 적당하나 프로그램 크기에 따른 복잡도를 정확하게 반영해주지 못하고 있다.

둘째, 혼합적 척도들은 각 척도에 맞는 요소들만을 기준으로 하여 측정하였다. Hansen의 복잡도 척도는 McCabe의 Cyclomatic number와 Halstead의 유일한 연산자수(n1), 두 요소만을 고려하였으며[11], Ovideo의 복잡도 척도는 데이터와 제어 흐름 측면의 요소만을 고려하였다[10]. 그러나 많은 다양한 요인들이 소프트웨어 복잡도에 영향을 주므로 가능한 이러한 요인들을 모두 고려한 척도가 필요하다[1].

셋째, 대부분의 척도들은 비율척도보다는 순위척도 또는 명목 척도를 전제로 하고 있다. 프로그램 복잡도 측정에는 네가지 방법의 척도(scale)가 채택된다[17]. 프로그램 이해성을 "어렵지 않다" "약간 어렵다" "어렵다" "매우 어렵다" 등으로 분류하는 명목(norminal)척도, 프로그램 A,B,C 모두가 "약간 어렵다"라고 말할 수 있을 뿐 아니라 B가 A보다 "조금 더 어렵다"라고 분류할 수 있는 순위(ordinal)척도, "프로그램 A가 B보다 10단위만큼 어렵다"라고 말할 수 있는 구간(interval)척도, 두 프로그램을 비율로서 나타낼 수 있는 비율(ratio)척도 등이다. 기존의 대부분의 척도들은 단순히 소프트웨어가 "어렵다" "어렵지 않다"라고 분류하는 명목 척도 또는 "~보다 어렵다"라고 말할 수 있는 순위 척도에 해당된다. 명목 척도는 매우 단순하고, 순위 척도는 정량적으로 어느 정도 어려운지를 나타내주지 못하며, 구간 척도는 단위(unit)의 정의가 필요하다는 단점을 안고 있다. 비율척도는 융통성(flexible)이 있으며 실제적(preactical)이기 때문에 소프트웨어 복잡도 측정에 가장 적합하고 좋은 척도라고 할 수 있다. 기존 품질 측정에 관련된 연구들에서는 명목척도 또는 순위척도에 적합한 척도들을 제시하였다[6,7].

이러한 기존 연구들의 문제점들을 해결하기 위하여, 본 연구에서는 다음과 같은 관점으로 문제를 해결하였다.

첫째, 척도들간에 가중치를 부여하여 척도들의 측정치가 객관적으로 판단될 수 있도록 한다. 즉, 해당 척도들의 측정값을 모두 동일한 기준으로 보기에는 객관적으로 판단하기 어려우므로 품질 척도들에 적절한 가중치를 부여할 수 있는 방법이 사용되어야 한다[8].

둘째, 소프트웨어 품질에 영향을 미치는 요인들을 모두 고려하여 소프트웨어 품질을 측정할 수 있는 척도를 산출한다. 한 가지 또는 두 가지 요인들만을 대상으로 산출된 척도들을 모두 종합하여 여러 가지 요인들을 다루는 척도를 산출한다.

셋째, 기존에 사용되었던 품질 척도들 중 비율척도 기준에 따르는 척도들을 추출·사용하여 해당 모듈이 다른 모듈에 비해서 정량적으로 얼마만큼 품질이 우수한가를 나타낸다. "프로그램 A의 품질이 프로그램 B의 품질보다 좋다"라고 말하는 것 보다 "프로그램 A의 품질이 프로그램 B의 품질보다 2배 우수하다"라고 말할 수 있으면 사용자나 개발자는 프로그램 유지관리면에서 객관적으로 프로그램의 품질을 판단할 수 있는 지침서가 될 수 있다.

넷째, 척도들의 측정치의 범위가 모두 제 각각이므로 이를 통일화시키는 방법 즉, 정규화 과정이 필요하다. 해당 척도들의 특성에 따라 측정된 결과값도 모두 의미를 가지고 있다. 그러나 척도들의 특성을 모두 살려 품질을 측정하기에는 어려움이 있다 즉, 척도 A의 측정 범위가 0~1이며, 척도 B의 측정 범위가 1000이상이라고 한다면, 측정 결과치에 대한 기준을 마련하기가 어렵다. 따라서 척도들의 측정 결과값을 정규화시킬 수 있는 방법이 고려되어야 한다.

이러한 네 가지 방향으로 연구 초점을 맞춰 기존 척도들이 갖는 문제점을 해결하여 본 논문에서 제시하고자 하는 소프트웨어 환경에 맞는 품질측정함수 모델을 산출한다. 산출된 품질측정함수 모델은 사용자뿐만 아니라 개발자에게 소프트웨어의 유지, 관리 측면의 정보를 제공하는 발판을 마련한다.

3. 품질평가 함수

3.1 품질평가 함수 모델

본 논문에서 제시하고자 하는 소프트웨어 환경에 맞는 품질측정함수 모델은 다음과 같은 다섯 단계를 거쳐 산출된다.

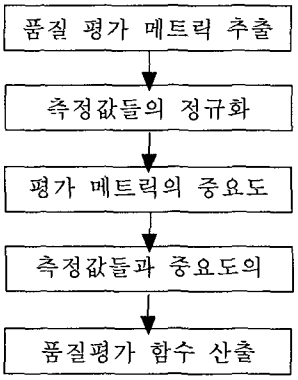

그림 1. 품질측정함수 모델
Fig. 1. Quality measure function

- 메트릭 추출: 메트릭 추출은 소프트웨어 품질 평가를 위한 기존의 척도들 중에서 비율 척도를 전제로 하는 척도들을 추출하는 과정이다.
- 정규화 과정: 정규화 과정은 사용하는 메트릭의 측정값들이 [0,1] 사이에 분포하도록 한다. 정규화 작업을 함으로써 측정값의 의미를 쉽게 파악할 수가 있다. 정규화 작업 시 척도들이 갖는 특성이 퇴색될 염려가 있으므로 검증된 정규 방법을 사용하여 척도들의 측정치를 정규화한다.
- 중요도 산출: 소프트웨어 품질 평가에서 모든 품질 특성(측정 속성)들이 동일한 중요성을 가지지는 않기 때문에 척도들에 대한 적절한 가중치를 부여하기 위해 러프 집합을 이용하여 측정 척도들의 중요도를 산출한다.
- 품질평가: 평가 메트릭들의 정규화된 측정값과 중요도를 Sugeno의 퍼지 적분을 이용하여 종합하여 소프트웨어의 복잡도를 산출한다. Sugeno의 퍼지 적분은 어떤 대상이 여러 항목에 대해서 평가되고 각 평가 항목의 중요도에 차이가 있을 때 이들 평가치를 종합하는데 유용한 방법이다[5].

· 평가 함수 산출: Sugeno의 퍼지적분을 이용하여 산출된 복잡도와 각 메트릭의 측정치들 간의 함수적인 관련성 유무를 측정하기 위해서 회귀분석 방법을 사용한다. 즉, 산출된 복잡도와 각 메트릭의 측정치 사이에 어떤 함수 관계가 생성되는지를 회귀분석 방법을 통해서 살펴볼 수 있다. y=f(x)라는 함수 식이 생성되면, 해당 모듈들의 메트릭 측정치를 함수 식의 x에 대입하면 해당 모듈의 복잡도 즉, 품질 y가 수치로 산출된다.

3.2 비율척도로서 쓰이는 메트릭들

품질함수를 구하기 위한 첫 번째 단계로서, 소프트웨어의 품질을 종합하기 위한 연산으로 비율척도를 다룬다[19]. 본 논문에서는 소프트웨어 복잡도 척도로 쓰이는 메트릭들을 ZUSE[19]가 증명한 정리에 의해서 비율 척도로 구분하였다. 즉, 서로 독립적인 경로의 수를 계산함으로써 프로그램 제어 흐름의 논리적인 복잡도를 측정하는 방법인 Mccabe의 Mcc-V2, 프로그램 길이 측정으로 많이 알려진 LOC(Lines of Code)의 변형 형태인 LOC″ [1,9], 프로그램 길이에 기반을 둔 Halstead의 Length N과 Vocabulary n, 프로그램내에 의사결정논리(decision-making logic)가 얼마나 많이 있는지를 측정하는 Gilb의 ALC(Measure Absolute Logical Complexity: ALC), 제어흐름 복잡도인 Oviedo의 CF, 경로(Path)에 대한 복잡도를 구하는 Schneidewind의 PATH 등을 품질 척도로써 사용하였다.

3.3 정규화

해당 모듈을 대상으로 척도들을 적용하였을 때 측정된 결과 값의 범위가 모두 다르므로, 이를 통일화할 수 있는 정규화 과정이 필요하다[17]. 정규화 작업을 하므로써 측정값의 의미를 쉽게 파악할 수가 있다. 정규화 작업시 척도들이 갖는 특성이 퇴색될 염려가 있으므로 검증된 정규 방법을 사용하여 척도들의 측정치를 정규화한다. 현재 정규화 방법으로 쓰이는 방법으로 퍼지 소속함수, 역수, Sigmoid 함수를 이용하는 방법들이 있다. 본 논문에서는 정규 방법으로 측정치의 지수값에 역수를 취하는 Sigmoid 함수를 사용한다. 그 결과 모든 결과치가 0~1 사이에 포함되도록 한다. 즉, 식(1)과 같은 함수를 사용하여 정규화시킨다.

$$f(x, params) = 1./(1+EXP(-A*(X-C))) \quad (1)$$

f(x, params)는 측정치 x의 값에 의해서 계산된다. params는 두 개의 인자로 f(x, params)의 모양과 위치를 결정해준다.

3.4 메트릭간의 중요도

정규화된 측정치들을 모두 동일하게 판단한다면, 척도들간의 상대적인 가치는 무의미하게 된다. 따라서 각 척도들에 대한 상대적인 가치를 부여하도록 적절한 가중치를 부여하기 위해 러프 논리를 이용하여 측정 척도들의 중요도를 산출한다. 러프 집합의 지식 표현시스템은 각 측정 속성들이 동일한 강도(strength)를 갖는지, 또는 각 측정 속성들이 분류 능력(classification power) 측면에서 얼마나 다른가를 쉽게 산출한다. 러프 집합에서는 속성들의 중요도를 발견하기 위해서 주어진 지식 표현 시스템에서 속성을 삭제하고, 이 속성이 없이 분류들이 어떻게 변화하는 가를 조사한다. 특정 속성의 제거가 분류를 많이 변화시킨다면, 그 속성은 지식 표현 시스템에서 중요도가 높은 속성이 된다[16]. 속성의 상대적 중요도는 전체 속성의 중요도에서 특정 속성의 중요도가 점유하는 비율로 표현될 수 있으며, 식(2)와 같다.

$$RS_i = \frac{S_i}{\sum_{j=1}^{n} S_j}, \ (i = 1, ..., n) \quad (2)$$

RS_i: 속성 i의 상대적 중요도

S_i, S_j: 속성 i, j의 중요도

3.5 Sugeno의 퍼지적분

본 모델에서는 각 측정 속성들이 품질 측정에 미치는 영향의 정도를 결합하여 모듈의 품질을 측정하기 위해서 Sugeno의 퍼지 적분을 이용한다. 집합 X가 유한 집합이고, P(X)는 X의 멱집합이고, 함수 $h(x)$는 평가항목 $x(x \in X)$에 대한 평가치이고, E∈P(X)에 대해 정의되는 퍼지척도 g(E)는 전체적인 평가에 대한 평가항목 E의 평가치에 기여하는 정도라고 할 때, $x_i \in X(i=1,2, \cdots,n)$에 대해서 $h(x_i) \le h(x_{i+1})$ 이고, $E_i = \{x_k \mid k = i, i+1, \cdots, n\}$ 이면, 평가함수 h의 상대적 중요도 함수 g에 대한 임의의 모듈 C의 품질은 식(3)에 의해 구해질 수 있다.

$$\begin{aligned} Qual_C &= \int_X h(x) \circ g(\cdot) \\ &= Max_{i=1}^{n} \, Min[h(x_i), \ g(E_i)] \end{aligned} \quad (3)$$

3.6 품질평가함수 산출

척도들을 종합하여 산출된 복잡도와 각 메트릭의 측정치들간에 어떤 함수적인 관련성 유무를 판단하여야 한다. 즉, 각 메트릭의 측정치들과 소프트웨어 복잡도간에 함수적인 연관성이 존재하는가를 살펴보기 위해서 회귀분석방법을 사용한다. 회귀분석 방법은 종속변수(소프트웨어 복잡도)의 변화에 영향을 미치는 몇 개의 변수들을 이용하여 종속변수의 변화를 예측하는 방법으로서 가장 대표적인 종속관계(dependence)에 관한 분석 방법이다. 또한, 종속변수(소프트웨어 복잡도)의 예측뿐만 아니라 가설이나 이론으로 알려진 가설적 함수관계의 타당성을 검정하기 위해서도 이용된다. 또한, 회귀분석 방법은 원칙적으로 등간척도나 비율척도로 측정된 변수이어야 하기 때문에 본 논문에서 채택한 척도와 일치된다.

4. 실험 및 검증

실험을 위하여, 본 연구에서는 제안된 모델을 기능중심 컴포넌트에 적용하여 품질평가함수를 산출하였다. 기능 중심 컴포넌트의 품질평가에는 Zuse가 제시한 비율척도들 즉, 수정된 Lines of Code, McCabe의 MCC-V2, Oviedo의 CF', Halstead의 Vocabulary와 length, Glib의 ALC, Schneidewind의 PATH를 이용하여 측정하였다. 본 실험에서는, C 언어 런타임 라이브러리에서 무작위 추출한 471개의 모듈을 측정한 결과 값들을 대상으로 품질평가 함수를 예측하였다.

4.1 품질평가 함수 산출

측정값들의 다양한 측정범위를 표준화시키기 위해 식(1)를 도입하여 <표 1>과 같이 측정값들을 재설정하였다.

표 1. 측정값의 정규화
table 1. standard of measured value

컴포넌트	LOC″	MCC-V2	CF′	VOC	LEN	ALC	PATH
1	0.406	0.354	0.231	0.214	0.207	0.310	0.261
2	0.403	0.331	0.214	0.182	0.205	0.289	0.260
3	0.413	0.377	0.268	0.249	0.240	0.331	0.264
·	·	·	·	·	·	·	·
469	0.425	0.401	0.310	0.259	0.246	0.354	0.266
470	0.418	0.354	0.259	0.205	0.242	0.310	0.261
471	0.418	0.354	0.259	0.205	0.240	0.310	0.261

각 척도들에 대한 상대적인 가치를 부여하도록 적절한 가중치를 부여하기 위해 표 1의 값에 식(2)를 이용한 측정 척도들의 상대적 중요도는 표 2와 같다.

표 2. 각 척도들의 상대적 중요도
table 2. the relative significance of metrics

척도	상대적 중요도
LOC″	0.114296
MCC-V2	0
CF′	0
VOC	0.371413
LEN	0.514290
ALC	0
PATH	0

<표 1>의 측정값에 <표 2>와 식(3)을 적용한 품질측정값들은 <표 3>과 같다.

표 3, 품질 측정값
table 3. quality measure value

컴포넌트	LOC″	MCC-V2	CF′	VOC	LEN	ALC	PATH	Quality
1	0.406	0.354	0.231	0.214	0.207	0.310	0.261	0.1142
2	0.403	0.331	0.214	0.182	0.205	0.289	0.260	0.1142
3	0.413	0.377	0.268	0.249	0.240	0.331	0.264	0.1247
·	·	·	·	·	·	·	·	·
469	0.425	0.401	0.310	0.259	0.246	0.354	0.266	0.1343
470	0.418	0.354	0.259	0.205	0.242	0.310	0.261	0.1174
471	0.418	0.354	0.259	0.205	0.240	0.310	0.261	0.1156

본 연구에서 궁극적으로 제시하고자 하는 품질평가 함수를 예측하기 위하여 <표 3>을 바탕으로 회귀분석을 수행한 결과 다음과 같은 회귀식을 도출할 수 있다.

$$\text{Quality} = 7.E\text{-}02 - 0.015LOC + 4.E\text{-}03MCC\text{-}V2 + 0.112CF' + 0.42VOC + 0.391LEN - 0.09ALC - 0.033PATH \quad (4)$$

4.2 검증

Kitchenham[12]은 예측 모델을 검증하는 것은 척도를 검증하는 것과는 다르나. 기초적인 통계 기법들은 속성들 간의 관계를 연구하기 위해 사용될 수 있으며, 1)관계의 원인이 된 속성의 변화에 따른 측정 속성에서의 기대된 변화가 관찰되고, 2)관계를 측정하는 방법이 관계 자체를 변화시키지 않고, 3)예측 능력을 평가하기 위해서 교차 검증(cross-validation)을 사용하여 동일한 결과가 나타나면 경험적 실험 관찰에 의한 통계적 검증은 타당하다고 하였다.

본 연구에서는 Kitchenham의 통계적 방법을 기초로 하여, 본 모델에서 제시한 함수가 다른 측정값들로써 품질도를 예측할 수 있음을 보여준다.

4.2.1 품질 함수 검증

본 연구에서 제시한 함수를 검증하기 위해서 통계적 검증방법인 회귀분석을 사용하였다. 회귀식이 얻어지면 그 식이 통계적으로 의미가 있는 것인지 검토할 필요가 있다[2]. 회귀식의 통계적 유의성을 검정하려면 분산분석표가 필요하게 된다. 표 3의 측정값을 바탕으로 분산분석을 실행한 결과 표 4와 같은 분산분석표가 얻어진다.

표 4. 분산분석
table 4. analysis of variance(ANOVA)

Model	Sum of Squares	df	Mean Square	F	Sig.
1 Regression	7.462	7	1.066	2480.780	.000
Residual	.199	463	4.297E-04		
Total	7.661	470			

표 4에서 F=2480.780 >F(1,439,0.05)=3.84이고 F-확률은 0.000으로 유의수준 5%에서 매우 유의하다고 할 수 있으며, 잔차평균제곱(residual mean square)이 0.0004297이므로 관찰값들이 회귀식 주위에 가깝게 밀집되어 있다고 할 수 있다. 또한, 회귀식의 유효성을 평가하기 위한 지표로서 자유도조정필 기여율(수정된 R^2)이 있다. 기여율 값이 1에 가까울수록 회귀식은 잘 들어맞는다고 할 수 있다.

표 5. 회귀모델의 요약
table 5. the summary of regression model

Model	R	R Square	Adjusted R Square	Std. Error of the Estimate	Durbin-Watson
1	.987	.974	.974	2.073E-02	2.046

표 5. 에서 수정된 결정계수가 0.974로 전체분산 중에서 97%를 설명할 수 있다는 것을 보여준다. 이외에 회귀식의 유효성을 평가하기 위한 또 하나의 지표가 잔차의 표준편차(추정값의 표준오차)이다. 잔차란 실제값과 회귀식에 의한 예측치와의 차로서 이 잔차가 전체적으로 작은 회귀식일수록 유효한 회귀식이라고 볼 수 있다.

그림 2는 잔차의 분포가 근사적으로 정규분포를 따른다는 것을 보여주며, 그림 3은 표준화된 잔차들이 정규분포상에 있으므로 예측정도가 높다고 할 수 있으며, 예

측값의 증감과 관계없이 특정한 추세를 보이고 있지 않으므로 등분산성의 가정에 위배된다고 볼 수 없다[3,14]. 분석결과 산출된 회귀식은 5% 유의수준에서 F확률 =0.00<0.01으로 매우 유의하므로 다른 측정치들로부터 품질도를 산출할 수 있다.

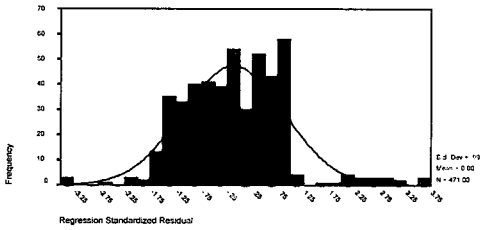

그림 2. 표준화된 잔차들
Fig. 2. standardized residuals

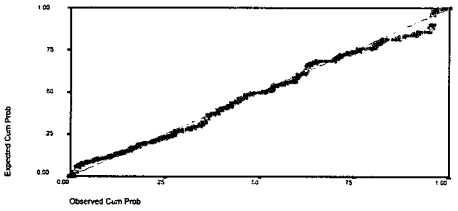

그림 3. 표준화된 잔차들에 대한 기대분포와 관측분포
Fig. 3. the sxpected and observed distribution of the standardized residuals

4.3 실험결과 분석

기존의 품질척도들에 관한 연구들과 비교하였을 때, 산출된 함수는 앞에서 제시한 문제점들을 모두 만족시킴을 알 수 있었다.

첫째, 소프트웨어 품질을 정량적으로 측정할 수 있는 비율척도들을 도입하여 품질을 평가하였다.

둘째, 척도들의 특성을 모두 수용할 수 있는 요소들을 모두 고려·종합하여 측정하였다.

셋째, 척도들의 측정치의 범위를 통일화시켰다. 즉, 각 척도들의 측정값들이 [0,1] 사이에 분포하도록 정규화 시켰다.

넷째, 척도들간에 가중치를 부여하여 척도들의 측정치가 객관적으로 판단될 수 있도록 한다.

또한, 위에서 제시한 산출함수는 프로그램이 갖는 기본적인 요소들, 즉, 프로그램 길이, 연산자 수, 피연산자 수, 결정 노드 수, 경로 길이 등 만을 이용하여 소프트웨어 품질을 정량화할 수 있어 소프트웨어 재사용에 도움을 줄 수 있으리라 기대된다.

5. 결 론

본 연구에서는 기존의 품질측정 척도들이 안고있는 문제점들을 해결하기 위하여 러프논리와 회귀분석을 이용한 소프트웨어 품질측정 함수를 산출하였다. 품질의 정량적 척도에 기반이 되는 여러 요인들을 종합하기 위하여 각 척도들의 중요도를 산출하여 품질을 평가하였다. 또한 이러한 요인들과 품질간의 상관관계를 분석하여 하나의 품질함수를 산출한다. 본 연구에서 제안한 함수는 쉽게 측정치만을 가지고 여러 특성들을 종합한 품질 평가와 다른 소프트웨어들간의 정량적 비교도 할 수 있다. 그러나, 본 연구에서 제시한 함수는 기능중심 소프트웨어 적용할 수 있는 함수 식이므로, 객체지향 중심의 소프트웨어 품질을 정량적으로 평가할 수 있는 함수가 필요하리라 본다.

참 고 문 헌

[1] 김태공·우치수, "프로그램 경로에 기반을 둔 복잡도의 척도," 한국정보과학회 논문지, 20(1):34-42, 1993.
[2] 노형진, 「SPSS에 의한 알기쉬운 다변량분석」, 서울: 형설출판사, 1993.
[3] 박성형, 「회귀분석」, 서울: 민영사, 1991.
[4] 송영재, 「C 언어로 구현한 소프트웨어 엔지니어링」, 홍릉과학출판사, pp. 513-521, 1992.
[5] 이광형·오길록, 「퍼지이론 및 응용 I권 이론」, 서울:홍릉과학출판사, 1991, pp.1.1-1.40, 9.1-9.36.
[6] 김혜경 등, "러프와 퍼지 집합을 이용한 재사용 컴포넌트의 재사용도 측정," *한국정보처리학회 논문지*, vol. 6, no. 9, 1999.
[7] 박병권 등, "러프집합과 퍼지집합에 기반한 기능중심 컴포넌트의 재사용도 측정," *한국 퍼지 및 지능시스템학회 논문지*, vol. 9, no. 4, pp. 375-383, 1999.
[8] Balog, A. Trifu, R. Raduta, A.. "Quality Evaluation of Public Administration Software Products: A Practical Study," The 5th European Conference on Software Quality, 1996.
[9] Halstead, M.H., "Elements of Software Science," New York: Elsevier North-Holland, 1977.
[10] Hansem, H., "Measurement of Program Complexity by the Pair(Cyclomatic Number, Operator Count)", ACM SIGPLAN Notics, pp. 146-152, Mar., 1978.
[11] Harrison, W., Magei, K., Kluczny,R., and Dekock, A., "Applying Software Complexity Metrics to program Maintenance," IEEE computer, Sept., 1982.
[12] Barbara Kitchenham, et.al., "Towards a Framework for Software Measurement Validation," IEEE Transactions On Software Engineering, vol. 21, no. 12, pp. 929-944, DEC., 1995.
[13] McCabe T.. "A Complexity Measure.", IEEE Transaction on Software Engineering 2:308-320, 1976.
[14] Neter, J. Wasserman, W.Kutner M.H.. Applied Statistical Models-3rd ed, Boston:IRWIN, 1990.
[15] Ovido,E., "Control Flow, Data Flow and Program Complexity," Proceeding COMPSAC 80, pp. 146-152, 1980.
[16] Pawlak Z.. 「Rough Sets-Theoretical Aspects of Reasoning about Data」, London: Kluwer Academic Publishers, 1991.
[17] Pu-Lin Yeh and Jin-Cherng Lin, "Toward Precise Measurements Using Software Normalization," Proceddings of the 1999 International Conference on Software Engineering, p. 736-737, LosAngeles, CA USA, 1999.

[18] Shen, V. Y., Conte, S. D., and Dunsmore, H. E., "Software Science Revisited: A Critical Analysis of the Theory and its Empirical Support," IEEE transactions on Software Engineering, SE-9(2):155-165,1983.
[19] Zuse.H., Bollmann, P., "Using Measurement Theory to Describe the Properties and Scales of Static Software Complexity Metrics," Sigplan Notices, vol. 24, no. 8, pp. 23-33, Aug., 1989.

저 자 소 개

김혜경(Hye-Kyoung Kim)
1993년 : 조선대학교 전산통계학과 졸업
1998년 : 조선대학교 전자계산학과(이학석사)
1999년~현재 : 조선대학교 전자계산학과 박사과정
2001.8~현재 : 안성여자기능대학 인터넷미디어과 전임강사

관심분야 : 소프트웨어 공학, 객체지향 시스템, 러프 집합
Phone : 031-650-7253
Fax : 031-651-4314
E-mail : kimhk@ans.ac.kr

최완규(Wan-Kyoo Choi)
1997년 : 조선대학교 전자계산학과(이학석사)
2000년 : 조선대학교 전자계산학과(이학박사)
2000년~현재 : 광주대학교 컴퓨터전자통신공학부 전임강사

관심분야 : 소프트웨어 공학, 프로그래밍 언어, 객체지향 시스템, 러프 집합
E-mail : wkchoi@kwangju.ac.kr

이성주(Sung-Joo Lee)
1970년 : 한남대학교 물리학과 졸업
1992년 : 광운대학교 전자계산학과(이학석사)
1998년 : 대구 가톨릭대학교 전자계산학과 (이학박사)
1988년~1990년 : 조선대학교 전자계산소 소장
1995년~1997년 : 조선대학교 정보과학대학장
1981년~현재 : 조선대학교 컴퓨터공학부 교수

관심분야 : 소프트웨어 공학, 프로그래밍 언어, 객체지향 시스템, 러프 집합
E-mail : sjlee@mail.chosun.ac.kr