Proceedings of KFIS 2001 Spring Conference, 2001. 5.

비율척도를 이용한 프로그램 모듈의 품질평가에 관한 연구

A Study on the quality estimate of the program module using the ratio metric

김 혜 경 · 이 성 주
조선대학교 전자계산학과
Hye-Kyoung Kim · Sung-Joo Lee
Department of computer science, chosun univ.
E-mail: hkkim@stmail.chosun.ac.kr

ABSTRACT

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.

Keyword: quality, rough set, complexity

I. 서론

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[2].

그러나 지금까지의 품질평가 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[3,4]. 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 제

시하고자 하는 품질평가모델에 대해 다루었다. 4장에서는 실제로 사용되고 있는 소스 프로그램을 대상으로 하여 실험·분석하였다. 마지막으로 5장에서는 품질평가 함수의 기대효과와 앞으로 연구할 방향 등을 제시하였다.

Ⅱ. 관련 연구

A 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[6,9,13].

둘째, 혼합적 척도들은 각 척도에 맞는 요소들만을 기준으로 하여 측정하였다[1].

셋째, 대부분의 척도들은 비율척도보다는 순위척도 또는 명목 척도를 전제로 하고 있다. 명목 척도는 매우 단순하고, 순위 척도는 정량적으로 어느 정도 어려운지를 나타내주지 못하며, 구간 척도는 단위(unit)의 정의가 필요하다는 단점을 안고 있다. 비율 척도는 융통성(flexible)이 있으며 실제적(preactical)이기 때문에 소프트웨어 복잡도 측정에 가장 적합하고 좋은 척도라고 할 수 있다.

기존 품질 측정에 관련된 연구들에서는 명목 척도 또는 순위척도에 적합한 척도들을 제시하였다[3,4].

이러한 문제점들에 연구 초점을 맞춰 본 논문에서는 제시하고자 하는 소프트웨어 환경에 맞는 품질측정함수 모델을 산출한다.

Ⅲ. 품질평가 함수

A. 품질평가 함수 모델

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

· 메트릭 추출: 메트릭 추출은 소프트웨어 품질 평가를 위한 기존의 척도들 중에서 비율척도를 전제로 하는 척도들을 추출하는 과정으로, 소프트웨어 복잡도 척도로 쓰이는 메트릭들을 ZUSE[14]가 증명한 정리에 의해서 비율 척도로 구분하였다.

· 정규화 과정: 정규화 과정은 사용하는 메트릭의 측정값들이 [0,1] 사이에 분포하도록 한다. 정규화 작업을 함으로써 측정값의 의미를 쉽게 파악할 수가 있다. 본 논문에서는 정규방법으로 측정치의 지수값에 역수를 취하는 Sigmoid 함수를 사용한다. 그 결과 모든 결과치가 0~1 사이에 포함되도록 한다.

· 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. 식(1)은 다음과 같다.

$$RS_i = \frac{S_i}{\sum_{j=1}^{n} S_j}, \ (i = 1, \ldots, n) \qquad (1)$$

RS_i: 속성 i의 상대적 중요도

S_i, S_j: 속성 i, j의 중요도

· 품질평가: 평가 메트릭들의 정규화된 측정값과 중요도를 Sugeno의 퍼지 적분을 이용하여 종합하여 소프트웨어의 품질도를 산출한다.

Ⅳ. 실험 및 평가

실험을 위하여, 본 연구에서는 제안된 모델을 기능중심 컴포넌트에 적용하여 품질평가 하였다. 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 결과값들을 대상으로 품질평가 하였다.

A. 품질 측정

측정값들의 다양한 측정범위를 표준화시키기 위해 Sigmoid 함수를 도입하여 <표 1>과 같이 측정값들을 재설정하였다.

<표 1 > 측정값의 정규화

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
.	.	.	.	.	.	.	.
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 1>의 값에 식(1)을 이용한 측정 척도들의 상대적 중요도는 <표 2>와 같다.

<표 2 > 각 척도들의 상대적 중요도

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 Sugeno의 퍼지적분을 적용한 품질측정값들은 <표 3>과 같다.

<표 3 > 품질 측정값

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
.	.	.	.	.	.	.	.	.
.	.	.	.	.	.	.	.	.
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

B. 실험결과 분석

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

첫째, 소프트웨어 품질을 정량적으로 측정할 수 있는 비율척도들을 도입하여 품질을 평가하였다. 소프트웨어간의 품질을 비교하는 데에는 정성적 척도보다는 정량적 척도가 유용하다.

둘째, 척도들의 특성을 모두 수용할 수 있다. 크기, 제어구조, 데이터구조, 또는 이들을 혼합한 척도들을 기반으로 하는 척도들은 소프트웨어 품질을 완전하게 측정하지 못한다. 각 척도에 맞는 요소들을 기준으로 측정하기 때문이다. 따라서 품질에 영향을 주는 요소들을 모두 고려·종합하여 측정하였다.

셋째, 척도들의 측정치의 범위를 통일화시켰다. 즉, 각 척도들의 측정값들이 [0,1] 사이에 분포하도록 정규화시켰다. 정규화에는 이미 여러 연구에서 검증된 Sigmoid 함수를 사용하였다.

넷째, 척도들간에 가중치를 부여하여 척도들의 측정치가 객관적으로 판단될 수 있도록 한다. 척도들의 특성이 각기 다르므로, 이를 객관적으로 평가할 수 있는 척도들간의 가중치가 필요하다. 이를 위해서 본 연구에서는 러프논리를 이용한 상대적 중요도를 산출하였다.

V. 결론

본 연구에서는 기존의 품질측정 척도들이 안고있는 문제점들을 해결하기 위하여 러프논리와 비율척도를 이용하여 소프트웨어 품질을 측정하였다.

즉, 품질의 정량적 척도에 기반이 되는 여러 요인들을 종합하기 위하여 각 척도들의 중요도를 산출하여 품질을 평가하였다.

본 연구에서 제안한 방법은 쉽게 측정치만을 가지고 여러 특성들을 종합한 품질 평가와 다른 소프트웨어들간의 정량적 비교도 할 수 있다.

VI. 참고 문헌

[1]김태공·우치수,"프로그램 경로에 기반을 둔 복잡도의 척도," 한국정보과학회 논문지, 20(1):34-42, 1993.

[2]송영재, 「C 언어로 구현한 소프트웨어 엔지니어링」, 홍릉과학출판사, pp.513-521, 1992.

[3]김혜경 등, "러프와 퍼지 집합을 이용한 재사용 컴포넌트의 재사용도 측정," 한국정보처리학회 논문지, Vol.6, No.9, 1999.

[4]박병권 등, "러프집합과 퍼지집합에 기반한 기능중심 컴포넌트의 재사용도 측정," 한국 퍼지 및 지능시스템학회 논문지, Vol.9, No.4, pp.375-383, 1999.

[5]Balog, A. Trifu, R. Raduta, A.. "Quality Evaluation of Public Administration Software Products: A Practical Study," The 5th European Conference on Software Quality, 1996.

[6]Halstead, M.H., "Elements of Software Science," New York: Elsevier North-Holland, 1977.

[7]Hansem, H.,"Measurement of Program Complexity by the Pair(Cyclomatic Number, Operator Count)", ACM SIGPLAN Notics, pp.146-152, Mar., 1978.

[8]Harrison,W.,Magei,K.,Kluczny,R., and Dekock,A., "Applying Software Complexity Metrics to program Maintenance," IEEE computer, Sept.,1982.

[9]McCabe T.. "A Complexity Measure.", IEEE Transaction on Software Engineering 2:308-320, 1976.

[10]Ovido,E., "Control Flow, Data Flow and Program Complexity," Proceeding COMPSAC 80, pp.146-152, 1980.

[11]Pawlak Z.. 「Rough Sets-Theoretical Aspects of Reasoning about Data」, London: Kluwer Academic Publishers, 1991.

[12]Pu-Lin Y도 and Jin-Cherng Lin, "Toward Precise Measurements Using Software Normalization," Proceddings of the 1999 International Conference on Software Engineering, p.736-737, LosAngeles, CAUSA,1999.

[13]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.

[14]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.