Rough set을 이용한 재사용성 평가 모델

최경욱*, 이성주**, 정환묵*
*대구효성가톨릭대학교 전자정보 공학부, **조선대학교 전자계산학과

Reusability Decision Model using Rough Set

Kyoung Oak Choi*, Sung Joo Lee**, Hwan Mook Chung*
*Faculty of Electronics & info. Engineering Catholic Univ. of Taeguhyosung
**Chosun University Dept's of Computer Science

Abstract

소프트웨어 재사용은 새로운 소프트웨어 개발에 소용되는 시간과 비용을 현저히 감소시켜 소프트웨어 개발환경과 생산성을 향상시키는 방법으로, 소프트웨어 위기를 해결하기 위한 중요한 방법이다. 그러나 소프트웨어 부품을 위한 형식 명세서(formal specification)의 부족, 소프트웨어 재사용에 대한 부정적 심리적인 효과 등의 이유 때문에 현실적으로 재사용이 잘 이루어지고있지 않다.

이러한 문제들을 해결하기 위해서는 부품의 품질 보증에 관한 연구가 소프트웨어 재사용에 관한 연구 분야에서 최우선적으로 이루어져야 하지만, 기존의 연구들은 일반적으로 설정된 재사용 품질 기준을 표준으로 하였으므로, 사용자의 요구가 복잡하고, 다양화되면서 소프트웨어의 크기, 알고리즘과 구조의 복잡도는 증가있는 변화하는 환경에 능동적으로 대처하지 못하고 있다.

그러므로 본 연구에서는 새로운 부품의 삽입과 기존 부품들의 삭제, 분류 기준의 변경 등의 환경 변화에 능동적으로 대처할 수 있는 적응성이 있는 재사용성 결정 모델을 제안한다. 이 모델은 적응성 있는 재사용 결정 알고리즘을 찾기 위해서 데이터에 숨겨진 패턴들을 발견하는 효율적인 알고리즘을 제공하는 Rough set 이론을 이용한다.

Ⅰ. 서론

70년대까지는 하드웨어의 비약적인 발전에 비하여 소프트웨어 공학 발전이 상대적으로 늦어짐에 따라 이에 대한 대책이 강구되기 시작하였다. 통계에 의하면 개발된 소프트웨어의 20-30% 만이 특정 프로그램에 한정되고, 70-80%는 여러 프로그램에서 공통적으로 이용할 수 있는 기능인 것으로 나타났다[3].

따라서, 신뢰성이 높은 고 품질의 소프트웨어를 생산하기 위해서는 기존의 개발 방법론으로는 한계가 있기 때문에 이를 해결하기 위한 하나의 방법으로 소프트웨어 재사용에 관한 연구가 활발히 진행 중이다.

소프트웨어 재사용은 신뢰성이 확보된 기존의 소프트웨어를 재사용 하므로 새로운 소프트웨어 개발에 소요되는 시간과 비용을 현저히 감소시켜준다. 그러므로 소프트웨어 위기를 극복하기 위해서는 개발된 프로그램을 재사용하는 연구가 절실히 요구된다. 그러나, 여러 가지 이유 때문에 소프트웨어 개1.발 과정에 널리 이용되지 않고 있다[5].

이러한 문제들을 해결하기 위해서는 부품의 품질 보증에 관한 연구가 소프트웨어 재사용 연구 분야에서 최우선적으로 이루어져야 한다. 부품의 품질 보증은 재사용 가능한 부품을 식별하여 사용자에게 제공함으로써, 부품 사용자들의 불신을 해소시키고 재사용율을 제고하여 소프트웨어의 생산성과 신뢰성을 향상시킨다.

그러나, 지금까지의 연구들은 사용자의 요구가 복잡하고, 다양화 되면서 나타나는 소프트웨어의 크기, 알고리즘과 구조의 복잡도 증가 등의 환경변화에 능동적으로 대처하지 못하고있다.

따라서, 본 연구에서는 이런 문제점을 기반으로하여 실제로 재사용되고 있는 부품들과 산업현장에서 그 타당성을 인정받은 정량적인 척도들과 분류 기준을 이용하여, 새로운 부품의 삽입과 기존 부품들의 삭제, 분류 기준의 변경 등의 환경 변화에 능동적으로 대처할 수 있는 적

응성이 있는 재사용성 결정 모델을 제안한다. 이 모델은 적응성 있는 재사용 결정 알고리즘을 찾기위해서 다음과 같은 장점을 갖는 Rough set 이론을 이용한다.

본 연구는 다음과 같이 구성하였다. 2장에서는 소프트웨어 재사용 개념의 고찰, 3장은 Rough set 이론 및 Rough logic를 간략하게 설명하였다. 4장에서는 본 연구에서 제안한 재사용성 결정 모델의 구조 및 구성 요소들을 제시하고, 5장에서는 본 모델을 이용하여 최소 재사용성 결정 알고리즘의 생성 과정을 구현하였고, 결론에서는 본 연구를 통하여 얻은 결론에 대해서 언급한다.

II. 소프트웨어 재사용

소프트웨어 재사용은 기능, 성능 및 품질을 인정받은 이미 개발된 소프트웨어 부품의 전체 혹은 일부분을 다시 사용하므로써, 새롭게 개발되는 소프트웨어의 품질과 생산성 및 신뢰성을 높이고, 개발 기간과 비용을 감소시켜 소프트웨어 위기를 극복하기 위한 방안 중의 하나이다.

그러나, 소프트웨어 부품을 위한 형식 명세서(Formal Specification)의 부족, 소프트웨어 부품의 정확성 증명의 어려움, 재사용 가능한 소프트웨어 부품들의 성능에 관한 문제, 소프트웨어 재사용에 대한 부정적 심리적인 효과 등의 이유 때문에 소프트웨어 개발 과정에 널리 이용되지 않고 있다[5].

그러므로, 실제로 소프트웨어 재사용에 관한 연구에서 최우선적으로 접근해야할 부분이 부품의 품질 측정에 대한 부분이다. 이것은 부품의 재사용성(Reusability)을 평가하여 재사용 가능한 부품을 식별하는 작업이다. 실제로 부품의 품질 보증은 부품 사용자들의 불신을 해소시키고 재사용율을 제고하여 소프트웨어의 생산성과 신뢰성을 향상시킨다.

재사용성 측정은 주로 모듈성을 정량적으로 측정하려는 방향으로 진행되어 왔다[2,14]. 이 연구들은 먼저 평가 기준들을 설정한 후 정량적인 측정 방법을 도입하여 평가 기준들을 측정하여 재사용성을 평가하며, 대표적인 연구로는 CARE시스템을 들 수 있다.

2.1 CARE 시스템

CARE 시스템은 재사용성을 정량적으로 평가하는 방법이다[2]. 이 시스템은 부품들을 재사용하기 위하여 재사용에 영향을 미치는 품질 인자로서 유용성(Function Usefulness), 비용(Cost), 품질(Quality)로 정의하였고, 이들은 Halstead의 부품의 크기(Volume), McCabe의 순환수(Cyclomatic number)로 정의한 모듈의 복잡도(Complexity)가 추정치 와 얼마나 근접하느냐에 따라 모듈의 정규성(Regularity), 모듈의 호출 횟수에 따른 재사용 빈도 수(Reuse frequency)를 이용하여 측정하였다.

Table 1 Extremes for ranges of acceptable values in CASE system

Measure	Minimum	Maximum
Volume	2.0	10.0
Complexity	5.0	15.0
Regularity	0.70	1.30
Reuse Frequency	0.30	-

CARE 시스템은 Case study를 통해서 ANSI C의 187,000 라인의 9개의 시스템을 분석하여 Table 1과 같은 측정값의 수용가능 범위를 결정하였다.

그러나 네 가지 척도로써 품질 인자를 정확히 측정하기 어렵다. 또한 모듈의 재사용에 필요한 다양한 품질 인자들을 제시하면서도 네 가지의 척도만을 사용하여 측정할 수밖에 없는 요인이 존재한다.

2.2 측정 방법들의 문제점

기존의 평가 시스템들은 일반적으로 설정된 재사용 품질 기준을 표준으로 하였으므로, 특정 소프트웨어를 개발하기 위하여 요구되는 부품들이 설정된 품질기준을 만족하지 못하여 재사용이 불가능한 부품으로 분류되는 경우가 있다. 산업 현장에서의 많은 연구들은 McCabe의 순환수(Cyclomatic number)가 20 이상이면 프로그램이 매우 복잡하거나 구조가 필요 이상으로 복잡한 프로그램이므로 프로그램을 분할하는

것이 바람직하다고 주장한다[7]. 그러나, 순환수 값이 20 이상인 프로그램이 구조화가 잘 됐을 경우, 유지 보수가 용이하다는 양면성이 존재하기도 한다[1].

소프트웨어 위기의 근본적인 원인 중 하나는 소프트웨어 복잡성의 증가로 프로그램의 크기는 물론 요구되는 알고리즘 등 소프트웨어 복잡도가 점차 높아지고 있다[13]. 소프트웨어의 크기, 알고리즘과 구조의 복잡도 증가는 기존의 재사용 평가시스템들이 이미 설정한 기준의 변경을 요구하게 된다. 그러나 기준의 변경은 재사용 관리 시스템 전체의 변경을 요구할 수 있으므로, 환경변화에 능동적으로 대처하지 못하게 하는 이유가 된다.

Ⅲ. Rough Sets

불완전한 지식(Imperfect Knowledge)의 문제에 대한 많은 접근 방법 중 가장 널리 알려진 방법론은 Zadeh에 의해 제안된 Fuzzy 집합 이론이다.

이 문제에 대한 또 다른 시도로써, 1982년 Pawlak에 의해 제안된 Rough set 개념은 모호성(vagueness)과 불확실성(uncertainty)에 대한 새로운 수학적 접근 방법이다[8].

Rough set에서의 지식은 전체 집합 U라 불리는 실질적인 또는 추상적인 세계의 특정 부분들과 관련된 다양한 분류 패턴과 관계가 있으며, 지식은 관심 있는 영역에 대한 다양한 분류 패턴들의 모임(family)으로 이루어진다.

3.1 Rough logic

Rough logic[9,10]은 Rough set이론에 근거하여 분류(classification)로서 간주되는 지식을 추론할 수 있게 한다. 객체들의 분류에 관한 지식은 정보 시스템(Information System)으로 표현될 수 있으며, 정보 시스템의 각 항목은 객체에 관한 내용(Statement)으로 보여질 수 있다. Rough logic은 의사결정 논리 언어(Decision logic)라는 논리적 도구로 정보 시스템을 다룰 수 있게 한다.

의사결정 논리 언어는 특정 지식표현 시스템에 포함된 지식을 표현한다.

의사결정 언어에서의 의사결정 규칙는 $\Phi \rightarrow \Psi$로 표현되며, 의사결정 규칙들의 유한 집합이 기본 의사결정 알고리즘(Basic Decision Algorithm)이다.

수용 가능하고 일치하는 알고리즘이 있을 때, Rough logic은 전체 시스템에서 불필요한 속성들을 제거, 동일한 의사결정류들과 관련된 불필요한 의사결정 규칙들의 제거를 통하여 최소 의사결정 규칙의 집합을 찾는다. Rough logic에서의 지식의 감축는 지식의 Reduct와 Core의 개념을 논리적 용어(logical term)들의 정의를 통하여 이루어진다.

Ⅳ. 재사용성 결정 모델

본 연구에서 제안한 재사용성 결정 모델은 라이브러리 관리 시스템의 서브 시스템으로서 부품의 등록, 삭제, 갱신 등의 작업이 이루어질 때 능동적으로 재사용 결정 알고리즘을 생성하는데 있으며, 제안한 재사용성 결정 모델의 라이브러리 관리 시스템 기능은 Fig. 1과 같다.

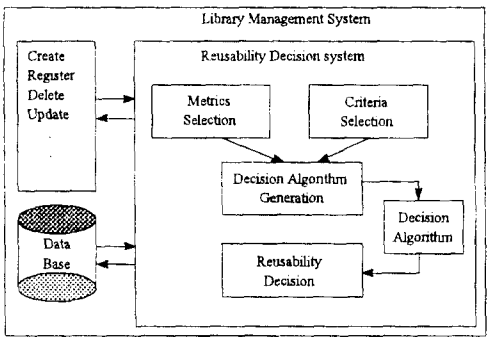

Fig. 1 Library Management System

본 연구에서 제안한 재사용 결정 모델은 측정 메트릭스 선택(Metric Selection), 결정 기준 설정(Criteria Selection), 결정 알고리즘 생성(Decision Algorithm Generation), 재사용성 결정(Reusability Decision)으로 구성된다.

4.1 측정 메트릭스

기능 중심으로 구성된 부품의 재사용성 평가는 규모(Size)와 난이도(Difficulty)의 메트릭스를 수집한 정적 분석기에 의해 가장 쉽게 측정될 수 있다[1]. 따라서 제안된 모델에서는 기본적인 측정 메트릭스로 LOC(Lines of Code), McCabe의 Cyclomatic number, Halstead의 Volume, Difficulty, Effort를 조합하여 재사용성을 평가한다.

4.2 결정 기준의 선택

본 연구에서는 기존의 여러 연구들을 수용하여 Table 2와 같이 초기 값을 설정하였다[1, 2, 6, 11, 12].

Table 2. Limit values of measures by In this paper

Domain / Measure	VS	LS	LC	VC
LOC	<50	50~100	100~150	>150
Cyclomatic Number	<10	10~20	20~50	>50
Volume	<2000	2000~5000	5000~10000	>10000
Difficulty	<10	10~30	30~100	>100
Effort	<50000	50000~100000	100000~300000	>300000

주) VS: Very simple LS: Little Simple
LC: Little Complex VC: Very Complex

이러한 경계들이 산업 현장에서의 연구와 경험을 통해서 증명되어 왔지만 이 경계들은 다소 임의적이며, 실제로 각 현장에서의 요구들은 변할 수 있다[1].

4.3 결정 알고리즘 생성

재사용성 결정은 실제로 모듈을 분류하는 기준에 따라서 모듈의 재사용성을 분류하는 것이며, 부품 분류에 관한 정보들은 부품 재사용에 관한 지식이 된다.

본 연구에서 제안한 결정 알고리즘 생성기는 Rough logic을 이용하여 최소 평가 규칙을 찾는 논리 구조는 다음과 같으며, 알고리즘 생성기의 구조는 Fig. 2와 같이 설계하였다.

[Step 1] 속성 도메인의 정의
[Step 2] 속성값 도메인의 정의
[Step 3] 지식 표현
[Step 4] 지식의 감축
  Step 4A 중복된 규칙의 제거
  Step 4B 알고리즘의 감축
  Step 4C 의사결정 규칙의 감축
[Step 5] 알고리즘의 최소화

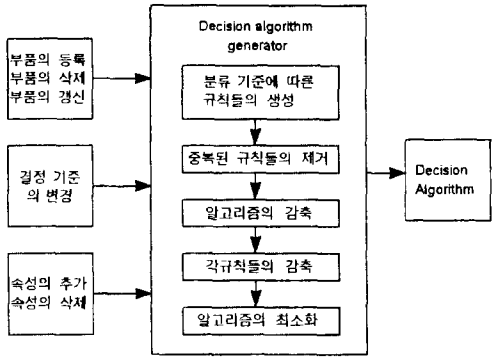

Fig. 2 Architecture of Decision Algorithm generator

실제로 각 현장에서의 요구나 외부 환경의 변화에 따른 기준에 합당하지 않은 부품들에 대한 추가나, 기존 부품의 삭제가 이루어질 수 있다. 이 경우에 재사용성 결정 모델은 추가 또는 삭제된 부품들의 정보를 이용하여 재사용성 결정 알고리즘을 재구성한다.

V. 실험

기존의 재사용 시스템에서 재사용이 이루어지고 있는 부품들에 대해서 정의된 속성 도메인에 대한 측정값을 구하여 이 값들을 속성값 도메인으로 분류하여 정보 표(Information Table)로 구성한다.

본 모델의 실험은 C 컴파일러의 175개에 해당하는 Run time 라이브러리를 대상으로 하며, 175개의 모듈을 정의된 각 메트릭스들에 의해

측정된 측정값을 통해 구성한 규칙들 중 중복된 157개의 규칙을 제거하여 획득된 18개의 규칙으로 구성된 정보 표는 Table 3과 같다.

Table 3. Information table removing duplicate rules

attribute / rules	a	b	c	d	e	f
r1(97)	0	0	0	0	0	1
r2(52)	0	0	0	1	0	1
r3(6)	0	0	0	2	0	1
r4(2)	0	1	0	1	0	1
r5(3)	1	1	0	2	0	1
r6(5)	1	1	0	2	1	1
r7(2)	1	1	0	1	0	1
r8(1)	2	1	1	2	2	1
r9(2)	0	1	0	2	0	1
r10(1)	1	0	0	0	0	1
r11(1)	2	2	0	2	2	1
r12(1)	1	0	0	2	0	1
r13(1)	1	1	1	2	2	1
r14(1)	1	0	0	2	2	1

{a, b, c, d, e, f}는 각 측정 메트릭스 {Lines of Code, Cyclomatic number, Volume, Difficulty, Effort, Reusability}에 대응하고, 각 항목들의 값인 {0, 1, 2}는 각 측정값에 대한 속성값 도메인 {VS, LS, LC}에 대응한다.

두 번째 단계로 전체 시스템에서 불필요한 속성을 식 $f(A)=\prod_{(x,y)\in U^2}\{\sum\delta(x,y):(x,y)\in U^2 \text{ and } \delta(X,Y)\neq 0\}$ 을 이용하여 제거하고, 세 번째 단계로 각 규칙에 대하여 불필요한 속성을 식 $f'(A)=\prod_{y\in U}\{\sum\delta(x,y):y\in U \text{ and } \delta(X,Y)\neq 0\}$ 을 이용하여 제거하고, 마지막 단계로 알고리즘 최소화를 수행하면, Table 3은 다음과 같은 최소 알고리즘으로 표현된다.

{ $a_0d_0\rightarrow f_1$, $b_0d_1\rightarrow f_1$, $a_0b_0d_2\rightarrow f_1$,
$a_0b_1d_1\rightarrow f_1$, $a_1b_1d_2e_0\rightarrow f_1$, $e_1\rightarrow f_1$,
$a_1d_1\rightarrow f_1$, $a_2b_1\rightarrow f_1$, $a_0b_1d_2\rightarrow f_1$,
$a_1d_0\rightarrow f_1$, $b_2\rightarrow f_1$, $a_1b_0d_2e_0\rightarrow f_1$,
$a_1b_1e_2\rightarrow f_1$, $b_0e_2\rightarrow f_1$ }

최종적으로 획득된 최소 의사결정 알고리즘은 임의의 모듈에 대한 재사용성을 판정하는 규칙들의 모임으로 이루어진다.

또한 실험들은 속성들의 변화와 모듈(규칙)들의 삽입, 삭제뿐만 아니라 속성들의 분류 기준의 변화에 능동적으로 대처하여 재사용 결정 알고리즘을 생성하는 것을 보여주었다.

Ⅵ. 결 론

본 연구에서는 실제로 재사용되고 있는 부품들과 산업 현장에서 그 타당성을 인정받은 정량적인 척도들과 분류 기준을 이용하여, 새로운 부품의 삽입과 기존 부품들의 삭제, 분류 기준의 변경 등의 환경 변화에 능동적으로 대처할 수 있는 적응성이 있는 재사용 결정 알고리즘을 얻기 위해서 Rough set 이론을 이용한 재사용성 결정 모델을 제안하였다.

본 연구에서 제안된 모델을 재사용되고 있는 C 컴파일러의 175개에 해당하는 Run time 라이브러리에 적용하여 다음과 같은 결론을 얻었다.

첫째, 프로그램의 크기, 구조, 알고리즘에서의 복잡도의 변화와 새로운 품질 기준을 쉽게 수용하여 모델 외부에서의 환경 변화에 능동적으로 대처할 수 있다.

재사용 부품 라이브러리에 대한 삽입과 삭제 및 부품의 변경이 발생하거나, 분류 기준의 변경이 발생 시 부품 라이브러리에 있는 모듈들의 정보를 추출하여 새로운 최소 재사용성 결정 알고리즘을 능동적으로 생성할 수 있다.

둘째, 재사용성 결정 알고리즘의 생성이 자동화 될 수 있으므로 재사용 관리 시스템에 가장 적합한 알고리즘을 쉽고 용이하게 구성할 수 있다.

셋째, 기존의 라이브러리 관리 시스템의 재사용성 평가 서브 시스템으로 이용될 수 있다.

본 연구에서 제안된 모델을 범용적으로 사용하기 위해서는 특정 언어를 사용해야하는 제한사항이 없이 다른 언어들을 포함할 수 있어야 하며, 절차 언어에 한정하지 않고 객체 지향 언어들에 대한 품질 평가도 수용할 수 있어야 하며, 각 속성들의 중요도를 평가하고 그들의 활용 방안에 대한 연구가 수행되어야한다.

[참 고 문 헌]

[1] L.J.Arthur, "Measuring Programmer Productivity and Software Quality," John Wiley & Sons, New York, 1985, pp.138-142

[2] Caldiera, G. and V.R. Basili, "Identifying and Qualifying Reusable Software Components", IEEE Computer, Feb, 1991, pp.61-70

[3] Langergan.R.G and Grasso.C.A, "Software engineering with reusable engine and code," IEEE transaction on software engineering, Vol. SE-10, No.5, Sep, 1984, pp.498-501

[4] M.Halstead, "Element of Software Science," Elsevier Noteh-Halland, New-York, 1977

[5] P.A.V.Hall "Software Components and Reuse-Getting More out of Your Code," Information and Software Technology, Feb, 1987, pp.38~43

[6] Lewis John, Henry Salie, "A Methodology for Integrating Maintainability Using Software Metrics," Proceedings:Conference on Software Maintenance, Miami, Florida, IEEE, Oct 16-19, 1989, pp.32-39

[7] T.McCabe, "A Complexity Measure," IEEE Trans.SE., SE-2, 1976, pp.308-320

[8] Pawlak Z., "Rough sets, International Journal of Computer and," Information Sciences, 11, pp.341-356

[9] Z. Pawlak, "Rough Logic," Bulletin of the Polish Academy of Sciences, Technical Science 35, 1997, pp.253-258

[10] Z. Pawlak, "Rough Sets-Theoretical Aspects of Reasoning about Data," Kluwer Academic Publishers, Dordrecht, Boston, London, 1991

[11] Szentes.J., Gras j., "Some Practical Views of Software - Complexity metrics and a Universal Measurement Tool," First Australian Software Engineering Conference, Canberra, May, 1986, pp.14-16

[122.3.4.5.] Horst Zuse "Software Complexity-measures and methods", Walter de Gruyter, Berlin • Ne6.w York, 1991, pp.25-37

[13] 강문호 "재사용을 위한 소프트웨어 부품의 분류 방식 및 검색모델", 박사학위 논문, 전남대학교, 1994

[14] 정화식, 양해술, "소프트웨어 재사용가능성의 정량적인 측도", 한국 정보처리학회 논문지 제2권 2호, 1995, pp.176-184