한국 퍼지 및 지능시스템학회 논문지 1999, Vol. 9, No. 6 (1999)583~590

러프논리에 기반한 객체지향 컴포넌트의 재사용 결정 알고리즘 생성

Generation of Reusability Decision Algorithm of Object-Oriented Components based on Rough Logic

박병권 · 이성주

Byung-Gweun Bak and Sung-Joo Lee

조선대학교 전자계산학과

요 약

본 연구에서는 컴포넌트의 획득시점에서 객체지향 컴포넌트의 재사용 가능성을 능동적으로 결정할 수 객체지향 컴포넌트의 재사용성 결정 모델을 제안한다. 먼저, 제안된 모델은 객체지향 컴포넌트의 재사용 결정을 위한 속성들을 선택한다. 다음으로, 여러 연구들에서 제시된 객체지향 클래스의 품질 척도들과 품질 분류기준들에 근거하여 실제 재사용중인 컴포넌트들에서 재사용 정보를 추출한다. 마지막으로 러프집합을 이용하여 추출된 정보에서 객체지향 컴포넌트의 재사용 결정알고리즘을 생성한다.

ABSTRACT

We propose the reusability decision model of the object-oriented components, which can decide the potentiality of reusability of the object-oriented components actively. Fisrt, we select attributes for the reusability decision of the object-oriented components. Then, we acquire information from the reused components based on the quality measures and criteria proposed by many researches. Lastly, we generate algorithm for the reusability decision of the object-oriented components from the acquired information employing rough set.

1. 서 론

소프트웨어 재사용은 과거의 비슷한 개발 경험이나 그에 관련된 코드 또는 자료를 재사용 함으로써 개발 비용의 감소, 재사용에 의한 다중 테스팅과 에러 제거에 따른 높은 품질의 소프트웨어 산물, 유지보수 비용 감소, 높은 신뢰성과 높은 응용 효율성, 프로토타입의 능력 증가 등과 같은 경제적인 이점을 가진다[5].

재사용성(reusability) 측정 메트릭은 컴포넌트의 유용성을 결정할 수 있는 측정값을 제공할 수 있기 때문에, 재사용 라이브러리 관리자와 개발자(사용자)들이 요구하는 것은 컴포넌트들의 재사용성을 측정할 수 있는 방법이다[16].

따라서, 컴포넌트가 재사용에 널리 이용되고 위해서는 먼저 컴포넌트의 재사용성에 대한 측정이 이루어져야 한다. 재사용성은 컴포넌트가 재사용될 수 있는 정도이다[18].

컴포넌트의 재사용성 측정은 두 단계로 나눌 수 있다.

첫째, 컴포넌트의 재사용도(the degree of reusability)의 측정이다. 컴포넌트의 재사용도 측정은 컴포넌트가 재사용 되는 시점에서 컴포넌트의 이해와 적용을 위해 소요되는 노력의 정도를 측정한다. 이런 연구들은 다수의 유사한 부품들을 재사용에 적합한 순으로 정렬하고 사용자는 제시된 순위 자료를 참고하여 최종적으로 컴포넌트의 재사용을 결정한다[21].

기능중심 컴포넌트의 적용과 이해에 적용되는 노력을 측정하려는 연구들[15,17,19,21]과 객체지향 컴포넌트인 클래스의 재사용 비용의 예측이나 재사용도의 측정을 위한 연구들[2,3,7,8,13,20]이 수행되어 왔다.

둘째, 컴포넌트의 재사용 가능성(the potentiality of reusability)의 측정이다. 재사용 가능성 측정은 컴포넌트의 획득 시점에서 컴포넌트의 품질을 평가하여 컴포넌트의 재사용 여부를 판정한다. 재사용 가능성의 측정은 일차적으로 품질이 평가된 컴포넌트를 사용자에게 제공함으로써 사용자의 불신을 해소시킬 수 있다.

컴포넌트의 재사용 가능성을 측정하는 연구들은 [4,20,22,24] 컴포넌트의 품질을 정량적으로 측정하여 평가 기준들을 설정하고 설정된 기준과 비교하여 컴포넌트의 재사용 여부를 판정을 돕는다.

그러나 컴포넌트의 획득 시점에서 컴포넌트의 재사용가능성을 판정하는 기존의 연구들은 다음과 같은 문제점을 갖고 있다.

첫째, 기존의 연구들[4,22,24]은 객체지향 컴포넌트보다는 기능중심 컴포넌트에 더욱 적합하다. 객체지향 컴포넌트에 대해서는 또 다른 평가기준들이 고려되어야 한다[23].

둘째, 소프트웨어 내·외적인 환경변환에 능동적으로 대처하지 못한다[24]. 재사용을 위한 범용적으로 개발된 재사용 컴포넌트들은 특정 목적을 위해 개발된 버전보다는 규모가 더 크고, 더 복잡한 경향이 있으므로, 재사용 컴포넌트의 재사용 식별을 위한 메트릭들은 컴포넌트의 재사용을 나타내기 위해서 개정되어야 한다[9]. 또한 소프트웨어 기능에 대한 사용자의 요구가 복잡해지고, 다양화됨에 따라 소프트웨어 재사용 컴포넌트 역시 이런 사용자의 요구를 충족시키기 위해서 점차 기능과 구조가 복잡해지고 있다[24]. 그러나 정량된 메트릭들을 이용하여 측정된 측정값에 근거한 재사용성 결정[20,22]은 이런 요구에 능동적으로 대체하지 못한다.

셋째, 재사용 결정을 위한 메트릭의 삽입과 삭제가 어렵다. 몇가지의 메트릭들을 결합하여 컴포넌트의 재사용을 결정하는 혼합적 메트릭들[4,20,22]의 경우에 새로운 메트릭을 삽입하거나 기존의 메트릭을 삭제하는 경우에 전체의 측정 메카니즘들이 수정되어야 한다.

따라서, 본 연구에서는 컴포넌트의 획득시점에서 객체지향 컴포넌트의 재사용 가능성을 능동적으로 결정할 수 객체지향 컴포넌트의 재사용성 결정 모델을 제안한다.

본 연구에서는 여러 연구들[1,4,6,12,14,20]에서 제시된 객체지향 클래스의 품질척도들과 품질 분류기준들에 근거하여 실제 재사용중인 컴포넌트들에서 추출된 정보와 러프집합을 이용하여 객체지향 컴포넌트의 재사용 결정알고리즘을 생성한다.

러프집합은 데이터로부터 최소 의사결정 규칙들의 집합을 생성하며, 획득된 결과들의 직접적 해석을 제공한다.

본 논문의 구성은 2장에서 러프논리에 대하여 간략하게 고찰하고, 3장에서 재사용 결정알고리즘 생성 모델을 정의하고, 4장에서 재사용 결정 알고리즘 생성 및 소프트웨어 내외적 환경 변화에 능동적으로 대처하여 재사용 결정 알고리즘을 생성하는 과정을 보여주며, 5장에서는 결론 및 향후 연구 과제를 기술한다.

2. 러프논리

불완전한 지식을 다루고 조작하는 문제에 대한 많은 접근 방법 중 가장 널리 알려진 방법론은 Zadeh에 의해 제안된 Fuzzy 집합 이론이다. 이 문제에 대한 또 다른 시도로서, Pawlak[10]에 의해 제안된 러프집합개념은 모호성과 불확실성에 대한 새로운 수학적 접근 방법이다[11].

러프집합 이론은 데이터에 숨겨진 패턴들을 발견하는 효율적인 알고리즘을 제공하며, 쉽게 최소 데이터 집합을 찾을 수 있으며, 데이터의 중요성을 평가하며, 데이터로부터 최소 의사결정 규칙들의 집합을 생성하며, 이해 하기가 쉽고 획득된 결과들의 직접적 해석을 제공한다.

객체들의 분류에 관한 지식은 정보 시스템으로 분류될 수 있으며, 정보시스템의 각 항목은 객체에 관한 내용으로 보여질 수 있다. Rough logic은 러프 집합 이론에 근거하여 분류로서 간주되는 지식을 추론할 수 있게 한다.

따라서 본 연구에서는 러프집합을 이용하여 컴포넌트의 분류지식에서 재사용 의사결정을 위한 최소알고리즘을 생성한다.

2.1 알고리즘과 알고리즘의 감축

러프논리에서의 의사결정 규칙 $\Phi \rightarrow \Psi$는 선행자(predecessor)와 후행자(successor)를 이용한 함축성 있는 표현식이다.

알고리즘 (P, Q)에서, a∈P일 때, ((P, {a}), Q)가 불일치이면 속성 a는 알고리즘 (P, Q)에서 필요이고, 모든 속성(a∈P)이 알고리즘 (P, Q)에서 필요이면 알고리즘(P, Q)는 독립이다. 알고리즘(P, Q)에서 모든 필요인 속성들의 집합이 알고리즘 (P, Q)의 core이다. R⊆P인 알고리즘 (P, Q)가 독립이고 일치이면 R(R⊆P)은 알고리즘 (P, Q)에서 P의 reduct이다. R이 알고리즘 (P, Q)에서 P의 reduct이면, 알고리즘 (R, Q)는 알고리즘 (P, Q)의 reduct이다.

2.2 결정 규칙의 감축

의사결정 알고리즘은 규칙들의 집합에서 조건들의 감축을 통해서 단순화된다

식 Φ가 P-기본식이고 Q⊆P이면, Φ/Q는 Φ에서 a∈P-Q인 모든 원자식 (a, Va)들을 제거함으로써 식 Φ로부터 획득되는 Q-기본식이다. 식 $\Phi \rightarrow \Psi$가 PQ-규칙이고, a∈P일 때, $|\Phi \rightarrow \Psi| \subseteq |\Phi/(P-a) \rightarrow \Psi|$이면 속성 a는 규칙 $\Phi \rightarrow \Psi$에서 필요이고, $\Phi \rightarrow \Psi$에서 필요인 모든 속성들의 집합이 $\Phi \rightarrow \Psi$의 core이다. 모든 속성들 a∈P가 $\Phi \rightarrow \Psi$에서 필요이면, $\Phi \rightarrow \Psi$는 독립이고, $\models_S \Phi \rightarrow \Psi$가 $\models_S \Phi/R \rightarrow \Psi$을 의미하면 R은 PQ-규칙 $\Phi \rightarrow \Psi$의 reduct이다.

2.3 결정규칙의 최소화

알고리즘은 동일한 의사결정류들과 관련된 불필요한 의사결정 규칙들을 제거함으로써 최소화한다.

알고리즘 (P, Q)에서 특정 후행자 Ψ를 갖는 기본 알고리즘 /A에서 모든 기본 규칙들의 집합은 $/A_\psi$로 표기하고 $|P_\psi$는 $/A_\psi$에 속하는 의사결정 규칙들의 모든 선행자들의 집합이다. $\vee|P_\psi$이 $|P_\psi$에서의 모든 식들의 논리합을 나타낼 때, $\models_S \vee|P_\psi \equiv \vee\{|P_\psi - \{\Phi\}\}$ 이면, /A에서의 기본 의사결정 규칙 $\Phi \rightarrow \Psi$는 /A에서 필요이다. $/A_\psi$에서의 모든 의사결정 규칙이 필요이며 규칙들의 집합 $/A_\psi$는 독립이다. $/A_\psi$의 의사결정 규칙들의 독립이고, $\models_S \vee|P \equiv \vee|P'_\psi$이면, /A'는 $/A_\psi$의 reduct이다.

러프 집합의 모든 개념들은 러프 논리로 표현할 수 있으며, 러프 논리는 러프 집합 이론에 근거하여 분류로 간주되는 지식을 논리적으로 추론할 수 있게 한다.

3. 객체지향 컴포넌트의 재사용성 결정 알고리즘 생성

3.1 객체지향 컴포넌트의 재사용성 결정 속성

재사용 컴포넌트의 본질적인 품질과 관련된 기준[9]들을 이용하여 객체지향 컴포넌트의 재사용 결정하기 위해서는 기존의 기능 중심 컴포넌트의 평가방법들(Lines of Code, McCabe의 Cyclomatic Number, Halstead의 Software Science)뿐만 아니라, 객체 지향 컴포넌트의 평가 방법들을 도입하여야 한다[20,23].

따라서 본 연구에서는 기능중심 컴포넌트 평가 방법들인 LOC(Lines of Code), McCabe의 Cyclomatic Number, Halstead의 Effort와 객체 지향 컴포넌트 평가 방법들인 Classs의 Cohesion 측정방법들, DIT (Depth of Inheritance Tree), EXT(EXTension) 등을 조합하여 객체지향 컴포넌트의 재사용 결정 알고리즘을 생성한다.

재사용 컴포넌트의 재사용 결정은 실질적인 재사용 컴포넌트인 클래스를 분류하는 기준에 따라 클래스의 재사용성을 분류하는 것으로, 여러 연구들은 실험적인 연구들을 통해서 클래스를 분류하는 기준을 제시하였다.

산업현장에서의 실험적인 연구들은 하나의 모듈에서, 코드의 라인수(LOC)는 30이하, 복잡도(Cyclomaticnumber)는 10 이하, 프로그램 노력도(Effort)는 1,000이하가 적합하다고 제시하였고[1,4,14], 하나의 클래스에서의 멤버 함수의 개수는 7±2가 적합하고, 클래스에서의 LCOM은 2이하, DIT(클래스의 상속도)는 3 이하가 적합하며, EXT(확장성)는 1이 적합하다고 제시하였다[6,12,20].

그러므로 본 연구에서는 기존의 연구들에서 제안된

표 1. 분류 기준 표

측정인자 \ 분류		0	1	2	3
멤버함수의 개수		≦ 5	5<~≦ 7	7<~≦ 9	>9
LOC	평균	≦ 30	>30		
	위반	있음	없음		
CNC	평균	≦ 10	>10		
	위반	있음	없음		
effort	평균	≦10000	>10000		
	위반	있음	없음		
LCOM		≦2	2<~ ≦9	9<~≦ 200	>200
DIT		≦3	3<~ ≦6	6<~ ≦10	>10
EXT		0	0<~ ≦0.5	0.5<~ ≦1	>1

분류 기준들에 근거하여 표 1과 같이 객체지향 컴포넌트의 재사용 분류기준을 설정하였다.

이러한 경계들이 산업 현장에서의 연구와 경험을 통해서 증명되어 왔지만 이 경계들은 다소 임의적이며, 실제로 각 현장에서의 요구들은 변할 수 있다[Arth, 85]. 그러므로 본 모델에서는 이 경계들이 변경되면 전체의 규칙들은 재구성되어 변경된 값들을 수용하는 새로운 규칙들을 생성하게 한다.

3.2 재사용성 결정 알고리즘 생성

재사용 결정은 실제로 컴포넌트를 분류하는 기준에 따라서 컴포넌트의 재사용 가능성을 분류하는 것이며, 컴포넌트 분류에 관한 정보들은 컴포넌트 재사용에 관한 지식이 된다.

러프집합을 이용한 결정알고리즘의 생성은 다음과 같은 단계로 구성된다.

[Step 1] 속성 및 속성 값 도메인의 정의
[Step 2] 지식 표현 및 중복된 규칙의 제거
[Step 3] 알고리즘의 감축
[Step 4] 의사결정 규칙의 감축
[Step 5] 알고리즘의 최소화

3.2.1 속성 및 속성 값 도메인의 정의

본 연구에서 재사용 판정의 위한 속성 도메인은 다음과 같이 정의된다.

A = {Number of Methods, Average of LOC, Violation of LOC, Average of Cyclomatic number, Violation of Cyclomatic number, Average of Effort, Violation of Effort, LCOM, DIT, EXT}

또한 표 1의 분류 경계 값들을 기준으로 하여 각

속성에 대한 속성값 도메인은 다음과 같이 정의된다.

$V_{NoM} = \{0, 1, 2, 3\}$
$V_{AoL} = V_{VoL} = V_{AoC} = V_{VoC} = V_{AoE} = V_{VoE} = \{0, 1\}$
$V_{LCOM} = V_{DIT} = V_{EXT} = \{0, 1, 2, 3\}$
※NoM : number of method, AoL : average of LOC
VoL : violation of LOC, AoE : average of Effort
AoC : average of Cyclomatic number
VoC:violation of Cyclomatic number
VoE : violation of Effort
EXT : EXTension
LC : LCOM, DIT : Depth of Inheritance Tree

표 1은 각 현장에서의 요구들의 변화에 따라서 변경될 수 있으며, 그런 경우 속성 및 속성값 도메인은 자동적으로 재 정의된다.

3.2.2 지식표현 및 중복된 규칙의 제거

재사용 컴포넌트들에서 각 속성들에 대한 측정값을 추출하고, 분류 기준에 근거하여 분류하며, 지식 표현 시스템(Knowledge Representation System)으로 표현된 지식들은 재사용에 관한 지식들을 포함하며, 지식 표현 시스템에 나타난 지식들은 재사용을 판단하는 규칙들이 된다.

지식 표현 시스템에는 중복된 규칙이 존재할 수 있으므로 중복된 규칙을 제거하며, 원소 범주들의 집합을 그대로 유지한 채, 불필요한 분할인 속성 또는 동치관계들을 제거하는 지식의 감축을 수행한다. 이 과정은 지식 베이스에서 모든 불필요한 지식을 제거하여 필요한 부분만을 갖게 한다.

3.2.3 알고리즘의 감축

알고리즘의 감축은 전체 시스템에서 불필요한 속성들을 제거하는 것으로서, 2.1에서 제시된 과정과 유사하다. 그러나, 본 연구에서 제시한 모델은 재사용 가능 컴포넌트들에서 최소 재사용 결정 알고리즘을 찾는 것이므로 다음과 같이 알고리즘의 의미에 대한 정의를 통하여 알고리즘의 감축을 수행한다.

알고리즘 (P, Q)가 규칙 r_1, r_2, r_3, ……, r_n의 모임일 때, 알고리즘 (P, Q)의 의미는 규칙들의 모임으로, 아래와 같이 정의한다[24].

$$|(P, Q)| = \{|r_1|, |r_2|, |r_3|, \cdots, |r_n|\}$$

알고리즘 (P, Q)에서 $a \in P$일 때, $|((P-\{a\}), Q)| \neq |(P, Q)|$이면 속성 a는 알고리즘 (P, Q)에서 필요이고, 모든 속성 $a \in P$가 알고리즘 (P, Q)에서 필요이면, 알고리즘 (P, Q)는 독립이다. $R \subseteq P$인 알고리즘 (R, Q)가 독립이고 $|(R, Q)| = |(P, Q)|$이면 R은 알고리즘 (P, Q)에서 P의 reduct이다.

일차적으로 감축된 알고리즘은 불필요한 속성들이 제거되어 반드시 필요한 속성들로 이루어진 새로운 정보표가 된다.

3.2.4 규칙의 감축

속성의 제거를 통해서 일차적으로 감축된 의사결정 알고리즘은 각 의사결정 규칙에서 불필요한 속성들을 제거함으로써 더욱 단순화 되도록 2.2의 결정규칙 감축의 방법을 적용하여규칙의 감축을 수행한다.

불필요한 속성이 제거된 감축된 규칙들은 그 규칙들에 필요한 속성들만으로 이루어진 새로운 규칙이다.

3.2.5 알고리즘의 최소화

감축된 알고리즘과 규칙에서 동일한 의사결정류들과 관련된 불필요한 의사결정 규칙들의 제거한다. 다른 규칙들이 제거된 규칙들의 역할을 감당할 수 있으므로, 의사결정 규칙들은 의사결정 과정을 방해하지 않고 삭제될 수 있도록 3.3의 알고리즘 최소화 방법을 적용하여 알고리즘의 최소화를 수행한다.

3.3 컴포넌트 재사용 결정

최소 의사결정 알고리즘은 데이터 베이스에 저장되어 재사용성 결정의 기초가 된다. 재사용성 결정은 자체 개발된 컴포넌트나 외부에서 획득된 컴포넌트들의 재사용판정에 관한 질의를 처리한다.

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

4. 실험 및 결과

본 연구에서는 테스트를 위해 표 2와 같은 측정값을 갖는 객체지향 재사용 컴포넌트들로 구성된 재사용 시스템이 있다고 가정한다.

4.1 재사용성 결정 알고리즘 생성

표 2와 같은 측정값들을 갖는 재사용 컴포넌트들을 표 1의 분류 기준에 따라 속성값 도메인으로 분류하여 표 3와 같은 지식 표현 시스템을 구성하였다.

표 3에서 규칙 1의 {3, 0, 0, 0, 0, 0, 0, 2, 0, 2}는 3.3.1절에서 정의된 속성값 도메인들이다.

4.1.1 중복 규칙의 제거

최소 재사용 결정을 생성하기 위한 첫 번째 단계로 지식 표현 표에서 중복된 규칙을 제거한다. 표 3의

표 2. 27개의 클래스에 대한 측정값들

컴포넌트	MoM	Tloc	AoL	VoL	Tcyc	AoC	VoC	컴포넌트	Teff	AoE	VoE	LC	DIT	EXT
1	15	37	2.47	0	23	1.53	0	1	4086.02	272.40	0	63	1	0.59
2	10	111	11.10	1	74	7.4	1	2	9268.45	926.95	0	0	0	0
3	6	38	6.33	0	30	5	0	3	10720.99	1786.83	0	0	0	0
4	10	139	13.9	1	74	7.4	1	4	71110.26	7111.03	0	0	0	0
5	13	64	4.92	0	61	4.69	1	5	19323.07	1486.39	0	0	0	0
6	23	57	2.48	0	198	8.61	1	6	4430.02	192.61	0	0	0	0
7	8	79	9.88	1	53	6.62	1	7	61801.12	7725.14	0	0	0	0
8	18	21	1.17	0	137	7.61	1	8	61801.12	7725.14	0	0	0	0
9	19	24	1.26	0	114	6	1	9	765.92	40.31	0	0	0	0
10	21	97	4.62	0	122	5.81	1	10	18819.07	896.15	0	0	0	0
11	6	38	6.33	0	31	5.17	0	11	15066.65	2511.11	0	0	0	0
12	22	173	7.86	1	228	10.36	1	12	143856.85	6538.95	0	0	0	0
13	19	132	6.95	0	135	7.11	1	13	44264.60	2329.72	0	0	0	0
14	3	18	6	0	9	3	0	14	2221.76	740.59	0	0	0	0
15	21	103	4.9	0	193	9.19	1	15	30887.5	1470.83	0	0	0	0
16	24	93	3.88	0	191	7.96	1	16	9685.11	403.55	0	0	0	0
17	5	34	6.8	0	16	3.2	0	17	7563.28	1512.66	0	0	0	0
18	9	34	3.78	0	35	3.89	1	18	3782.38	420.26	0	0	0	0
19	21	50	2.38	0	194	9.24	1	19	6400.65	304.79	0	0	0	0
20	8	126	15.75	0	38	4.75	0	20	52912.79	6614.10	0	0	0	0
21	24	56	2.33	0	306	12.75	1	21	17194.28	716.43	0	0	0	0
22	6	47	7.83	0	31	5.17	0	22	10997.05	1832.84	0	0	0	0
23	11	44	4	0	154	14	1	23	54724.71	4974.97	0	0	0	0
24	35	483	13.80	1	613	17.51	1	24	248139.78	7089.71	0	0	0	0
25	7	7	1	0	22	3.14	1	25	551.31	78.76	0	0	0	0
26	4	4	1	0	10	2.5	0	26	118.24	29.56	0	0	0	0
27	12	87	7.25	0	58	4.83	1	27	26883.73	2240.31	0	0	0	0

※Tloc=total LOC,Tcyc=total cuclomatic number, Teff=total effort.

표 3. 지식 표현 시스템

U	NoM	AoL	VoL	AoC	VoC	AoE	VoE	LC	DIT	EXT	U	NoM	AoL	VoL	AoC	VoC	AoE	VoE	LC	DIT	EXT
1	3	0	0	0	0	0	0	2	0	2	14	0	0	0	0	0	0	0	0	0	0
2	3	0	1	0	1	0	0	0	0	0	15	3	0	0	0	1	0	0	0	0	0
3	1	0	0	0	0	0	0	0	0	0	16	3	0	0	0	1	0	0	0	0	0
4	3	0	1	0	1	0	0	0	0	0	17	0	0	0	0	0	0	0	0	0	0
5	3	0	0	0	1	0	0	0	0	0	18	2	0	0	0	1	0	0	0	0	0
6	3	0	0	0	1	0	0	0	0	0	19	3	0	0	0	1	0	0	0	0	0
7	2	0	1	0	1	0	0	0	0	0	20	2	0	0	0	0	0	0	0	0	0
8	3	0	0	0	1	0	0	0	0	0	21	3	0	0	3	1	0	0	0	0	0
9	3	0	0	0	1	0	0	0	0	0	22	1	0	0	0	0	0	0	0	0	0
10	3	0	0	0	1	0	0	0	0	0	23	3	0	0	3	1	0	0	0	0	0
11	1	0	0	0	0	0	0	0	0	0	24	3	0	1	3	1	0	0	0	0	0
12	3	0	1	3	1	0	0	0	0	0	25	1	0	0	0	1	0	0	0	0	0
13	3	0	0	0	1	0	0	0	0	0	26	0	0	0	0	0	0	0	0	0	0
											27	3	0	0	0	1	0	0	0	0	0

규칙들 중에서 중복된 규칙을 제거하면 표 4와 같은 11개의 규칙으로 구성된 지식 표현 표가 된다.

4.1.2 알고리즘의 감축

두 번째 단계는 전체 지식 표현 시스템에서 불필요

표 4. 표 2에서 중복된 규칙 제거한 지식 표현 표

	NoM	AoL	VoL	AoC	VoC	AoE	VoE	LC	DIT	EXT
r_1	3	0	0	0	0	0	0	2	0	2
r_2	3	0	1	0	1	0	0	0	0	0
r_3	1	0	0	0	0	0	0	0	0	0
r_4	3	0	0	0	1	0	0	0	0	0
r_5	2	0	1	0	1	0	0	0	0	0
r_6	3	0	1	3	1	0	0	0	0	0
r_7	0	0	0	0	0	0	0	0	0	0
r_8	2	0	0	0	1	0	0	0	0	0
r_9	2	0	0	0	0	0	0	0	0	0
r_{10}	3	0	0	3	1	0	0	0	0	0
r_{11}	1	0	0	0	1	0	0	0	0	0

표 5. 알고리즘 감축

	NoM	VoL	AoC	VoC
r_1	3	0	0	0
r_2	3	1	0	1
r_3	1	0	0	0
r_4	3	0	0	1
r_5	2	1	0	1
r_6	3	1	3	1
r_7	0	0	0	0
r_8	2	0	0	1
r_9	2	0	0	0
r_{10}	3	0	3	1
r_{11}	1	0	0	1

한 속성을 제거한다. 전체 지식 표현 시스템에서 불필요한 속성을 제거하기 위하여 표 4에 대한 core와 reduct를 3.2.3절에 제시된 과정을 통하여 구하면, {NoM, VoL, AoC, VoC}가 표 4의 core이면서 reduct이므로 표 4는 표 5로 감축된다.

4.1.3 규칙의 감축

각 규칙에 대한 core와 reduct를 구하여 각 규칙들에서 불필요한 속성들을 제거하기 위하여 2.2절의 감축과정을 표 5에 적용하면 표 5는 표 6으로 감축된다.

4.1.4 알고리즘의 최소화

마지막 단계인 알고리즘 최소화 단계에서는 임의의 규칙이 다른 규칙들이 생성하는 역할을 담당할 수 있으면, 그 규칙은 결정 과정에서 제거될 수 있다. 이 과정은 임의의 집합들의 합집합에서 불필요한 집합을 제거하는 것과 유사하다. 표 6의 모든 의사결정 규칙들은 이미 감축된 것이므로, 더 이상 단순화될 수 없다. 그러므로 표 2는 최종적으로 그림 1과 같은 최소 알고리즘으로 표현된다. 최종적으로 획득된 최소 의사결정 알고리즘은 임의의 클래스에 대한 재사용성을 판정하는 규칙들의 집합을 구성한다.

표 6. 규칙의 감축

	NoM	VoL	AoC	VoC
r_1	3	-	-	0
r_2	3	1	0	-
r_3	1	-	-	0
r_4	3	0	0	1
r_5	2	1	-	-
r_6		1	3	-
r_7	0	-	-	-
r_8	2	0	-	1
r_9	2	-	-	0
r_{10}		0	3	-
r_{11}	1	-	-	1

(Rule_1) if NoM > 9 and VoC="no" then Reusable
(Rule_2) if NoM > 9 and VoL="yes" and AoC ≤ 10 then Reusable
(Rule_3) if 5<NoM ≤7 and VoC="no" then Reusable
(Rule_4) if NoM > 9 and VoL="no" and AoC ≤ 10 and VoC="no" then Reusable
(Rule_5) if 7<NoM ≤9 and VoL="yes" then Reusable
(Rule_6) if VoL="yes" and AoC >50 then Reusable
(Rule_7) if NoM ≤5 then Reusable
(Rule_8) if 7<NoM ≤9 and VoL="no" and VoC="yes" then Reusable
(Rule_9) if 7<NoM ≤9 and VoC="yes" then Reusable
(Rule_10) if VoL="no" and AoC >50 then Reusable
(Rule_11) if 5<NoM ≤7 and VoC="yes" then Reusable

그림 1. 최소 의사결정 알고리즘

4.1.5 환경변화에 따른 결정 알고리즘의 재구성

속성의 제거 및 추가, 컴포넌트의 삽입 등과 같은 환경 변화에 능동적으로 적응하는 알고리즘을 재구성하여 본다.

속성의 제거 및 추가의 경우, 속성들이 추가 또는 변경되었다는 정보가 재사용 결정 모델에 통보되면, 결정 모델은 재사용 컴포넌트들에서 변경된 속성들에 대한 측정값들을 추출하여 능동적으로 재사용 결정 알고리즘을 생성한다. 표 4에서 속성 VoC(Violation of Cyclomatic number)를 제거하고, 속성 VoC를 제외한 나머지 속성들로 구성된 최소 결정 알고리즘은 표 7과 같다.

기존의 재사용 결정 알고리즘을 통해서는 재사용 불가능한 것으로 판명된 컴포넌트를 재사용 라이브러리에 강제적으로 삽입하는 경우가 있을 수 있다. 이 경

표 7. 속성 VoC를 제거한 지식 표현 시스템에 대한 최소 결정 알고리즘

	NoM	VoL	AoC	DIT
r_1	-	-	-	2
r_2	3	0	1	-
r_3	1	-	0	-
r_4	2	-	1	-
r_5	-	3	-	-
r_6	0	-	-	-
r_7	2	-	0	-
r_8	1	-	1	-
r_9	3	-	0	0

표 8. 새로운 규칙이 추가된 지식 표현 시스템에 대한 최소 결정 알고리즘

	NoM	VoL	AoC	DIT	EXT
r_1	-	-	-	2	-
r_2	3	0	1	-	-
r_3	1	-	0	-	-
r_4	2	-	1	-	-
r_5	-	3	-	-	-
r_6	0	-	-	-	-
r_7	2	-	0	-	0
r_8	1	-	1	-	-
r_9	3	-	0	0	-
r_{10}	-	-	-	-	1

우 재사용 결정 알고리즘은 추가된 컴포넌트의 정보를 기존의 재사용 결정 알고리즘과 결합하여 새로운 결정 알고리즘을 능동적으로 생성한다. 표 4에서 속성 VoC(Violation of Cyclomatic number)가 제거되고, 새로운 컴포넌트(규칙: NoM_2 AoL_0, VoL_0, AoC_0, AoE_1, VoE_0, LC_0, DIT_0, EXT_1)가 삽입된 지식 표현 시스템의 최소 결정 알고리즘은 표 8과 같다.

4.2 실험 결과

먼저, 기존의 재사용 가능성 평가를 위한 연구들은 주로 기능중심 컴포넌트를 대상으로 하는데 비해 본 연구는 객체지향 컴포넌트인 클래스에 대한 재사용 결정알고리즘 생성한다. 여러 연구들에서 제시한 클래스의 품질 척도들과 품질 분류 기준들에 근거하여 객체지향 컴포넌트의 재사용 결정을 위한 속성들을 정의하고, 재사용 컴포넌트들에서 추출된 정보들을 이용하여 컴포넌트의 재사용결정 알고리즘을 생성한다.

둘째, 러프집합을 이용함으로 제안된 모델은 다양한 속성들에 근거한 재사용 결정 알고리즘을 쉽게 도출할 수 있다. 또한 메트릭들의 추가와 삭제가 용이하게 이루어진다.

셋째, 소프트웨어의 내외적 환경변화에 능동적으로 대처할 수 있다. 소프트웨어 기능에 대한 사용자의 요구가 복잡해지고, 다양화됨에 따라 재사용 컴포넌트 역시 이런 사용자의 요구를 충족시키기 위해서 점차 기능과 구조가 복잡해지고 있는 상황에서 정량화된 메트릭들을 이용하여 측정된 측정값에 근거한 재사용성 결정은 이런 요구에 능동적으로 대체하지 못하지만, 본 모델은 이러한 요구를 쉽게 수용하여 변경된 정보에 근거하여 재사용 결정 알고리즘을 동적으로 재구성한다.

5. 결 론

본 연구에서는 기존의 재사용가능성 결정에 관한 연구들이 갖고 있는 문제들을 해결하기 위해서 러프논리에 근거한 객체지향 컴포넌트의 재사용 결정알고리즘 생성 모델을 제안한다. 제안된 모델은 객체지향 컴포넌트의 재사용 결정을 위한 속성들을 정의하고 실제로 재사용되고 있는 클래스들에서 추출된 정보에 근거하여 소프트웨어 내외적인 변화에 능동적으로 대처할 수 있는 재사용결정알고리즘을 생성한다.

본 연구에서 제안한 재사용 결정 알고리즘 생성 모델은 기존의 컴포넌트 관리시스템의 하부시스템으로 쉽게 이용될 수 있고 현업에서의 요구의 변화를 동적으로 수용할 수 있다.

현재까지의 연구가 주로 코딩 단계의 클래스에 한정되어 있으므로, 객체지향 설계와 분석단계에서의 적용을 위한 연구와 객체지향 컴포넌트를 재사용하는 시점에서 컴포넌트의 재사용도 평가를 위한 연구가 필요하다.

참고문헌

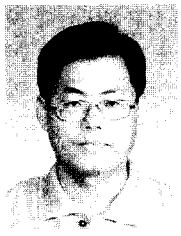
[1] L. J. Arther, "Measuring Programmer Productivity and Software Quality", John Wiley & Sons, New York, pp.138-142, 1985.

[2] J. Basiley and V. Basili, "A meta-model for software development resource expenditures", In Proc. 5th Int. Conf. Software Engineering(ICSE), pp. 107-116, 1981.

[3] J. Bieman. "Deriving measures of software reuse in object-oriented systems", Proc. BCS Workshop on Formal Aspects of measurement(1991): Formal Aspects of Measurement, ed. T. Denvir, R. Herman, and R. Whitty, editors, pp. 63-83. Springer-Verlag, 1992.

[4] Caldiera G. and V. R. Basili, "Identifying and Qualifying Reusable Software Components", *IEEE*

Computer, pp.61-70, Feb, 1991.
[5] J.Chang, "A Reusability Based S/W Development Environment," IEEE ACM SIGSOFT SE. Notes, Vol. 19, No. 2, pp. 57-62, Apr. 1994.
[6] S. R.Chidamber and C. F. Kemerer, "A Metrics Suite for Objected-Oriented Design", *IEEE Trans. Software Eng.,* Vol. 20, No. 6, pp. 476-493, June, 1994.
[7] S. Conte, H. Dunsmore, and V. Shen, "Software Engineering Metrics and Models", The Benjamin/ Cummings Publishing, Inc., Menlo Park, CA, 1986.
[8] N. Fenton. "Software Metrics A Rigorous Approach", Chapman & Hall, London, 1991.
[9] Hafedh Mili, Fatma Mili, Ali Mili, "Reusing Software: Issues and Research Direction", *IEEE Transactions On Software Engineering*, Vol. 21, No. 6, pp. 528-562, 1992.
[10] Pawlak Z., "Rough sets", International Journal of Computer and Information Sciences, 11, pp. 341-356, 1982.
[11] Z. Pawlak, "Rough sets-Theoretical Aspects of Reasoning about Data", Kluwer Academic Publishers, Dordrecht, Boston, London, 1991.
[12] Robert C. Sharble and Sammuel S. Cohen, "The Object-Oriented brewery: A Comparison of Two Object-Oriented Development Methods", ACM SIGSOFT. SE. Note, Vol. 18, No. 2, pp. 60-73, 1993.
[13] R. Selby. "Quantitative studies of software reuse". Software Reusability Vol. II Applications and Experience, ed. T. J. Biggerstaff and A. J. Perlis, Addison-Wesley, pp. 213-233, 1989.
[14] Szentes. J., Gras j., "Some Practical Views of Software-Complexity metrics and a Universal Measurement Tool", First Austrlian Software Engineering Conference, Canberra, pp. 14-16, May, 1986.
[15] B.A. Burton, *et. al.*, "The Reusable Software Library", Software Reuse: Emerging Technology, *IEEE CS Press,* pp. 129-137, 1990.
[16] Judith Barnard, "A New Reusability Metric for Object-Oriented Software", Software Quality Journal, Vol. 7, No. 1, pp. 35-50, 1998.
[17] R. Prieto-Diaz and P. Freeman, Classifying Software for Reusability, Tutorial: Software Reusability, *IEEE CS Press,* pp.106-116, 1987.
[18] W. Frake, C. Terry, "Software Reuse: Metrics and Models", ACM Computer Survey, Vol. 28, No.2, pp. 415-435, 1996
[19] Sung-joo Lee, Wan-gyu Choi, Hwan-mook Chung, "Design and Implementation of the Reuse-Easiness Measurement System Using Fuzzy Logic", 한국퍼지 및 지능시스템 학회 논문지, 제6권 제4호, pp. 17-26, 1996.
[20] 김재생, 송영재, "클래스의 재사용을 위한 정보 분석 및 품질척도", 정보과학회 논문지, Vol. 26, No. 3, pp.393-400, 1999.
[21] 권기현, 이경환, "학습알고리즘을 이용한 재사용 의사결정 지원", 한국정보과학회지, 제 21권, 제12호, pp. 2235-2242, 1994.
[22] 장화식, 양해술, "소프트웨어 재사용가능성의 정량적인 측도", 한국 정보처리학회 논문지, 제2권 2호, pp. 176-184, 1995.
[23] 이경환 외, "소프트웨어 재이용을 위한 연구", 연구보고서, 과학기술처 BSN20592, pp. 102-103, 1989.
[24] 최완규, 이성주, "러프집합을 이용한 재사용성 결정 알고리즘 생성 시스템", 한국퍼지 및 지능시스템 학회 논문지, 제8권 제2호, pp. 96-105, 1998.



박 병 권 (Byung-Gweun Bak)
1988년 : 조선대학교 전자계산학과 졸업 (학사)
1990년 : 조선대학교 전자계산학과 졸업 (석사)
1991 ~ 1994년 : 광주은행 전산업무부
1998년 : 조선대학교 대학원 전자계산학과 박사과정 수료
1995~현재 : 서강정보대학 전자계산학과 조교수
관심분야 : 소프트웨어 공학, 객체지향시스템, 퍼지집합, 러프집합

이 성 주 (Sung-Joo Lee)
1970년 : 한남대학교 물리학과 (학사)
1992년 : 광운대학교 전자계산학과 (이학석사)
1998년 2월 : 대구 효성카톨릭대학교 (이학박사)
1988~1990년 : 조선대학교 전자계산소 부소장
1995~1997년 : 조선대학교정보과학대학장
1981~현재 : 조선대학교 전자계산학과 교수
관심분야 : 소프트웨어 공학, 프로그래밍 언어, 객체지향 시스템, 러프집합