객체 지향 시스템에서의 클래스 간 의존성 강도 측정을 위한 커플링 척도

화지민° 이숙희 권용래
한국과학기술원 전자전산학과
jmhwa@salmosa.kaist.ac.kr shlee@salmosa.kaist.ac.kr kwon@salmosa.kaist.ac.kr

A Coupling Metric for Measuring Strength of Dependency between Classes in Object-Oriented Systems

Jimin Hwa° Sukhee Lee Yong-Rae Kwon
Dept. of Electronic Engineering and Computer Science, KAIST

객체지향 패러다임에서 커플링 척도는 클래스 및 시스템의 품질 예측 모델에서 중요한 지표로 많은 연구들이 진행되어 왔다 [1]. 하지만 대부분의 기존 연구들은 클래스 또는 메소드 간의 의존성(dependency)을 정의하고 그 정의에 따라 대상 클래스가 대상 시스템에서 몇 개의 클래스 또는 메소드와 의존 관계가 존재하는지에 대해서만 보고한다. 즉, 기존의 커플링 척도들은 클래스 간의 의존성 유무만 보여줄 수 있을 뿐 의존성 강도를 측정하지 못한다. 그러므로 이들을 이용한 품질 예측 모델에서 시스템 디자인 개선을 위한 의사 결정을 지원하는 데 한계가 있다.

이러한 문제를 해결하기 위해 이 논문에서는 클래스 간의 의존성 유무뿐만 아니라 클래스 간의 의존성 강도도 측정할 수 있는 커플링 척도 RCBC(Relative Coupling Between Classes)를 제안한다. 의존성 강도는 척도를 적용할 분야와 측정하려는 외부 품질 요소에 따라 다양한 측면에서 고려될 수 있다. 이 논문에서 유지 보수성 측면에서 시스템의 구조적 특성 분석을 통해 RCBC를 정의한다.

재검사 유발성은 한 클래스가 다른 수정된 클래스로부터의 리플 효과 [2]로 인해 재검사되어야 할 가능성을 나타내며, 유지보수 비용적인 측면에서 클래스간의 의존성을 반영한다. 랜덤 유지보수 활동 [3]을 가정할 때, 재검사 유발성 및 그 비용은 호출 관계에 있는 메소드의 SLOC로 표현할 수 있다.

다음은 RCBC를 정의하기 위해 필요한 몇 가지 용어들을 <표 1>에 정의한다.

<표 1. 메소드에 대한 관계와 집합>

용어	정의
$m1 \rightarrow m2$	메소드 $m1$과 $m2$에 대해 $m1$이 $m2$를 호출
$C(S)$	System S내의 모든 클래스들의 집합
$M(c)$	$\{m \mid m$은 $c \in C$의 메소드$\}$
R(c1,c2)	$\{\ m2 \mid c1 \in C(S) \wedge c2 \in C(S) \wedge m1 \in M(c1) \wedge m2 \in M(c2) \wedge (m1 \rightarrow 2) \wedge (c1 \neq c2)\ \}$
$AR(c)$	$\bigcup_{x \in C(S)} R(c,x)$
$MLOC$	MS를 메소드의 집합이라 할 때, $MLOC(MS) = \sum_{m \in MS} m$의 $SLOC$

이를 이용하여 RCBC를 정의하면 다음과 같다.

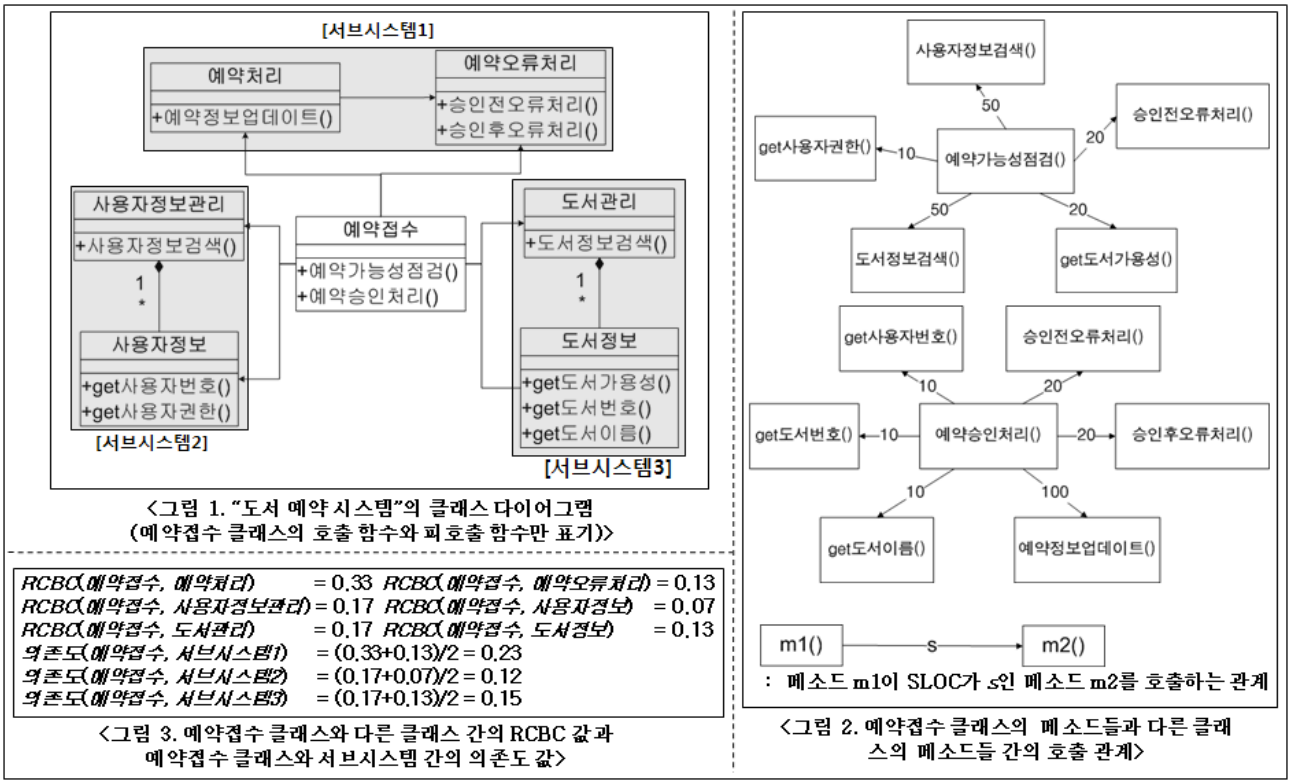
[정의] RCBC : 임의의 두 클래스 $c1, c2 \in C(S)$ (단, $c1 \neq c2$)에 대하여

$$\mathbf{0 \le RCBC(c1,c2) = \frac{MLOC(R(c1,c2))}{MLOC(AR(c1))} \le 1}$$

$RCBC(c1, c2)$의 분모 $MLOC(AR(c1))$은 클래스 $c1$이 시스템 내에서 호출하는 모든 메소드들의 SLOC 합으로 클래스 $c1$의 전체 시스템에 대한 재검사 유발성 측정값을 나타내며 분자 $MLOC(R(c1, c2))$는 클래스 $c1$이 클래스 $c2$에서 호출하는 모든 메소드들의 SLOC 합으로 클래스 $c2$에 의한 재검사 유발성 측정값을 나타낸다. 따라서 $RCBC(c1, c2)$는 전체 시스템 중 클래스 $c2$에 의해 발생하는 클래스 $c1$의 재검사 유발성을 의미하며, 0부터 1사이의 값을 가진다. $RCBC(c1, c2)$가 0의 값을 가지면 클래스 $c1$이 클래스 $c2$의 메소드를 호출하지 않는다는 것을 의미한다. 그리고 $RCBC(c1, c2)$가 1의 값을 가지면 클래스 $c1$이 클래스 $c2$의 메소드만 호출하는 경우로 시스템 내에서 클래스 $c2$에 대해서만 의존성을 가지는 것을 나타낸다. 즉, $RCBC(c1, c2)$는 클래스 c1과 의존관계에 있는 모

든 클래스들 중 클래스 *c2*와의 의존성 강도를 측정하고 있기 때문에 시스템 분해, 리팩토링 등과 같이 클래스 간의 의존성 강도의 측정이 필요한 작업에 유용한 정보를 제공할 수 있다.

RCBC가 의존성 강도 측정이 필요한 분야에 적용될 수 있음을 <그림 1>의 "도서 예약 시스템" 예제에 대한 시스템 분해 예제를 통해 알아볼 수 있다. 이 때, 같은 서브시스템에 속하는 클래스를 판별하는 과정은 클러스터링 알고리즘 [4] 중 널리 사용되는 WPGMA를 이용했다.



<그림 1. "도서 예약 시스템"의 클래스 다이어그램
(예약접수 클래스의 호출 함수와 피호출 함수만 표기)>

<그림 2. 예약접수 클래스의 메소드들과 다른 클래스의 메소드들 간의 호출 관계>

RCBC(예약접수, 예약처리)	= 0.33	*RCBC(예약접수, 예약오류처리)*	= 0.13
RCBC(예약접수, 사용자정보관리)	= 0.17	*RCBC(예약접수, 사용자정보)*	= 0.07
RCBC(예약접수, 도서관리)	= 0.17	*RCBC(예약접수, 도서정보)*	= 0.13
의존도(예약접수, 서브시스템1)	= (0.33+0.13)/2 = 0.23		
의존도(예약접수, 서브시스템2)	= (0.17+0.07)/2 = 0.12		
의존도(예약접수, 서브시스템3)	= (0.17+0.13)/2 = 0.15		

<그림 3. 예약접수 클래스와 다른 클래스 간의 RCBC 값과
예약접수 클래스와 서브시스템 간의 의존도 값>

<그림 1>의 클래스 다이어그램에는 총 7개의 클래스와 현재 형성 과정 중인 3개의 서브시스템이 나타나 있고 <그림 2>에는 예약접수 클래스의 메소드들과 다른 클래스에 속한 메소드들 간의 호출 관계와 피호출 함수의 SLOC가 나타나있다. <그림 3>에는 예약접수와 다른 클래스 간의 RCBC 값과 RCBC를 기반으로 한 예약접수 클래스와 각 서브시스템 간의 의존성 강도가 나타나 있으며, 이를 바탕으로 예약접수 클래스가 의존도가 0.23으로 가장 높은 서브시스템1에 속해야 한다고 판단할 수 있다. 이렇듯이 RCBC는 클래스 간의 의존성 강도를 측정함으로서 서브시스템을 판별하는 데 적용되어 유용한 정보를 제공해 줄 수 있다.

이 논문에서는 클래스 간의 의존성 강도를 측정할 수 있는 커플링 메트릭 RCBC를 유지보수성-재검사 비용적 측면에서 정의하고, RCBC를 시스템 분해 예제에 적용해 봄으로써 시스템 유지 보수를 위한 여러 분야에 사용될 수 있음을 보였다. 현재 가장 우선적인 향후 과제는 실험을 통한 RCBC의 검증이며, 이를 위해 자동으로 RCBC를 측정할 수 있는 도구를 구현하고 있으며 이 도구를 이용하여 실제 산업 시스템에 RCBC를 적용하여 그 유용성을 검증할 계획이며, 또한 시스템 분해와 같은 시스템 재구성 분야에 RCBC를 적용해 봄을 통해 RCBC의 유용성을 검증해 볼 것이다.

참고문헌

[1]L.C. Briand and J. Wuest, "Empirical Studies of Quality Models in Object-Oriented Systems," *Advances in Computers*, vol. 59, pp. 97-166, 2002.

[2]D.C. Kung, J. Gao, P. Hsia, Y. Toyoshima, and C. Chen. "On Regression Testing of Object-Oriented Software Maintenance," *The Journal of Syst. and Software*, vol. 32, no. 1, pp. 21-40, 1996.

[3]R. Leitch and E. Stroulia, "Assessing the Maintainability Benefits of Design Restructuring using Dependency Analysis," *Proc. 9th Int'l Symp. Software Metrics*, pp. 309-322, 2003.

[4]N. Anquetil, and T. C. Lethbridge, "Comparative Study of Clustering Algorithms and Abstract Representations for Software Remodularisation," *Proc. Softw.*, vol. 150, no. 3, pp. 185 - 201, 2003