컴포넌트 기반 시스템에서 클래스들 간의 정적 그리고 동적 특성을 적용한 컴포넌트 메트릭스

(Component Metrics Based on Static and Dynamic Characteristics between Classes for Component based Systems)

최 미 숙 † 이 종 석 ††
(Misook Choi) (Jongseok Lee)

요 약 컴포넌트 기반 시스템에서 재사용 단위인 컴포넌트의 품질은 시스템 개발의 성공을 위해서 가장 중요하다. 컴포넌트의 품질을 향상시키기 위해서는 개발 이전에 측정 가능해야 하고, 그 결과를 컴포넌트 개발 과정에 반영할 수 있어야 한다. 또한 컴포넌트의 품질을 정확하게 측정하여야 한다.

따라서 본 논문에서는 클래스들 간의 상호작용에 의한 정적 그리고 동적인 특성을 적용하여 식별된 컴포넌트의 품질을 좀 더 정확하게 측정할 수 있는 컴포넌트의 응집도와 결합도 메트릭스를 제안한다. 제안된 메트릭스가 이론적으로 타당하다는 것을 검증하기 위하여 Briand이 정의한 프레임워크에 적용하여 증명한다. 또한 메트릭스의 실용성을 검증하기 위해서 사례를 제시하고 기존의 메트릭스와의 비교분석을 통해서 평가 결과를 제시한다. 본 논문에서 제안한 컴포넌트 메트릭스는 식별된 컴포넌트의 품질을 좀 더 정확하게 측정함으로써 컴포넌트 설계를 위한 개발 시간과 노력을 절감한다.

키워드 : 컴포넌트의 품질, 컴포넌트 응집도, 컴포넌트 결합도, 클래스들 간의 종속성

Abstract In component-based system, the qualities of components as reusable units are the most important to success the component-based development. Therefore, before software implementation phase, the designed components should be measurable to improve the qualities of the components and the measured results should be reflected in the component-based development phase. In addition, the qualities of the components should be measured accurately.

Accordingly, this paper proposes cohesion and coupling metrics applying static and dynamic dependency characteristics by the interdependence between classes. We prove the theoretical soundness of the proposed metrics by the axiom of briand et al. A case study and a comparison with the conventional metrics verify the practicality of the proposed metrics. The development times and endeavors to design the components is reduced, because the proposed metrics measure the qualities of components accurately.

Key words : the quality of component, component cohesion, component coupling, dependency between classes

1. 서 론

컴포넌트 기반의 시스템 구축을 위하여 가장 중요한 점은 기능적으로 재사용이 가능하고 유지보수 단계의 효율적인 시스템 관리를 위해서 상호 의존성이 적은 독립적인 컴포넌트, 즉, 품질이 좋은 컴포넌트를 잘 설계하는 것이다. 품질이 좋은 컴포넌트의 설계를 돕기 위해서는 식별된 컴포넌트가 보다 독립적인지 아닌지를 정량적으로 평가할 메트릭스(Metrics)가 필요하고 컴포넌트 메트릭스(Component Metrics) 중 하나의 단위 안에 있는 요소들 간의 관계 정도를 측정하는 응집도(Cohesion)와 단위들 간의 의존도를 측정하는 결합도(Coupling)는 재사용이 가능한 독립된 부품 단위인 컴포넌트의 개발을 위해서 중요한 의미를 갖는다. 이 때 단위가 되는 컴포넌트 내의 응집도는 최대이고,. 컴포넌트 간의 결합도는 최소가 되도록 하는 것이 가장 이상적인 시스템이다[1-5].

현재, 컴포넌트 기술은 새로운 기술이 아닌 객체지향

† 정 회 원 : 우석대학교 컴퓨터공학과 교수
khc67_kr@hanmail.net
†† 종신회원 : 우석대학교 컴퓨터공학과 교수
jong1007@nate.com
논문접수 : 2005년 7월 21일
심사완료 : 2006년 1월 7일

기술을 기반으로 재사용이라는 목적을 달성하기 위해 발전된 기술이기 때문에 객체지향의 연장으로 생각하여 컴포넌트를 정량적으로 평가할 수 있는 메트릭스에 대한 연구가 미흡하다. 또한 기존의 컴포넌트 메트릭스[1-5]가 존재한다 해도 객체지향 메트릭스[6-8]를 그대로 적용하여 제시하거나 약간 수정 후 사용하는 경우가 대부분이다. 그러나 클래스가 재사용 단위가 되는 객체지향 시스템과 밀접한 클래스들의 그룹으로 이루어져 있는 컴포넌트가 재사용 단위가 되는 컴포넌트 기반 시스템은 구조적 측면과 기능의 단위적 측면에서 다르기 때문에 객체지향 메트릭스(Object-Oriented Metrics)를 그대로 컴포넌트의 품질 평가에 적용한다면 정확하게 컴포넌트의 품질을 측정할 수가 없어 올바르게 컴포넌트를 설계할 수가 없다.

따라서 본 논문에서는 컴포넌트의 구조적 측면과 기능의 단위적 측면을 고려하여 기존의 컴포넌트 메트릭스에서 적용하지 않았던 클래스와 클래스 간의 구조적 관계에 의한 정적인 종속성, 클래스와 클래스 간의 메소드 호출 유형에 따른 동적인 종속성, 그리고 메소드 호출 수의 증가에 따른 종속성의 비선형적인 증가 특성을 적용하여 식별된 컴포넌트의 품질을 좀 더 정확하게 측정할 수 있는 컴포넌트의 메트릭스인 응집도와 결합도를 제안하였다. 또한 제안된 메트릭스가 이론적으로 타당하다는 것을 검증하기 위하여 Briand et al[14]이 정의한 프레임워크(FrameWork)에 적용하여 증명하였고 본 논문에서 제안한 메트릭스가 기존의 메트릭스 보다 좀 더 정확히 컴포넌트의 품질을 평가한다는 것을 사례와 기존의 메트릭스와의 비교 분석을 통하여 제시하였다. 제안된 컴포넌트 메트릭스는 개발 이후의 측정이 아닌 개발 이전의 측정으로, 상대적으로 빨리 컴포넌트에 대한 품질을 평가하여 이후 컴포넌트 개발에 반영할 수 있다는 장점이 있어 기능적으로 보다 응집도가 높으며 결합도가 낮은 컴포넌트를 설계하므로 설계된 컴포넌트의 품질을 향상시켜 유지보수성, 재사용성이 향상된 컴포넌트 기반 시스템이 구축되도록 한다.

본 논문의 구성은 다음과 같다. 2장에서는 기존의 컴포넌트 메트릭스의 문제점을 기술한다. 3장에서는 컴포넌트의 메트릭스에 적용하기 위한 클래스들 간의 상호작용에 의한 종속적 특성을 제시한다. 4장에서는 3장에서 제시한 특성을 적용하여 컴포넌트의 응집도와 결합도 메트릭스를 제시하고 Briand이 정의한 공리에 적용하여 제시한 메트릭스가 이론적으로 타당함을 증명한다. 5장에서는 전자 상거래 시스템과 입찰관리 시스템에 본 논문에서 제시한 응집도와 결합도 메트릭스를 적용하고 기존의 메트릭스와 비교 평가를 통해서 본 논문에서 제시한 메트릭스가 좀 더 정확히 컴포넌트의 품질을 평가한다는 것을 보인다. 6장에서는 결론 및 향후 연구 과제를 제시한다.

2. 관련연구

본 장은 기존의 메트릭스 중 본 논문과 관련이 있는 응집도와 결합도 메트릭스를 중심으로 살펴본다.

Kim[1]이 제안한 컴포넌트의 응집도는 응집 결여도(LCOM)를 나타내는 메트릭스로서 컴포넌트가 포함하는 전체 클래스의 개수에서 조인트 액션(Joint Action)을 공유하는 클래스의 개수를 뺀 값을 조인트 액션의 수로 나누어 계산한다. Kim이 제안하는 결합도는 컴포넌트 간에 공유하는 조인트 액션의 수로 계산한다. Kim이 제안하는 컴포넌트의 응집도와 결합도 메트릭스는 분석 초기 단계에서 적용할 수 있는 장점은 있지만 단순히 클래스들이 공유하는 액션의 수나 조인트 액션을 공유하는 클래스의 수에 의해서만 응집도와 결합도를 평가한다. Lee[2]가 제안한 응집도는 유스케이스(Use-case)인 각 기능을 실행하기 위해서 포함된 클래스들의 중요도에 대한 가중치를 곱하여 응집도를 정의한다. 결합도는 시퀀스 다이어그램(Sequence Diagram)에서 호출된 메서드의 수를 이용해서 상호관련 동적 결합도를 측정하고 클래스 다이어그램에서 각 클래스들의 관계를 통해서 정적 결합도를 측정한 후 동적 결합도와 정적 결합도를 곱하여 클래스 간의 결합도를 측정한다. Lee가 제안한 메트릭스는 클래스들의 상호작용에 의해서 컴포넌트의 기능이 실행되는 수평적 특성은 단순히 메소드 호출 수만을 적용하고 있다. Cho[3]가 제안한 컴포넌트 메트릭스는 응집도와 결합도를 제안한 것이 아니라 복잡도 메트릭스를 제안했다. 이 방법을 이용하기 위해서는 소프트웨어 설계자들은 어플리케이션 도메인(Application Domain)과 시스템 종류에 대해서 확고한 지식을 가지고 있어야 하고 컴포넌트 메트릭스이기는 하나, 객체지향적 메트릭스를 그대로 적용하고 있다.

다음은 객체지향 메트릭스 중 Henderson-Sellers가 제안한 결합도[7]는 MPC(Message Passing Coupling)이다. MPC는 클래스 내에 정의된 send 문장의 수로 정의된다. 응집도는 Chidamber[11]가 제안한 LCOM(Lack of Cohesion in Methods)을 확장하여 LCOM*를 정의하였는데 클래스가 포함하는 메소드의 기능을 실행하기 위해서 공유하는 속성(Attribute)의 수가 많을수록 응집도가 증가한다.

따라서 기존의 컴포넌트의 결합도를 보면 단순히 메소드 호출(method call) 수, 컴포넌트 간에 공유하는 오퍼레이션(operation)의 수나 기능(function)의 수를 사용하여 결합도를 정의한다. 또한 응집도를 보면 각 유스케이스의 기능을 실행하기 위해서 얼마나 컴포넌트가 포

함하는 클래스들을 공유하고 있는가를 가지고 응집도를 정의한다. 즉, 기존의 컴포넌트의 응집도와 결합도는 객체지향 메트릭스와 거의 유사하다고 볼 수 있다.

그러나 클래스의 구조적 특성과 컴포넌트의 구조적 특성이 다르기 때문에 객체지향 메트릭스를 컴포넌트의 메트릭스로 그대로 적용하여 컴포넌트의 응집도와 결합도를 측정할 경우 정확하게 컴포넌트의 품질을 측정할 수 없다. 즉, 클래스는 메소드와 속성으로 구성되어 있고 클래스가 포함하고 있는 메소드의 수행은 클래스 내부의 속성을 참조하여 수행하거나 클래스 간의 메소드 호출에 의하여 기능이 실행된다. 그러나 컴포넌트는 기능 및 구조적으로 밀접한 연관성이 있는 클래스들을 그룹화 하여 독립적인 단위로 개발되어지고 실행된다. 따라서 컴포넌트는 기능을 실행하기 위해서 컴포넌트 내부나 외부의 클래스들의 객체를 생성(create), 삭제(delete), 수정(update) 그리고 참조(read)하여 컴포넌트의 기능을 수행한다. 따라서 컴포넌트의 기능을 실행하기 위해서 컴포넌트 내부적 측면에서는 컴포넌트가 포함하고 있는 클래스들의 상호 교류에 의해서 기능을 실행하기 때문에 클래스간의 상호 교류에 의한 종속성을 고려하여야 한다. 컴포넌트의 외부적 측면에서는 각 컴포넌트가 포함하고 있는 클래스에 의한 컴포넌트 간의 상호 교류에 의해서 기능이 실행된다. 그러므로 컴포넌트 내부나 외부의 클래스들의 상호 교류에 의한 종속적 특성을 부여하는 컴포넌트의 구조적 특성을 반영하여야 좀 더 정확하게 컴포넌트의 품질을 측정할 수 있다.

따라서 본 논문에서는 기존의 컴포넌트 메트릭스에서 적용하지 않았던 컴포넌트가 포함하고 있는 클래스들의 상호 교류에 의한 클래스 간의 종속적 특성과 메소드 호출 수에 의한 클래스 간의 비 선형적 증가 특성을 정의하고 그러한 특성을 컴포넌트의 품질 측정에 반영하여 좀 더 정확하게 컴포넌트의 품질을 측정하는 컴포넌트의 응집도와 결합도 메트릭스를 정의한다.

3. 클래스들 간의 종속적 특성

본 장에서는 좀 더 정확하게 컴포넌트의 품질을 평가하기 위해 클래스들 간의 종속적 특성을 분류하였다. 즉, 클래스와 클래스 간의 구조적 관계에 의한 정적인 종속성, 클래스와 클래스 간의 메소드 호출 유형에 따른 동적인 종속성, 그리고 메소드 호출 수의 증가에 따른 종속성의 비선형적인 증가 특성으로 분류하였다.

특성 1. 클래스 간의 구조적 관계에 의한 종속적 강도는 포함관계 > 상속관계 > 연관관계 순이고 클래스 간의 관계가 포함관계나 상속관계라면 그들 사이에 강한 종속관계가 존재한다.

클래스들의 구조적 관계는 포함관계, 상속관계 그리고 연관관계가 존재한다. 클래스들의 관계가 포함관계라면, 즉, A 클래스가 B 클래스와 C 클래스를 포함하고 있다면 A 클래스의 수정은 B클래스와 C 클래스에 영향이 그대로 전파되고 또한 A 클래스에 대한 객체의 연산은 B 클래스에 대한 객체와 C 클래스에 대한 객체들에 그대로 전파된다. 예를 들어 A 클래스에 대한 객체가 생성, 삭제, 수정 그리고 참조 되어진다면 동시에 B 클래스와 C 클래스에 대한 객체에 똑같은 연산이 전파되어진다. 또한 포함관계이므로 A 클래스의 기능을 수행하기 위해서는 반드시 B 클래스와 C 클래스가 존재해야 하고 라이프 사이클 역시 같다. 클래스들의 관계가 상속관계라면, 즉, A 클래스가 부모 클래스이고 자식 클래스가 B 클래스라면 B 클래스는 A 클래스의 멤버 변수와 메소드를 상속 받기 때문에 A 클래스의 수정은 B 클래스에 영향이 그대로 전파된다. 따라서 포함관계와 상속관계의 클래스들은 상위 클래스의 수정이나 상위 객체의 수정은 하위 클래스나 객체들에 직접적으로 강한 영향을 전파함으로 수직적으로 주종관계를 이루며 강하게 결합되어있고 종속되어있다고 정의할 수 있다. 그러나 상속관계는 포함관계처럼 A 클래스와 B 클래스가 반드시 라이프 사이클이 같거나 B 클래스가 존재하지 않는다 해서 A 클래스가 완전히 존재할 수 없는 것은 아니기 때문에 상속관계 보다 포함관계가 더욱 더 강한 종속관계를 갖는다고 정의할 수 있다.

A클래스와 B 클래스 사이에 연관관계가 존재한다면 A 클래스와 B 클래스 사이에 메시지 호출에 의한 수평적 관계이다. 연관관계에 있는 클래스들은 메소드의 호출 유형에 따라서 종속의 관계가 달라진다.

특성 2. 클래스 간의 관계가 연관관계라면 메시지 호출 유형인 생성, 삭제 > 수정 > 참조의 순으로 클래스 간의 종속적 관계의 강도가 다르다.

클래스들 간의 의존은 메소드 호출 유형에 따라서 달라진다. 메소드 호출 유형에 의해서, 클래스 간의 의존의 강도를 평가하기 위해서 본 논문에서는 메소드 호출 유형을 다음과 같이 분류한다. 즉 <A, B> ∈ R의 관계에서 메시지의 유형은 다음과 같다.

① A 클래스에 대한 객체가 B 클래스에 대한 객체를 생성하는 메소드를 호출할 경우
② A 클래스에 대한 객체가 B 클래스에 대한 객체를 삭제하는 메소드를 호출할 경우
③ A 클래스에 대한 객체가 B 클래스에 대한 객체를 수정하는 메소드를 호출할 경우
④ A 클래스에 대한 객체가 B 클래스에 대한객체를 참조하는 메소드를 호출할 경우
⑤ A 클래스에 대한 객체가 B 클래스에 대한 객체에게 ①,②,③,④의 경우 등을 포함하는 메소드를 호

출할 경우

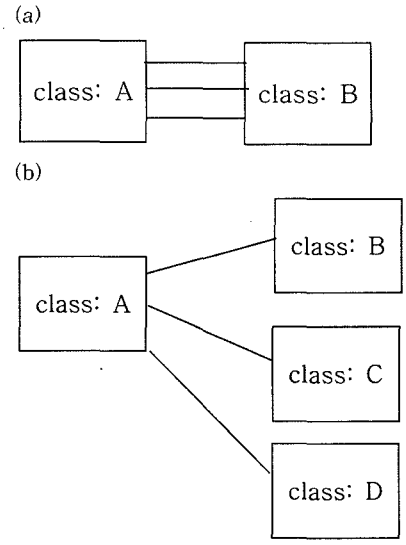
①과 ②의 경우는 A 클래스에 대한 객체에 의해서 B 클래스에 대한 객체의 구조가 변경되므로 B 클래스의 객체를 참조하여 기능을 수행하는 다른 객체들은 전체적으로 구조적 측면에서 영향을 받는다. 또한 컴포넌트는 기능을 실행하는 하나의 독립적인 단위가 되므로 기능을 실행하기 위해서 필요한 데이타를 생성하거나 삭제하는 클래스는 핵심적인 클래스가 된다. 따라서 클래스 간에 데이타를 생성하거나 삭제하는 메소드를 호출하는 관계의 클래스들은 서로 강한 종속관계를 가지고 있다고 정의할 수 있다. ③의 경우는 ①과 ②의 경우처럼 구조가 변경되는 것이 아니라 객체의 값이 변경되므로 B 객체를 참조하여 기능을 수행하는 다른 객체들은 기능을 수행할 수 없는 것이 아니라 기능은 수행하되 값의 변경만 이루어진다. ④의 경우는 ①과 ②의 경우처럼 구조가 변경되는 것도 아니고 ③의 경우처럼 값의 변경이 이루어지는 것도 아니므로 B 클래스의 객체를 참조하여 기능을 수행하는 다른 객체들은 전혀 영향을 받지 않는다. 따라서 ④의 경우 클래스 A의 객체가 a 컴포넌트에 그리고 클래스 B의 객체가 b컴포넌트에 포함되어 a 컴포넌트의 클래스 A의 객체가 b컴포넌트의 클래스 B의 객체를 참조하는 메시지를 보낸다면 ①과 ②의 경우처럼 구조가 변경되는 것도 아니고 ③의 경우처럼 값의 변경이 이루어지는 것이 아니므로 b 컴포넌트 내의 클래스 B의 객체를 참조하여 기능을 수행하는 다른 객체들은 전혀 영향을 받지 않으므로 두 컴포넌트 간의 종속관계는 ①, ② 그리고 ③의 경우 보다 가장 약한 의존관계를 가지고 있다고 정의할 수 있다. ⑤의 경우는 ①,②,③,④ 중에 포함된 메소드의 호출 유형에 따라서 달라진다.

특성 3. 클래스 간에 양 방향의 메소드 호출이 존재한다 해도 메소드 호출 유형에 따라서 종속 관계의 강도가 달라진다.

A 클래스에 대한 객체가 B 클래스에 대한 객체에게 메소드를 호출할 경우 A 클래스의 객체가 B 클래스의 객체에게 어떠한 유형의 메소드를 호출하느냐에 따라 의존의 강도가 달라진다. 또한 기존에는 무조건 양방향의 메소드 호출 관계가 존재한다면 강한 종속관계를 가지고 있다고 정의[8]하였으나 본 논문에서는 양 방향의 메소드 호출관계가 존재하더라도 메소드 호출 유형에 따라서 의존의 강도가 달라진다.

특성 4. 클래스들 간의 메소드 호출 수에 의한 클래스 들 간의 연결 강도는 비선형적으로 증가한다.

클래스들 사이의 상호작용은 메소드 호출에 의해서 이루어진다. 따라서 클래스들 간의 상호작용에 의한 연결강도는 메소드 호출 수와 메소드 호출 타입에 따라서 달라진다. 그러나 클래스 간의 메소드 호출 수의 증가에 따라서 클래스들 간의 연결 강도는 반드시 선형적으로 증가되지는 않는다. 그 이유는 다음과 같다.



a)의 경우, A 클래스는 오직 한 클래스인 B 클래스와 3번의 메소드 호출이 일어났고 b)의 경우, A 클래스는 각각 3개의 클래스에 대하여 각각 한번씩의 메소드 호출이 일어났다. 그러나 A 클래스와 다른 클래스와의 상호작용 관계에서 메소드 호출 수는 같다고 해도 의미적으로 a)의 경우에 A 클래스의 결합도 보다 b)의 경우의 A 클래스의 결합도가 더 높다고 볼 수 있으므로 클래스들 사이의 메소드 호출 수의 증가에 따라서 클래스들 간의 연결 강도는 반드시 선형적으로 증가되지는 않는다.

4. 컴포넌트의 메트릭스

본 장에서는 클래스들 간의 관계적 특성을 적용한 컴포넌트의 응집도와 결합도 메트릭스를 정의하고 제안된 메트릭스가 타당하다는 것을 Briand이 정의한 프레임워크를 통하여 이론적으로 증명한다. 본 논문에서 제안하는 메트릭스는 기존에 적용하지 않았던 컴포넌트의 특성을 좀 더 정확하게 반영하여 설계된 컴포넌트의 품질을 좀 더 정확하게 평가한다.

4.1 컴포넌트의 응집도와 결합도 메트릭스

컴포넌트의 설계가 바람직하다면, 컴포넌트 구성요소인 클래스 사이에는 밀접한 관련성이 높은 응집도를 가지며, 다른 컴포넌트와의 상호작용은 적어 낮은 결합도를 가지는 컴포넌트가 될 것이다. 따라서 본 절에서는 3장에서 제시한 클래스들 간의 상호작용에 의한 관계적

특성을 적용하여 컴포넌트의 응집도와 결합도 메트릭스를 정의한다.

4.1.1 추상화 모델의 정의

본 절은 컴포넌트의 응집도와 결합도 메트릭스를 정의하기 위해서 필요한 컴포넌트에 대한 추상화 모델을 정의한다.

정의 1. 시스템을 구성하는 컴포넌트

시스템은 유한개의 컴포넌트들로 구성되어 있으므로 컴포넌트 기반 시스템을 S라하고 포함된 컴포넌트들을 $BC_i(i=1...l)$라 하면, 시스템은 다음과 같이 정의한다.

$$S=\{BC_1, BC_2, ..., BC_l\}$$

이때, 컴포넌트 기반 시스템의 전체 컴포넌트의 수는 l 개 이다.

정의 2. 컴포넌트의 구성 요소

시스템을 구성하고 있는 컴포넌트 $BC_i(i=1...l)$는 유한개의 클래스 C로 구성되어 있으므로 컴포넌트 BC_i $(i=1...l)$는 다음과 같이 정의한다.

$$BC_i = \{C_{i1}, C_{i2}, ..., C_{im}\}$$

정의 3. 클래스가 포함하는 메소드

컴포넌트는 클래스들의 집합으로 이루어져 있고 클래스들 간의 상호작용이나 컴포넌트 간의 상호작용은 각 클래스가 포함하고 있는 메소드들의 집합에 의해서 이루어진다. 따라서 각 클래스가 포함하고 있는 메소드 $M(C_j)(j=1...k)$는 다음과 같이 정의한다.

$$M(C_j)=\{m_{j1}, m_{j2}, ..., m_{jn}\}$$

정의 4. 클래스 간의 메소드 호출 방향

클래스 간의 상호작용은 메소드 호출에 의해서 이루어진다. 서로 다른 클래스 $C_g(1 \le g \le m)$, $C_y(1 \le y \le m)$에 대해서 메소드 $m' \in M(C_g), m \in M(C_y)$이 존재한다고 할 때 메소드 m'이 m을 호출한다면 (m', m)으로 정의한다.

정의 5. 클래스 간의 상호 작용

클래스들 간의 상호 작용은 클래스들 사이의 메소드 호출에 의해서 이루어진다. 서로 다른 클래스 $C_g(1 \le g \le m)$, $C_y(1 \le y \le m)$에 대해서 클래스 C_g와 C_y간의 메소드 호출에 의한 상호작용은 다음과 같이 정의한다.

$$Calling(C_g, C_y)=\{m \in M(C_y) \mid \text{for some } g \neq y \; \exists m' \in M(C_g) \; s.t. \; (m', m)\}$$

$$Called(C_g, C_y)=\{m' \in M(C_g) \mid \text{for some } g \neq y \; \exists m \in M(C_y) \; s.t. \; (m, m')\}$$

메소드 호출 유형인 생성(Create), 삭제(Delete), 수정(Write), 참조(Read)에 의한 클래스 간의 상호작용은 다음과 같이 정의한다.

a. 클래스 C_g와 C_y간의 메소드 호출이 생성일 경우

$$Calling(C_g, C_y)=:C(C_g, C_y),$$
$$Called(C_g, C_y)=:C(C_y, C_g)$$

b. 클래스 C_g와 C_y간의 메소드 호출이 삭제일 경우

$$Calling(C_g, C_y)=:D(C_g, C_y),$$
$$Called(C_g, C_y)=:D(C_y, C_g)$$

c. 클래스 C_g와 C_y간의 메소드 호출이 수정일 경우

$$Calling(C_g, C_y)=:W(C_g, C_y),$$
$$Called(C_g, C_y)=:W(C_y, C_g)$$

d. 클래스 C_g와 C_y간의 메소드 호출이 참조일 경우

$$Calling(C_g, C_y)=:R(C_g, C_y),$$
$$Called(C_g, C_y)=:R(C_y, C_g)$$

정의 6. 메소드 호출 유형에 의한 가중치

메소드 호출 유형에 따라서 클래스 간에 의존의 강도가 달라지므로 메소드 호출 유형에 따라서 가중치(W)를 부여한다. 따라서 클래스 간의 메소드 호출 유형에 따른 가중치를 다음과 같이 정의한다.

- 클래스 간의 메소드 호출 유형이 생성일 경우 : W_c
- 클래스 간의 메소드 호출 유형이 삭제일 경우 : W_d
- 클래스 간의 메소드 호출 유형이 수정일 경우 : W_w
- 클래스 간의 메소드 호출 유형이 참조일 경우 : W_r

클래스들 간의 관계적 특성에 의한 종속의 강도는 생성, 삭제>수정>참조 순이다. 따라서 클래스 간의 메소드 유형에 의한 가중치의 크기는 W_c , $W_d > W_w > W_r$ 의 순으로 부여한다.

4.1.2 클래스 연관도

본 논문에서 제안하는 클래스 간의 연관도는 클래스들이 얼마나 긴밀하게 연관되어 있는지를 나타낸다. 따라서 클래스들 사이에 메소드 호출 수를 고려하고 메소드 호출 수에 따른 클래스 간의 연결 강도에 대한 비선형적 증가 특성을 고려한다. 또한 컴포넌트의 기능의 단위적 특성에 의하여 메소드 호출 유형인 C, D, W, R에 따라서 클래스 간의 의존의 정도가 달라지므로 메소드 호출 유형인 C, D, W, R 에 따라서 W_c , $W_d > W_w > W_r$ 의 가중치를 적용한다. 따라서 클래스의 연관도인 CC(Class Coherency)는 클래스 C_g와 C_y에 대해서 다음과 같이 정의한다.

정의 7. 클래스의 연관도(Class Coherency)

$$CC(C_g, C_y)=\sum_{i=1}^{NIC} W_c^i+\sum_{j=1}^{NID} W_d^j+\sum_{k=1}^{NIW} W_w^k+\sum_{l=1}^{NIR} W_r^l$$

where:

클래스 C_g와 C_y 간에 메소드 호출 유형에 대한 메소드 호출 수 다음과 같이 정의한다.

$$NIC(Nmber\ of\ Invocation\ for\ Creation) = |C(C_g, C_y)| + |C(C_y, C_g)|$$

$$NID(Nmber\ of\ Invocation\ for\ Delete) = |D(C_g, C_y)| + D(C_y, C_g)|$$

$$NIW(Nmber\ of\ Invocation\ for\ Write) = |W(C_g, C_y)| + W(C_y, C_g)|$$

$$NIR(Nmber\ of\ Invocation\ for\ Read) = |R(C_g, C_y)| + |R(C_y, C_g)|$$

4.1.3 컴포넌트의 응집도(Component CoHesion)

컴포넌트 내의 클래스들은 서비스를 제공하기 위해서 서로 긴밀하게 연관되어 있어야 한다. 따라서 클래스들 간의 긴밀도를 정의한 클래스 연관도를 이용하여 응집도를 측정한다. 응집도를 정량적으로 측정하기 위한 컴포넌트의 응집도(Component CoHesion)를 비즈니스 컴포넌트 BC_h에 대해서 다음과 같이 정의하였다.

정의 8. 컴포넌트의 응집도(Component CoHesion)

$$CCH(BC_h) = \begin{cases} \dfrac{\sum_{C_{hi}, C_{hj} \in BC_h} CC(C_{hi}, C_{hj})}{{}_{|C|}C_2 \times MAX(W)}, & \text{if } |C| > 1 \\ 1, & \text{if } |C| = 1 \end{cases}$$

where:

$CC(C_{hi}, C_{hj})$: 클래스 간의 연결 강도

${}_{|C|}C_2$: 컴포넌트 BC_k내에 존재하는 클래스가 관계를 가지는 수

$MAX(W)$: 클래스 간의 메소드 호출에 의한 최대 가중치, 즉,

$CC(C_g, C_y) = \sum_{i=1}^{\ni C} W^i{}_c + \sum_{j=1}^{\ni D} W^j{}_d + \sum_{k=1}^{\ni W} W^k{}_w + \sum_{l=1}^{\ni R} W^l{}_r$ 라고 할 때

$$MAX(W) = \sum_{i=1}^{\infty} W_c^i + \sum_{j=1}^{\infty} W_d^j + \sum_{k=1}^{\infty} W_w^k + \sum_{l=1}^{\infty} W_r^l = \frac{W_c}{1-W_c} + \frac{W_d}{1-W_d} + \frac{W_w}{1-W_w} + \frac{W_r}{1-W_r}$$

따라서, 만약 가중치를 $W_c, W_d = 0.4 > W_w = 0.15 > W_r = 0.05$ 라고 하면

$$MAX(W) = \frac{0.4}{1-0.4} + \frac{0.4}{1-0.4} + \frac{0.15}{1-0.15} + \frac{0.05}{1-0.05} = 1.563$$

이다.

4.1.4 컴포넌트의 결합도(Component Coupling)

유지보수가 용이하기 위해서는 컴포넌트 간의 의존도가 적어야 한다. 따라서 컴포넌트 간의 상호 의존도를 정량적으로 측정하기 위한 컴포넌트의 결합도(Component CouPling)를 컴포넌트 BC_k에 대하여 다음과 같이 정의하였다.

정의 9. 컴포넌트의 결합도(Component CouPling)

$$CCP(BC_k) = \sum_{C_{ki} \in BC_k, C_{nj} \notin BC_k} CC(C_{ki}, C_{nj})$$

where: $CC(C_{ki}, C_{nj})$: 클래스 간의 연결 강도

4.2 컴포넌트 메트릭스의 이론적 검증

Briand[14]는 특정 소프트웨어 산출물에만 적용되지 않도록 일반적이며, 정확한 수학적 개념에 근거하여 엄격한 수학적 공리를 제안했다. 이 공리는 메트릭스에 대해 크기(size), 길이(length), 복잡도(complexity), 응집도(cohesion), 결합도(coupling)등의 분야에 대한 개념과 성질을 정의하였다. 또한 새로운 소프트웨어 메트릭스를 정의할 때 만족해야할 기준을 제공해 주고 있다. 본 논문에서는 컴포넌트의 재사용 측면과 유지보수 측면에서 타당한 컴포넌트가 식별되었는지를 정량적으로 측정하고 평가하기 위한 컴포넌트의 응집도와 결합도 메트릭스를 정의하였다. 따라서 본 절에서는 제안된 컴포넌트 메트릭스의 이론적 타당성을 검증하기 위하여 Briand이 정의한 프레임워크에 적용하여 증명한다.

4.2.1 응집도

응집도 속성에 대한 이론적 검증은 다음과 같다.

응집도 성질 1: Non-negativity and Normalization
(응집도는 음수 값을 가져서는 안 되며, 항상 일정한 구간 내에 존재 하도록 정규화 되어야 한다.)
$(\forall BC_k)(0 \le CCH(BC_k) \le 1)$

증명. 컴포넌트 BC_k에 대한 응집도 $CCH(BC_k)$는 컴포넌트 BC_k의 내부 클래스들 간의 메시지 호출 수에 따른 메시지 호출 유형의 가중치의 합으로 정의하였으므로 비 음수성을 가지며, 분모는 컴포넌트 BC_k의 내부 클래스들의 모든 쌍의 개수와 내부 클래스들 간에 메시지 호출의 수에 따른 메시지 호출 유형의 가중치의 최대 값의 곱으로 계산되고 분자는 내부 클래스들 간의 메시지 호출 수에 따른 메시지 호출 유형의 가중치의 합으로 정의하였으므로 컴포넌트 BC_k에 대한 응집도 $CCH(BC_k)$는 0과 1 사이의 값으로 정규화 된다.

$$CC(C_g, C_y) = \sum_{i=1}^{NIC} W_c^i + \sum_{j=1}^{NID} W_d^j + \sum_{k=1}^{NIW} W_w^k + \sum_{l=1}^{NIR} W_r^l$$

$$\le \sum_{i=1}^{\infty} W_c^i + \sum_{j=1}^{\infty} W_d^j + \sum_{k=1}^{\infty} W_w^k + \sum_{l=1}^{\infty} W_r^l =: MAX(W)$$

이다. 따라서 컴포넌트 내부에 존재하는 클래스들 간의 연관도의 합에 대해서는

$$\sum_{C_{ki}, C_{kj} \in BC_k} CC(C_{ki}, C_{kj}) \le {}_{|C|}C_2 \times MAX(W)$$

을 얻는다.

그러므로,

$$0 \le CCH(BC_k) = \frac{\sum_{C_{hi}, C_{hj} \in BC_h} CC(C_{hi}, C_{hj})}{{}_{|C|}C_2 \times MAX(W)} \le 1$$

이 성립한다.

응집도 성질 2: Null Value

(컴포넌트 내부 구성요소들 사이에 상호작용이 없다면, 응집도는 0이라는 것을 의미한다.)

(for *some* BC_k, $\exists C_{ki}, C_{kj}$)
($CC(C_{ki}, C_{kj}) = 0 \rightarrow CCH(BC_k) = 0$)

증명. 컴포넌트 내부의 클래스들 사이에 상호작용이 없다면 $CC(C_{ki}, C_{kj}) = 0$이다. 따라서 응집도의 분자 값은 0이 된다. 그러므로 응집도의 값이 0이 된다.

$$CCH(BC_k) = \frac{\sum_{C_{ki}, C_{kj} \in BC_k} CC(C_{ki}, C_{kj})}{|C|C_2 \times MAX(W)} \text{ 에 대해서}$$

$$CCH(BC_k) = \frac{0}{|d|C_2 \times MAX(W)} \text{ 이기 때문에}$$

$CCH(BC_k) = 0$ 이 성립한다.

응집도 성질 3: Monotonicity

(컴포넌트 내부 구성 요소들 사이의 양적 변화 없이 상호작용만 증가한다면 응집도는 감소하지 않는다는 것을 의미한다.)

(for *some* BC_k, $\exists C_{ki}, C_{kj}$)
($CC(C_{ki}, C_{kj}) \le CC'(C_{ki}, C_{kj}) \rightarrow CCH(BC_k) \le CCH'(BC_k)$)

증명. 컴포넌트 내부 구성요소들 사이의 양적 변화가 없다는 것은 컴포넌트가 포함하는 클래스의 개수가 변하지 않았다는 것을 의미하므로 응집도의 분모 값은 그대로 존재한다. 그러나 컴포넌트 내부 구성요소들 사이의 상호작용이 증가한다는 것은 컴포넌트 내부 구성요소인 클래스들 사이의 상호 관계가 증가된다는 것이므로 응집도의 분자 값은 증가한다는 것이다. 그러므로 응집도의 값은 감소하지 않는다.

$CC(C_{ki}, C_{kj}) \le CC'(C_{ki}, C_{kj})$ 라 하면

$\sum_{C_{ki}, C_{kj} \in BC_k} CC(C_{ki}, C_{kj}) \le \sum_{C_{ki}, C_{kj} \in BC_k} CC'(C_{ki}, C_{kj})$이고

$\frac{\sum_{C_{ki}, C_{kj} \in BC_k} CC(C_{ki}, C_{kj})}{|C|C_2 \times MAX(W)} \le \frac{\sum_{C_{ki}, C_{kj} \in BC_k} CC'(C_{ki}, C_{kj})}{|C|C_2 \times MAX(W)}$이므로

$CCH(BC_k) \le CCH'(BC_k)$이 성립한다.

응집도 성질 4: Cohesive Components

(컴포넌트 사이에 상호작용이 전혀 없는 두 컴포넌트를 합친다면 응집도는 원래 두 컴포넌트의 최대 응집도보다 크지 않다는 것을 의미한다.)

(for *some* BC_i, BC_j, BC_k)
($BC_i = BC_j \bigcup BC_k$ and $CC(BC_j, BC_k) = 0 \rightarrow \max\{CCH(BC_j), CCH(BC_k)\} \ge CCH(BC_i)$)

증명. 컴포넌트 사이에 상호작용이 전혀 없는 두 컴포넌트를 합친다면 컴포넌트 내부 구성요소들 사이의 상호작용의 관계는 컴포넌트를 합치기 전이나 합친 후나 같으므로 응집도의 분자 값은 그대로 있으나 두 컴포넌트를 하나로 합친다면 컴포넌트 내부의 구성 요소의 수, 즉, 클래스의 개수가 늘어나므로 응집도의 분모 값이 증가한다. 따라서 컴포넌트를 합친 응집도는 합치기 전의 최대 응집도 보다 증가하지 않는다는 것을 의미한다.

$$CCH(BC_j) = \frac{\sum_{C_{ja}, C_{jb} \in BC_j} CC(C_{ja}, C_{jb})}{|C|C_2 \times MAX(W)}$$

$$=: \frac{A}{|C|C_2 \times MAX(W)} \text{ 라 하고}$$

$$CCH(BC_k) = \frac{\sum_{C_{ki}, C_{kj} \in BC_k} CC(C_{ki}, C_{kj})}{|C'|C_2 \times MAX(W)}$$

$$=: \frac{B}{|C'|C_2 \times MAX(W)} \text{라 하자.}$$

$BC_i = BC_j \bigcup BC_k$ and $CC(BC_j, BC_k) = 0$ 이기 때문에,

$$CCH(BC_i) = \frac{\sum_{C_{ja}, C_{jb} \in BC_j} CC(C_{ja}, C_{jb}) + \sum_{C_{ki}, C_{kj} \in BC_k} CC(C_{ki}, C_{kj})}{|C+C'|C_2 \times MAX(W)}$$

이고, 위의 정의에 의하여,

$$\frac{\sum_{C_{ja}, C_{jb} \in BC_j} CC(C_{ja}, C_{jb}) + \sum_{C_{ki}, C_{kj} \in BC_k} CC(C_{ki}, C_{kj})}{|C+C'|C_2 \times MAX(W)}$$

$$= \frac{A+B}{|C+C'|C_2 \times MAX(W)} \text{ 이다.}$$

다음에서 $\max\{CCH(BC_j), CCH(BC_k)\} \ge CCH(BC_i)$을 증명한다.

$\max\{CCH(BC_j), CCH(BC_k)\}$에 의하여

$CCH(BC_j) \ge CCH(BC_k)$라 가정하면 위의 정의에 의하여, $\frac{A}{|C|C_2 \times MAX(W)} \ge \frac{B}{|C'|C_2 \times MAX(W)}$ 이다.

만약 $\alpha \ge 0$라 하면,

$$\frac{A}{|C|C_2 \times MAX(W)} = \frac{B+\alpha}{|C'|C_2 \times MAX(W)} \text{ 이 존재하고}$$

$$\frac{A}{|C|C_2 \times MAX(W)} = \frac{B+\alpha}{|C'|C_2 \times MAX(W)}$$

$$= \frac{A+(B+\alpha)}{(|C|C_2 + |C|C_2) \times MAX(W)} \text{ 이다.}$$

따라서,

$$\frac{A+(B+\alpha)}{(|C|C_2 + |C|C_2) \times MAX(W)} \ge \frac{A+B}{(|C|C_2 + |C|C_2) \times MAX(W)} \ge$$

$$= \frac{A+B}{|C+C'|C_2 \times MAX(W)}$$

$$= \frac{\sum_{C_{ja}, C_{jb} \in BC_j} CC(C_{ja}, C_{jb}) + \sum_{C_{ki}, C_{kj} \in BC_k} CC(C_{ki}, C_{kj})}{|C+C'|C_2 \times MAX(W)}$$

$= CCH(BC_i)$이다.

그러므로 $\max\{CCH(BC_j), CCH(BC_k)\} \geq CCH(BC_i)$ 이 성립한다.

4.2.2 결합도

결합도 속성에 대한 이론적 검증은 다음과 같다.

결합도 성질 1: Non-Negativity

(결합도의 측정값은 음수가 될 수 없다는 것을 의미한다.)

$(\forall BC_k)(CCP(BC_k) \geq 0)$

증명. 컴포넌트 간에 메시지 호출관계가 없다면 0이고 메시지 호출관계가 하나라도 존재한다면 결합도는 항상 0 보다 크다.

$CCP(BC_k) = \sum_{C_{ki} \in BC_k, C_{nj} \notin BC_k} CC(C_{ki}, C_{nj}) \geq 0$이 성립한다.

결합도 성질 2: Null Value

(컴포넌트들 사이에 상호작용이 없다면 0이라는 것을 의미한다.)

(for *some* BC_k , $\exists C_{ki} \in BC_k, C_{nj} \notin BC_k$)
$(CC(C_{ki}, C_{nj}) = 0 \rightarrow CCP(BC_k) = 0)$

증명. 컴포넌트 간에 호출관계가 존재하지 않는다면 $CC(C_{ki}, C_{nj}) = 0$이므로 결합도는 0이다.

$CCP(BC_k) = \sum_{C_{ki} \in BC_k, C_{nj} \notin BC_k} CC(C_{ki}, C_{nj})$에 대해서

$CC(C_{ki}, C_{nj}) = 0$이므로 $\sum_{C_{ki} \in BC_k, C_{nj} \notin BC_k} CC(C_{ki}, C_{nj}) = 0$

이다.

$CCP(BC_k)$이 성립한다.

결합도 성질 3: Monotononicity

(컴포넌트 내부 구성요소에는 변화 없이, 컴포넌트들 사이의 상호작용만 증가한다면, 결합도는 감소하지 않는다는 것을 의미한다.)

(for *some* BC_k, $\exists C_{ki} \in BC_k, C_{nj} \notin BC_k$)
$(CC(C_{ki}, C_{nj}) \leq CC'(C_{ki}, C_{nj}) \rightarrow$
$CCP(BC_k) \leq CCP'(BC_k))$

증명. 컴포넌트 간에 상호작용이 증가한다는 것은 컴포넌트 간에 메시지 호출이 증가한다는 것으로 컴포넌트의 결합도는 그만큼 증가한다는 것이다. 따라서 결합도는 감소하지 않는다는 것을 의미한다.

$CC(C_{ki}, C_{nj}) \leq CC'(C_{ki}, C_{nj})$ 라 하면

$\sum_{C_{ki} \in BC_k, C_{nj} \notin BC_k} CC(C_{ki}, C_{nj}) \leq \sum_{C_{ki} \in BC_k, C_{nj} \notin BC_k} CC'(C_{ki}, C_{nj})$이므로

$CCP(BC_k) \leq CCP'(BC_k)$이 성립한다.

결합도 성질 4: Merging of Components

(두 컴포넌트를 결합하여 새로운 컴포넌트를 만든다면 새로운 컴포넌트의 결합도는 두 컴포넌트의 결합도의 합보다 증가하지 않는다는 것을 의미한다.)

(for *some* BC_i, BC_j, BC_k)
$(BC_i = BC_j \bigcup BC_k$ and $CC(BC_j, BC_k) \neq 0 \rightarrow$
$\sum CCH(BC_j), CCH(BC_k) \geq CCH(BC_i))$

증명. 두 개의 컴포넌트를 병합한 후의 결합도는 각각의 컴포넌트의 결합도의 합에서 두 개의 컴포넌트 간에 메시지 호출에 의한 상호관계의 값을 뺀 값이 되므로 각각의 컴포넌트의 결합도의 합보다는 병합된 컴포넌트의 결합도가 크지 않다는 것을 의미한다.

$$
\begin{aligned}
CCH(BC_i) &= \sum_{C_i \in BC_i, C_j \notin BC_i} CC(C_i, C_j) \\
&= \sum_{C_i \in BC_j \bigcup BC_k, C_j \notin BC_j \bigcup BC_k} CC(C_i, C_j) \\
&= \sum_{C_i \in BC_j, C_j \notin BC_j \bigcup BC_k} CC(C_i, C_j) \\
&\quad + \sum_{C_i \in BC_k, C_j \notin BC_j \bigcup BC_k} CC(C_i, C_j) \\
&\leq \sum_{C_i \in BC_j, C_j \notin BC_j} CC(C_i, C_j) \\
&\quad + \sum_{C_i \in BC_k, C_j \notin BC_k} CC(C_i, C_j) \\
&= CCH(BC_j) + CCH(BC_k) \text{ 이므로}
\end{aligned}
$$

$\sum\{CCH(BC_j), CCH(BC_k)\} \geq CCH(BC_i)$이 성립한다.

결합도 성질 5: Disjoint Component Aditivity

(컴포넌트 간 상호작용이 전혀 없는 두 컴포넌트를 결합하여 새로운 컴포넌트를 만든다면, 결합도는 원래 두 컴포넌트의 결합도의 합과 같다는 것을 의미한다.)

(for *some* BC_i, BC_j, BC_k)
$(BC_i = BC_j \bigcup BC_k$ and $CC(BC_j, BC_k) = 0 \rightarrow$
$\sum\{CCH(BC_j), CCH(BC_k)\} = CCH(BC_i))$

증명. 두 개의 컴포넌트 간에 상호작용이 전혀 없다면 $DCC(BC_j, BC_k) = 0$이 되므로 두 개의 컴포넌트를 병합한 후의 결합도는 각각의 컴포넌트의 결합도의 합과 같다는 것을 의미한다.

$BC_i = BC_j \bigcup BC_k$ 이고 $CC(BC_j, BC_k) = 0$ 라면

$\sum_{C_i \in BC_j, C_j \notin BC_j \bigcup BC_k} CC(C_i, C_j) = \sum_{C_i \in BC_j, C_j \notin BC_j} CC(C_i, C_j)$ 이고

$\sum_{C_i \in BC_k, C_j \notin BC_j \bigcup BC_k} CC(C_i, C_j) = \sum_{C_i \in BC_k, C_j \notin BC_k} CC(C_i, C_j)$이다.

그러므로,

$$
\begin{aligned}
&\sum_{C_i \in BC_j, C_j \notin BC_j \bigcup BC_k} CC(C_i, C_j) + \sum_{C_i \in BC_k, C_j \notin BC_j \bigcup BC_k} CC(C_i, C_j) \\
&= \sum_{C_i \in BC_j, C_j \notin BC_j} CC(C_i, C_j) + \sum_{C_i \in BC_k, C_j \notin BC_k} CC(C_i, C_j)
\end{aligned}
$$

따라서,

$CCH(BC_i) = CCH(BC_j) + CCH(BC_k)$가 성립한다.

5. 사례연구 및 비교 평가

본 장에서는 전자 상거래 시스템과 입찰관리 시스템

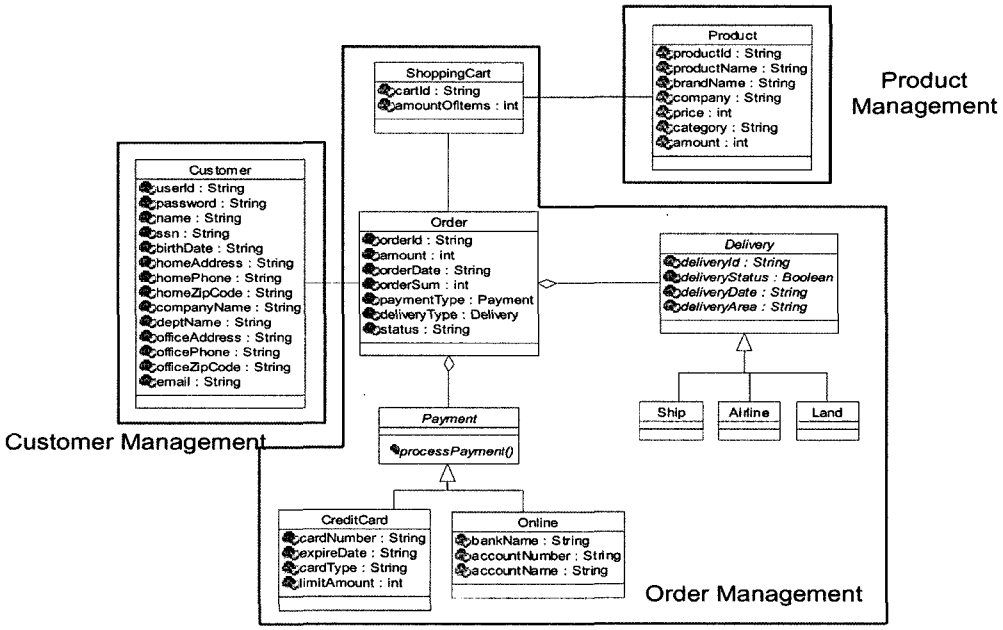


그림 1 클래스 다이어그램과 식별된 컴포넌트

을 선택하여 본 논문에서 제안한 컴포넌트 메트릭스 적용사례와 실험 결과를 제시하고 기존의 메트릭스와의 비교 평가를 통해서 그 효과성을 검증한다.

5.1 사례연구

본 절은 전자 상거래 어플리케이션의 도메인 모델에 대하여 본 논문에서 제안한 컴포넌트 메트릭스를 이용하여 도출된 컴포넌트의 응집도와 결합도를 측정하였다.

위의 그림 1에서 제시되듯이 전자상거래 도메인 모델로부터 도출된 컴포넌트는 CM(Customer Management), OM(Order Management), PM(Product Management) 3개이다. 또한 각각의 컴포넌트가 포함하고 있는 클래스 즉, Customer Management 컴포넌트는 Customer 클래스를 포함하고 있고, Order Management 컴포넌트는 ShoppingCart 클래스, Order 클래스, Payment 클래스, Delivery 클래스를 포함하고 있다. 그리고 Product Management 컴포넌트는 Product 클래스를 포함하고 있다.

컴포넌트는 유스케이스 다이어그램, 시퀀스 다이어그램 그리고 클래스 다이어그램으로부터 유스케이스를 실행하기 위해서 밀접하게 연관된 클래스들의 그룹으로 컴포넌트가 도출된다. 따라서 유스케이스의 기능을 실행하기 위해서 클래스 간에 어떠한 메소드를 호출하는지, 메소드 호출 방향 그리고 메소드 호출 수의 분석이 필요하다. 따라서 우리는 각각의 유스케이스의 기능을 실행하기 위해서 각각 어떠한 클래스가 필요하고 각각의 클래스는 어떠한 메소드를 호출하는지를 분석하기 위하여 유스케이스의 시나리오를 통해서 또한 시나리오를 구체화한 시퀀스 다이어그램을 통하여 분석하였다. 즉, 전자상거래 도메인에서 유스케이스의 기능을 실행하기 위해서 클래스 간에 어떠한 메소드를 호출하는지, 메소드 호출 방향 그리고 메소드 호출 수를 분석하여 다음 표 1에 제시하였다.

표 1 클래스들 간의 상호 작용에 의한 메소드 호출 유형과 메소드 호출 수

열 Class / 행 클래스	Customer(A)	Order(B)	Payment(C)	Delivery(D)	Shopping Cart(E)	Product(F)
Customer(A)						
Order(B)	R(9) W(2)		C(1), D(1) W(1), R(1)	R(2), W(3) D(2), C(1)	R(2) D(2)	
Payment(C)						
Delivery(D)						
Shopping Cart(E)						R(3)
Product(F)						

위의 표 1에서 A는 Customer 클래스, B는 Order 클래스, C는 Payment Class, D는 Delivery 클래스, E는 Shopping Cart 클래스, F는 Product Class로 명명하고 C는 Create, D는 Delete, W는 Write 그리고 R은 Read로 한다. 메시지 호출 방향은 행 클래스 -> 열 클래스이고 셀 내부는 메시지 호출 유형(개수)를 표시하였다. 예를 들면 표 1의 3행 2열에서 메시지 호출 방향은 Order -> Customer이고 메시지 호출 유형(개수)는 Read(9), Write(2)를 의미한다. 본 논문의 3장의 특성 2에 의하여 클래스 간의 수평적 관계인 연관 관계에서 메시지 호출 유형에 따라서 클래스 간의 의존의 정도가 달라지므로 메시지 호출 유형인 생성.삭제 > 수정 > 조회 순으로 가중치의 크기를 달리하였다. 메시지 호출 유형에 따른 가중치는 C=0.4, D=0.4, W=0.15, R=0.05를 부여하여 계산하였다. 또한 본 논문의 3장의 특성 1에 의하여 클래스 다이어그램에서 Order 클래스는 Delivery 클래스와 Payment 클래스를 포함하는 포함관계로 강한 종속 관계가 존재하므로 클래스 간의 연결 강도의 최대 값인 $MAX(W)=1.563$을 부여하여 계산한다. 따라서 Order 클래스와 Delivery 클래스 사이의 클래스 연관도 = 1.563, Order 클래스와 Payment 클래스 사이의 클래스 연관도 = 1.563을 부여하여 계산하였다.

다음은 본 논문에서 제안한 컴포넌트 메트릭스를 전자 상거래 시스템의 컴포넌트에 적용하여 클래스의 연관도, 결합도, 응집도를 계산한 결과이다.

① 클래스의 연관도

$$CC(Order, Shopping\ Cart)=\sum_{i=1}^{2}0.05^{i}+\sum_{j=1}^{2}0.4^{j}=0.053+0.56=0.61$$

$$CC(Order, Payment)=1.563$$

$$CC(Order, \nabla ivery)=1.563$$

② 컴포넌트의 응집도

$$CCH(Order\ Management)=\frac{0.61+1.563+1.563}{{}_4C_2\times MAX(W)}=\frac{3.74}{6\times 1.563}=0.4$$

③ 컴포넌트의 결합도

$$CCP(Customer\ \ Management)=\sum_{i=1}^{9}0.05^{i}+\sum_{j=1}^{2}0.15^{j}=0.053+0.17=0.223$$

$$CCP(Order\ \ Management=\sum_{i=1}^{9}0.05^{i}+\sum_{j=1}^{2}0.15^{j}+\sum_{k=1}^{3}0.05^{k}=0.053+0.17+0.0526=0.275$$

$$CCP(\mathrm{Product}\ \ Management)=\sum_{j=1}^{3}0.05^{j}=0.0526=0.0526$$

④ 컴포넌트의 응집도와 결합도의 측정 결과

시스템을 구성하는 컴포넌트들은 응집도가 높고 결합도가 낮은 컴포넌트들로 구성이 되어있을 때 보다 독립적인 컴포넌트들로 잘 식별되었다고 할 수 있다. 표 2에서 제시한 측정 결과를 보면 각각의 컴포넌트는 응집도가 높고 결합도가 낮음을 알 수 있다.

표 2 컴포넌트의 응집도와 결합도의 측정 결과

컴포넌트	Customer Management	Order Management	Product Management
응집도	1	0.4	1
결합도	0.223	0.275	0.0526

5.2 비교 분석 및 평가

본 절은 KIM이 제안한 컴포넌트 메트릭스, Hander-son Sellers의 객체지향 메트릭스 그리고 본 논문에서 제안한 메트릭스를 전자상거래 시스템과 입찰관리 시스템에 적용하여 측정 결과를 정량적으로 비교 평가하였으며 기존 메트릭스와의 정성적 비교 분석을 통해서 그 효과성을 검증하였다.

다음은 전자상거래 시스템의 클래스 다이어그램에서 클래스들의 연결 관계에 의해 클래스들을 가능한 모든 경우로 조합하여 후보 컴포넌트들을 추출하였고 각 후보 컴포넌트들의 응집도와 결합도를 측정하여 그 결과를 비교 분석하였다.

① KIM의 컴포넌트 메트릭스

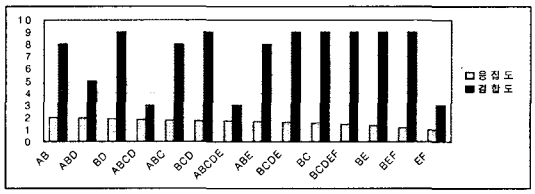


그림 2 Kim의 컴포넌트 응집도와 결합도

② Handerson Sellers의 객체지향 메트릭스

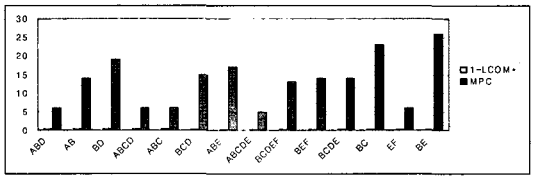


그림 3 Handerson Sellers의 컴포넌트 응집도와 결합도

③ 제안한 메트릭스

Kim이 제안한 컴포넌트의 응집도는 조인트 액션을

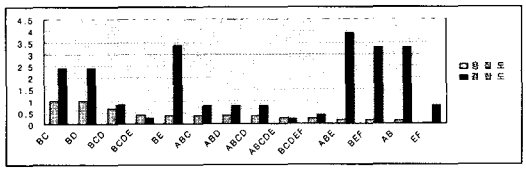


그림 4 제안된 컴포넌트의 응집도와 결합도

공유하는 클래스의 개수가 많을수록 응집도가 높아지고 결합도는 컴포넌트 간에 공유하는 조인트 액션의 수가 많으면 많을수록 결합도가 증가한다. 또한 Henderson sellers에 의한 객체지향 메트릭스는 각 기능을 실행하기 위해서 공유하는 클래스의 개수가 많을수록 컴포넌트의 응집도는 증가하고 결합도는 메시지 호출 수에 비례하여 결합도가 증가함을 알 수 있다. 따라서 위에서 제시한 그래프를 보면 Kim의 경우 가장 응집도가 높고 결합도가 낮은 형태는 ABCD 클래스들을 그룹화한 컴포넌트이고 Henderson sellers의 경우는 ABCDE 클래스를 그룹화한 컴포넌트이다. 즉, Kim과 Henderson sellers의 경우 가장 메시지 호출이 많은 A(Customer) 클래스와 B(Order) 클래스를 포함한 컴포넌트가 가장 응집도가 높고 결합도가 낮다는 것을 확인할 수 있었다. 또한 B(Order) 클래스와 C(Payment) 클래스, B(Order) 클래스와 D(Delivery) 클래스가 포함관계이기 때문에 B(Order) 클래스와 C(Payment) 클래스 그리고 D(Delivery) 클래스가 포함된 컴포넌트가 응집도가 높아야 하는데 그런 부분은 전혀 고려되어지지 않고 있다. 그리고 전체적으로 컴포넌트의 응집도와 결합도의 결과는 실제 식별된 컴포넌트의 결과와 일치하지 않는다는 것을 알 수 있다. 본 논문에서 제안한 메트릭스로 평가한 컴포넌트의 응집도와 결합도의 결과를 보면 높은 응집도를 가진 컴포넌트는 BC, BD, BCD, BCDE 순이다. 이 때 BC와 BD 그리고 BCD는 포함관계이기 때문에 당연히 이 클래스들을 포함한 컴포넌트는 응집도가 높아야하는 것은 당연하므로 이 결과는 우리의 직관과 일치한다. 또한 가장 응집도가 높고 결합도가 낮은 컴포넌트는 BCDE이므로 B, C, D, E의 클래스들을 포함한 컴포넌트가 식별되어야 한다. 이는 개발자들에 의해서 식별된 컴포넌트의 결과와 일치한다. 이 결과는 메시지 호출 유형에 따른 클래스 간의 상호작용에 의한 종속관계를 가중치로 부여하여 측정하였기 때문이다. 결론적으로 클래스는 속성과 메소드로 구성되어 있고 클래스가 포함하고 있는 메소드의 수행은 속성을 참조하여 수행한다. 즉, 속성과 속성끼리의 상호 교류에 의해서 메소드의 기능이 실행되는 것이 아니라 단순히 존재하는 속성을 참조하여 수행한다. 따라서 객체지향 메트릭스는 단순히 메소드의 기능을 실행하기 위해서 얼마나 많은 속성을 공유하느냐에 따라서 클래스의 응집도를 측정해도 되지만 컴포넌트는 밀접하게 연관된 클래스들의 그룹으로 되어있고 컴포넌트의 기능은 각 클래스들의 상호 교류에 의해서 수행된다. 따라서 앞에서도 살펴보았듯이 기존의 컴포넌트 메트릭스나 객체지향 메트릭스의 응집도는 클래스간의 상호 교류에 의한 종속성을 전혀 고려하지 않고 단순히 기능을 실행하기 위해서 참조되는 클래스의 개수로만 측정하고 있으므로 정확히 측정되어지지 않음을 알 수 있었다. 컴포넌트의 결합도는 컴포넌트 내부가 클래스 간의 상호 작용에 의해서 기능을 수행하듯이 컴포넌트 간에도 각 컴포넌트가 포함하고 있는 클래스 간의 상호작용에 의해서 이루어지고 있으므로 컴포넌트 간의 결합도 역시 클래스 간의 메소드 호출 유형에 의한 종속적 특성을 적용해야 한다.

다음은 A 사에서 개발된 입찰관리 시스템에 대하여 그 결과를 측정하였다. 입찰관리 시스템은 32개의 유스케이스(Usecase)와 15개의 클래스(Class)에 대해서 5개의 컴포넌트(Component)가 추출되었다. 입찰관리 시스템의 15개의 클래스들 중에서 상속관계나 포함관계에 있는 클래스들은 8개였고 전자 상거래 시스템이나 입찰관리 시스템에서 상속관계나 포함관계에 있는 클래스들은 전체 클래스의 약 50% 정도를 차지하였다. 따라서 기존의 메트릭스 중에서 상속관계나 포함관계의 특성을 고려하지 않은 메트릭스는 정확히 컴포넌트를 평가할 수 없다는 것을 알 수 있다. 또한 전자상거래 시스템에서도 제시했듯이 클래스 간의 상호작용에 의한 클래스

표 3 클래스 간의 상호작용 관계에 의한 메시지 호출 수와 클래스 간의 연관도

클래스간의 관계	메소드 호출 수	클래스의 관계	클래스 간의 연관도
(A1, A2)	4	1.563	1.563
(A1, A3)	4	1.563	1.563
(B1, B2)	3	1.563	1.563
(B1, B3)	3	1.563	1.563
(B2, B4)	3	1.563	1.563
(B3, B5)	5	R(3), W(1), C(1)	0.6525
(E1, E2)	5	1.563	1.563
(E1, E3)	5	1.563	1.563
(E1, E4)	5	1.563	1.563
(E1, E5)	5	C(2), W(1), R(2)	0.785
(A1, B1)	4	R(4)	0.0526
(B3, E4)	1	W(1)	0.15
(B4, E4)	1	W(1)	0.15
(B1, E1)	3	R(3)	0.0526
(E1, F1)	3	R(3)	0.0526
(B2, E1)	2	R(2)	0.0525
(B4, E1)	1	R(1)	0.05
(E1, G1)	1	R(1)	0.05

표 4 입찰관리 시스템의 컴포넌트

추출된 컴포넌트	포함된 클래스
컴포넌트1(A)	A1, A2, A3
컴포넌트2(B)	B1, B2, B3, B4, B5
컴포넌트3(E)	E1, E2, E3, E4, E5
컴포넌트4(F)	F1
컴포넌트5(G)	G1

간의 연관도는 단순히 메소드 호출 수만으로 정확히 측정될 수 없음을 알 수 있다. 따라서 입찰관리 시스템에 대하여 클래스 간의 메시지 호출 수와 본 논문에서 제시한 클래스 간의 연관도 비교를 통해서 컴포넌트가 좀 더 정확히 평가되어짐을 다음 표 4의 실험 결과를 통해서 제시한다.

위의 표 4에서 보면 단순히 메소드 호출 수만으로 클래스 간의 상속관계나 포함관계의 특성을 고려할 수 없고 메소드 호출 수에 비례해서 클래스 간의 연관도의 강도가 증가되어지지 않는다는 것을 확인할 수 있다. 또한 본 논문에서 제안한 클래스 간의 정적 그리고 동적인 종속적 특성을 적용한 클래스 연관도의 강도에 비례하여 연관도가 높은 클래스들끼리 그룹화 되어 컴포넌트로 추출되었음을 확인할 수 있다. 따라서 그 결과는 우리의 직관과 일치한다. 또한 클래스의 연관도에 의하여 컴포넌트의 응집도가 측정되므로 클래스의 연관강도가 높은 것끼리 그룹화한 컴포넌트일수록 응집도가 높음을 알 수 있다. 또한 위의 그림 5에서 제시하듯이 식별된 컴포넌트는 응집도는 높고 결합도는 낮게 산출되었음을 알 수 있다. 객체지향 메트릭스나 기존의 컴포넌트 메트릭스는 기능을 실행하기 위해서 얼마나 많은 클래스를 공유하느냐에 따라서 컴포넌트의 응집도를 측정한다. 그러나 컴포넌트는 밀접하게 연관된 클래스들의 그룹으로 되어 있고 컴포넌트의 기능은 각 클래스들의 상호 교류에 의해서 수행되기 때문에 컴포넌트가 포함하고 있는 클래스간의 상호 교류에 의한 종속성을 전혀 고려하지 않는다면 컴포넌트의 내부의 클래스들이 상호 교류에 의해서 얼마나 밀접하게 연관되었는지를 평가할 수가 없는 문제점이 존재한다. 즉, 그림 5와 같이 단순

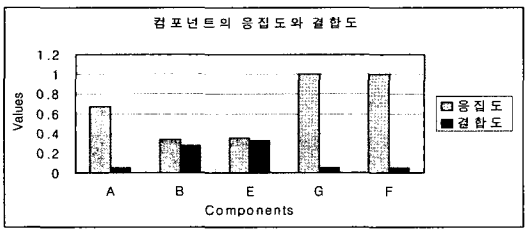


그림 5 입찰관리 시스템의 각 컴포넌트의 응집도와 결합도

히 컴포넌트의 응집도와 결합도를 수치로만 측정한다면 컴포넌트의 내부의 각각의 클래스들이 클래스 간의 상호교류에 의해서 얼마나 밀접하게 연관 되었는지를 확인할 수 없다.

또한, 본 논문에서 제안한 응집도와 결합도 메트릭스를 적용하였을 경우 시스템이 타당한 컴포넌트로 분할이 잘 됨을 검증하기 위하여 우리는 프로그램을 구현하였다. 즉, 시스템에 포함된 클래스들을 조합하여 모든 경우의 후보 컴포넌트들을 추출하고 추출된 후보 컴포넌트들을 조합하여 가능한 모든 경우의 후보 시스템을 추출했다. 그리고 추출된 후보 시스템들에 대하여 본 논문에서 제안한 응집도와 결합도 메트릭스를 적용하여 후보 시스템의 평균 응집도와 평균 결합도를 측정하였다. 그 결과 가장 응집도가 높고 결합도가 낮은 후보 컴포넌트들로 구성된 후보 시스템의 결과가 개발자들의 직관과 경험에 의하여 식별된 컴포넌트들의 결과와 일치하는지 3개의 사례를 가지고 실험해 보았다.

먼저 그림 1에서 제시한 전자상거래 시스템에 적용하여 실험해 보았다. 6개의 클래스에 대하여 가능한 경우의 후보 컴포넌트의 수는 29개가 추출되었으며, 추출된 후보 컴포넌트들을 조합하여 추출된 모든 경우의 후보 시스템의 수는 31개가 추출되었다. 그리고 응집도와 결합도의 차가 높은 후보 시스템 안에 포함된 후보 컴포넌트들의 결과는 개발자의 직관과 경험에 의하여 식별된 컴포넌트의 결과와 일치함을 확인할 수 있었다. 그 결과는 다음 표 5, 6, 7에 제시한다.

다음 표 5는 클래스 다이어그램에서 클래스 간의 연결강도를 계산하여 제시하였다.

표 5 전자상거래 시스템의 클래스 연관도(Class Coherency)

클래스 간의 관계	클래스 간의 연관도
CC(A:Customer, B:Order)	0.225132
CC(B:Order, C:Payment)	1.563000
CC(B:Order, D:Delivery)	1.563000
CC(B:Order, E:Shopping Cart)	0.612500
CC(E:Shopping Cart, F:Product)	0.052625

다음 표 6은 클래스들을 조합하여 가능한 모든 경우의 후보 컴포넌트들을 추출한 후 추출된 후보 컴포넌트 각각의 응집도를 계산하여 제시하였다.

다음 표 7은 표 6에서 제시한 후보 컴포넌트들을 중복되지 않게 조합하여 가능한 모든 경우의 후보 시스템들을 추출한 후 후보 시스템이 포함하고 있는 후보 컴포넌트들의 응집도의 평균, 그리고 후보 컴포넌트 간의 결합도의 평균을 구한 후 응집도와 결합도의 차를 계산하여 제시하였다.

표 6 전자상거래 시스템의 후보 컴포넌트들과 그들의 응집도

후보 컴포넌트	포함된 클래스의 수	클래스들의 조합의 수	분모(1.563*클래스들 간의 조합의 수)	분자(클래스 간의 연관도의 합)	응집도 (분자/분모)
A	1	1	1.000000	1.000000	1.000000
B	1	1	1.000000	1.000000	1.000000
C	1	1	1.000000	1.000000	1.000000
D	1	1	1.000000	1.000000	1.000000
E	1	1	1.000000	1.000000	1.000000
F	1	1	1.000000	1.000000	1.000000
AB	2	1	1.563000	0.225132	0.144038
BC	2	1	1.563000	1.563000	1.000000
BD	2	1	1.563000	1.563000	1.000000
BE	2	1	1.563000	0.612500	0.391875
EF	2	1	1.563000	0.052625	0.033669
ABC	3	3	4.689000	1.788132	0.381346
ABD	3	3	4.689000	1.788132	0.381346
ABE	3	3	4.689000	0.837632	0.178638
BCD	3	3	4.689000	3.126000	0.666667
BCE	3	3	4.689000	2.175500	0.463958
BDE	3	3	4.689000	2.175500	0.463958
BEF	3	3	4.689000	0.665125	0.141848
ABCD	4	6	9.378000	3.351132	0.357340
ABCE	4	6	9.378000	2.400632	0.255985
ABDE	4	6	9.378000	2.400632	0.255985
ABEF	4	6	9.378000	0.890257	0.094930
BCDE	4	6	9.378000	3.738500	0.398646
BCEF	4	6	9.378000	2.228125	0.237591
BDEF	4	6	9.378000	2.228125	0.237591
ABCDE	5	10	15.63000	3.963632	0.237591
ABCEF	5	10	15.63000	2.453257	0.156958
ABDEF	5	10	15.63000	2.453257	0.156958
BCDEF	5	10	15.63000	3.791125	0.242554

표 7 후보 시스템들의 응집도, 결합도 그리고 응집도와 결합도의 차

후보 시스템 (후보 컴포넌트들)	후보 시스템의 평균 결합도	후보 시스템의 평균 응집도	응집도-결합도
S_1(A, BCDE, F)	0.138878	0.799549	0.660670
S_2(A, BCD, E, F)	0.296752	0.916667	0.619914
S_3(ABCDE, F)	0.052625	0.626796	0.574171
S_4(ABCD, E, F)	0.332563	0.785780	0.453217
S_5(A, BCDEF)	0.225132	0.621277	0.396146
S_6(A, BC, D, E, F)	0.613314	1.000000	0.386686
S_7(A, BD, C, E, F)	0.613314	1.000000	0.386686
S_8(A, BCE, D, F)	0.613586	0.865990	0.252404
S_9(A, BDE, C, F)	0.613586	0.865990	0.252404
S_{10}(A, B, C, D, E, F)	0.803215	1.000000	0.196749
S_{11}(A, BCD, EF)	0.418816	0.566779	0.147963
S_{12}(ABC, D, E, F)	0.742708	0.845337	0.102628
S_{13}(ABD, C, E, F)	0.742708	0.845337	0.102628
S_{14}(A, BE, C, D, F)	0.850939	0.878375	0.027436
S_{15}(A, BC, D, EF)	0.800211	0.758417	-0.041793
S_{16}(A, BD, C, F)	0.800211	0.758417	-0.041793
S_{17}(ABCE, D, F)	0.807812	0.751995	-0.055817
S_{18}(ABDE, C, F)	0.807812	0.751995	-0.055817
S_{19}(AB, C, D, E, F)	0.947781	0.828808	-0.118974
S_{20}(A, BCEF, D)	0.894066	0.745864	-0.148202
S_{21}(A, BDEF, C)	0.894066	0.745864	-0.148202
S_{22}(A, B, C, D, EF)	0.990908	0.806734	-0.184174
S_{23}(ABE, C, D, F)	1.059542	0.794659	-0.264882
S_{24}(A, BEF, C, D)	1.117044	0.785462	-0.331582
S_{25}(ABCD, EF)	0.612500	0.195504	-0.416996
S_{26}(ABC, D, EF)	1.087750	0.471672	-0.616078
S_{27}(ABD, C, EF)	1.087750	0.471672	-0.616078
S_{28}(AB, C, D, EF)	1.246167	0.544427	-0.701740
S_{29}(ABEF, C, D)	1.563000	0.698310	-0.864690
S_{30}(ABCEF, D)	1.563000	0.578479	-0.984521
S_{31}(ABDEF, C)	1.563000	0.578479	-0.984521

표 8 기존 메트릭스와의 정성적 비교 평가

메트릭스 / 비교 요소	객체지향 메트릭스	기존의 컴포넌트 메트릭스	제안된 메트릭스
메소드 호출 수	고려됨	고려됨	고려됨
메소드 호출 방향	고려되지 않음	고려되지 않음	고려됨
메소드 호출의 유형	고려되지 않음	고려되지 않음	고려됨
클래스 간의 정적인 종속성	부분적으로 고려됨	부분적으로 고려됨	고려됨
클래스 간의 동적인 종속성	부분적으로 고려됨	부분적으로 고려됨	고려됨
메소드 호출에 의한 클래스 간의 비선형적 증가에 따른 종속성	고려되지 않음	고려되지 않음	고려됨
측정을 위한 고려 범위	클래스	컴포넌트	컴포넌트
컴포넌트의 구조적 특성	고려되지 않음	부분적으로 고려됨	고려됨
컴포넌트의 기능적 재사용 단위	고려되지 않음	고려되지 않음	고려됨
컴포넌트 식별 결과의 타당성	보통	보통	높다
응집도의 정규화	부분적으로 고려됨	부분적으로 고려됨	고려됨

위의 표 7에서 제시하고 있듯이 응집도와 결합도의 차가 가장 높은 후보 시스템이 포함한 후보 컴포넌트들은 개발자의 직관과 경험에 의해서 식별된 컴포넌트들의 결과와 일치하다는 것을 알 수 있고 전체 후보 시스템들의 결과 역시 타당함을 확인할 수 있었다.

또한 온라인 북 주문 시스템[19]에 적용한 결과 응집도와 결합도의 차가 가장 높은 후보 시스템이 포함한 후보 컴포넌트들은 개발자의 직관과 경험에 의해서 식별된 컴포넌트들의 결과와 정확히 일치하다는 것을 확인하였다. 그리고 표 3에서 제시한 입찰관리 시스템에 적용한 결과 역시 타당한 결과가 산출되어짐을 우리는 확인할 수 있었다.

위의 표 8은 몇 가지의 비교 요소들에 의하여 본 논문에서 제안된 메트릭스와 기존의 메트릭스의 비교 결과를 정성적으로 평가한다.

6. 결 론

본 논문에서는 기존의 컴포넌트 메트릭스에서 적용하지 않았던 클래스와 클래스 간의 정적 그리고 동적인 종속성과 메소드 호출 수에 의한 클래스 간의 비 선형적인 증가 특성을 적용하여 식별된 컴포넌트의 품질을 좀 더 정확하게 측정할 수 있는 컴포넌트의 응집도와 결합도 메트릭스를 제안하였다. 제안한 메트릭스를 Briand이 정의한 프레임워크에 적용하여 본 논문에서 정의한 응집도와 결합도 메트릭스가 이론적으로 타당하다는 것을 증명하였다. 또한 메트릭스의 실용성을 검증하기 위해서 실 사례인 전자상거래 시스템, 입찰관리 시스템그리고 온라인 북 주문 시스템에 적용하여 기존의 메트릭스와의 비교 분석을 통해서 본 논문에서 제안한 메트릭스가 좀 더 정확하게 정량적으로 측정됨을 확인하였다. 본 논문에서 제안한 응집도와 결합도 메트릭스는 컴포넌트 분석단계나 설계단계에서 식별된 컴포넌트를 정량적으로 평가함으로써 좀 더 독립적인 컴포넌트를 설계하는데 개발 시간과 노력을 절감할 수 있다.

참 고 문 헌

[1] H. H. Kim and D. W. Bae, "Component Identification via Concept Analysis," Journal of Object Oriented Programming, 2001.

[2] J. K. Lee, S. J. Jung and S. D. Kim, "Component Identification Method with Coupling and Cohesion," Proceedings of Asia-Pacific Software Engineering Conference, pp.79-88, 2001.

[3] E. S. Cho, M. S. Kim and S. D. Kim, "Component Metrics to Measure Component Quality," Proceedings of Asia-Pacific Software Engineering Conference, pp.419-426, 2001.

[4] 고병선, 박재년, "컴포넌트 메트릭스를 이용한 컴포넌트 설계 재정비", 한국 정보과학회 논문지:소프트웨어 및 응용, Vol. 31, No. 8, pp.980-990, 2004.

[5] 이은주, 신우창 et al, "객체지향 모델로부터 정적 메트릭을 이용하여 컴포넌트 기반 시스템으로 변환하는 기법", 한국 정보과학회 논문지:소프트웨어 및 응용, Vol. 31, No. 6 pp.728-740, 2004.

[6] S.R. Chidamber and C.F. Kemerer, "A Metric Suite for Object-Oriented Design," IEEE Transactions on Software Engineering, vol. 17, No. 6, pp.636-638, 1994.

[7] Henderson-Sellers, Brian, *Object-Oriented Metrics*, Prentice-Hall, 1996.

[8] Mark Lorenz, Jeff Kidd, *Object-Oriented Software Metrics : A Practical Guide*, Prentice-Hall, 1994.

[9] Lionel Briand, Sandro Morasca, Victor Basili, "Property-based Software Engineering Measurement," IEEE Transactions on Software Engineering, Vol.22, No.1, pp.68-86, 1996.

[10] David C. Kung, Jerry. Gao, Pei Hsia, F. Wem, Y. Toyoshima and C. Chen, "Change Impact Identification in Object Oriented Software Maintenance," Proceedings International Technical Conference on

Ciecuit/Systems, Computers and Communications, 1999.
[11] Hirohisa AMAN, Hiroyuki YAMADA, Matu-Tarow NODA, and Torao YANARU, "A Graph-Based Class Structural Complexity Metric and Its Evaluation," IEICE Trans. Information & System, Vol.E85-D, NO.4, pp.674-684, 2002.
[12] E. J. Weyuker, "Evaluation Software Complexity Measures," IEEE Transactions on Software Engineering, Vol.14, No.9, pp.1357-1365, 1988.
[13] N. Fenton, "Software measurement: A necessary scientific basis," IEEE Transactions on Software Engineering, Vol.20, No.3, pp.199-206, 1994.
[14] 최미숙, 이종석, 송행숙, "컴포넌트 설계를 위한 결합도 메트릭", 한국정보처리학회 논문지:D, Vol.12-D, No. 4, pp.609-616, 2005, 8.

최 미 숙

1990년 전북대학교(이학사). 1994년 숙명여자대학교 일반대학원 컴퓨터과학과(이학석사). 2002년 숙명여자대학교 일반대학원 컴퓨터과학과(이학박사). 1995년~1999년 나주대학 소프트웨어 개발과 전임교수. 2004년~현재 우석대학교 컴퓨터공학과 초빙 교수. 관심분야는 소프트웨어 개발 방법론, CBD 방법론, 소프트웨어 아키텍처, 소프트웨어 메트릭

이 종 석

1988년 2월 서울대학교 계산통계학과 졸업(이학사). 1990년 8월 서울대학교 대학원 계산통계학과 졸업(이학석사). 2001년 2월 서울대학교 대학원 전기컴퓨터공학부 졸업(공학박사). 1993년 3월~현재 우석대학교 컴퓨터공학과 부교수. 관심분야는 객체지향 소프트웨어 공학, 컴포넌트기반 개발, 소프트웨어 메트릭