Journal of Korea Multimedia Society Vol. 7, No. 7, July 2004(pp. 1009-1016)

분석 클래스 간의 종속적 특성을 적용한 시스템 컴포넌트 기반의 비즈니스 컴포넌트 식별

최미숙[†], 조은숙[††], 하종성[†††]

요 약

시대의 환경적 변화에 따른 소프트웨어 개발의 발달은 빠른 개발과 높은 생산성을 향상시키기 위한 소프트웨어의 재사용 기술의 확산으로 컴포넌트 기반 개발 방법론이 널리 사용되기 시작했다. 이러한 컴포넌트 기반 개발에서 재사용 가능한 독립적인 컴포넌트의 식별은 컴포넌트 기반 시스템 구축을 위하여 가장 중요한 작업이다. 컴포넌트 식별 방법을 제시하고 있는 기존 방법론들에서는 비즈니스 컴포넌트를 식별함에 있어서 개발자의 경험적 토대를 기반으로 독립적인 컴포넌트를 식별하도록 제시하고 있으므로 평이한 개발자에 의한 비즈니스 컴포넌트 식별이 쉽지 않은 문제점을 가지고 있다. 따라서 본 논문에서는 시스템 컴포넌트를 먼저 식별한 후 비즈니스 컴포넌트를 식별하고 비즈니스 컴포넌트를 식별하기 위하여 분석 클래스 간의 메소드 호출 유형과 메소드 호출 방향에 의한 클래스 간의 종속적 특성과 의존의 강도를 부여하여 효율적으로 컴포넌트를 식별할 수 있는 기준과 방법을 제안한다. 또한 사례 연구를 통하여 시스템 컴포넌트를 중심으로 비즈니스 컴포넌트가 효율적으로 식별됨을 검증한다.

Business Component Identification Based on System Component Applying Dependency Characteristics between Analysis Classes

Choi Mi-Sook[†], Cho Eun-Sook[††], Ha Jong-Sung[†††]

ABSTRACT

Component-based development is being generalized as the spread of software reuse technology for rapid development productivity and high quality software.In the CBD, the identification of independent and reusable component is the one of important tasks for component-based system development. Because existing methodologies providing component identification techniques provide techniques based on heuristic techniques of component developer, it is difficult for general developers to identify components using these methods. Therefore, this paper suggests new identification factors and a technique by considering dependency characteristics according to method call types and method call directions and dependency degree. Furthermore, proposed technique is verified through case study; business components based on system components are identified effectively.

Key words: Business Component Identification(비즈니스 컴포넌트 식별), Component Identification Method(컴포넌트 식별방법), Dependency between Classes(클래스 간의 종속성)

※ 교신저자(Corresponding Author) : 최미숙, 주소 : 전북 완주군 삼례읍 490번지(565-701), 전화 : 063)246-3465, FAX : 063)290-1453, E-mail : khc67_kr@hanmail.net
접수일 : 2004년 4월 22일, 완료일 : 2004년 6월 17일
[†] 정회원, 우석대학교 컴퓨터공학과 강사
[††] 동덕여자대학교 정보학부 강의전임교수 (E-mail : escho@dongduk.ac.kr)
[†††] 정회원, 우석대학교 컴퓨터공학과 교수 (E-mail : jsha@core.woosuk.ac.kr)

1. 서 론

컴포넌트란 소프트웨어의 한 단위로써 독립적으로 개발, 배포되어질 수 있고 요구되는 시스템 구성을 위해서 다른 컴포넌트와 연결되어질 수 있는 소프트웨어의 응집력 있는 패키지이고 일정한 기능을 수행할 수 있는 실체화의 단위이어야 한다[1]. 이러한 컴포넌트 기반의 시스템 구축을 위하여 가장 중요한 점은 기능적으로 재사용이 가능하고 유지보수 단계의 효율적인 시스템 관리를 위해서 상호 의존성이 적은 독립적인 컴포넌트를 잘 식별하는 것이다. 또한 이러한 재사용이 가능한 독립적인 컴포넌트를 경험이 적은 평이한 개발자라 하더라도 효율적으로 식별할 수 있도록 식별 방법과 식별 기준을 잘 정의하는 것이다. 그러나 컴포넌트 기반의 다양한 CASE 도구들이 소프트웨어 개발 방법론을 지원하고 있고 컴포넌트를 구현할 수 있는 기술은 제시되어 있지만 컴포넌트 개발을 위한 컴포넌트 식별 방법에 대한 연구는 미흡한 상태이다[2]. 즉, 컴포넌트를 식별하기 위한 기존 방법론들은 재사용 가능한 독립적인 컴포넌트를 식별하기 위하여 개발자의 직관과 경험에 의하여 컴포넌트를 식별한다. 또한 컴포넌트 아키텍쳐는 시스템 컴포넌트와 시스템 컴포넌트의 기능을 실행하기 위한 비즈니스 컴포넌트의 조합으로 구성되는데 기존의 컴포넌트 방법론들은 대다수 비즈니스 컴포넌트 식별 방법만을 제시하고 있다.

따라서 본 논문에서는 기존 방법론들의 문제점을 보완 확장하여 재사용 가능한 독립적인 컴포넌트를 평이한 개발자라 하더라도 효율적으로 식별할 수 있는 기준과 방법을 제시한다. 본 논문에서 제시하는 컴포넌트 식별방법은 시스템 컴포넌트를 식별하고 식별된 시스템 컴포넌트를 중심으로 비즈니스 컴포넌트를 식별한다. 비즈니스 컴포넌트를 식별하는 방법은 클래스 간의 구조적 특성, 메소드 호출의 유형에 의한 클래스 간의 종속성 그리고 메소드 호출 방향에 의한 클래스의 영향 범위를 고려하여 좀 더 효율적으로 비즈니스 컴포넌트를 식별할 수 있는 기준과 방법을 제시한다. 본 논문에서 제시하는 컴포넌트 식별 방법은 시스템 컴포넌트에 의한 기능적 재사용성, 요구사항 분석 단계로부터 비즈니스 컴포넌트로의 추적성, 평이한 개발자라 하더라도 효율적으로 비즈니스 컴포넌트를 식별할 수 있는 이점이 있다.

2. 관련연구

2.1 시스템 컴포넌트와 비즈니스 컴포넌트

컴포넌트 기반의 시스템은 컴포넌트를 단위로 두 가지로 구분될 수 있다. 유사한 기능을 그룹화하여 기능적 재사용이 가능하도록 컴포넌트화 한 시스템 컴포넌트와 시스템 컴포넌트의 기능을 수행하기 위하여 구성되는 비즈니스 컴포넌트가 존재한다[4].

시스템 컴포넌트는 기능적 재사용의 단위이면서 서브시스템의 단위이다. 시스템 컴포넌트는 시스템 인터페이스로 이루어지고 의미적으로 유사한 기능을 수행할 수 있는 유스케이스를 중심으로 식별됨으로 각각의 유스케이스는 시스템 컴포넌트가 수행해야 할 시스템 인터페이스가 된다. 시스템 컴포넌트의 인터페이스를 실행하기 위하여 독립적인 부품의 단위가 되는 컴포넌트가 비즈니스 컴포넌트이다.

2.2 기존의 컴포넌트 개발 방법론

CBD96[6]은 Computer Associates 사의 개발 도구인 COOL Joe를 이용한 컴포넌트 개발 아키텍쳐 모델이면서 개발 방법론이다. CBD96은 시스템 서비스 측면의 기능을 재사용 할 수 있는 시스템 컴포넌트에 대한 개념이 없고 전체 도메인을 중심으로 비즈니스 컴포넌트를 식별한다. 또한 비즈니스 컴포넌트를 식별하는데 있어서도 비즈니스 타입 다이어그램의 핵심타입(Core Type)을 식별하고 식별된 핵심타입을 중심으로 클래스 간의 의존성을 고려하여 비즈니스 컴포넌트를 식별한다. 따라서 개발자의 직관과 경험에 의하여 비즈니스 컴포넌트를 식별해야 하는 어려움이 존재한다.

CBD96[JD99]의 컴포넌트 개발 방법을 확장하여 Cheesman과 Daniels가 제안한 UML Components 방법[4]은 시스템 서비스 측면의 기능적 재사용이 가능하도록 시스템 컴포넌트의 정의가 명확히 명시되어 있지만 시스템 컴포넌트와 비즈니스 컴포넌트가 명확히 분리되어서 식별되기 때문에 비즈니스 컴포넌트를 조합하여 시스템 컴포넌트를 완성하기 위해서는 다시 시스템 컴포넌트의 인터페이스인 각 유스케이스를 분석하여 어떠한 타입, 즉 어떠한 클래스가 어느 시스템 컴포넌트에 포함될 것인지를 결정해야 한다. 또한 그 클래스는 결정된 시스템 컴포넌트의

어느 비즈니스 컴포넌트에 포함되는지를 다시 한번 고려하여야 한다. 비즈니스 컴포넌트를 식별하는 방법은 CBD96과 같은 방법으로 개발자의 직관과 경험에 의하여 비즈니스 컴포넌트를 식별하여야 하는 어려움이 있다.

Rational사의 RUP[7]는 Booch의 방법론, Rumbaugh의 방법론, Jacopson의 방법론을 통합한 방법론이다. RUP 방법에서의 컴포넌트 식별은 컴포넌트 식별단계에서 컴포넌트를 식별하기 위한 방법만을 제시하고 있지 컴포넌트를 식별하기 위한 구체적인 지침과 절차를 제시하고 있지 않으므로 개발자의 직관과 경험에 의하여 시스템 컴포넌트와 비즈니스 컴포넌트를 식별할 수 밖에 없다.

ETRI가 제안한 마르미III[11]의 컴포넌트 식별 방법은 대표적으로 객체모형을 통한 컴포넌트 식별방법과 UDA 테이블을 통한 컴포넌트 식별방법이 존재한다. 객체모형을 통한 컴포넌트 식별방법은 엔티티 유형의 클래스를 중심으로 핵심 클래스를 식별한다. 핵심 클래스는 유사한 관련 클래스들을 통해 참조가 많은 중심 클래스로 결정하고 핵심클래스와 관련된 주변 클래스들을 묶어 컴포넌트 후보를 도출한다. 이 방법은 기존의 방법론인 CBD96 방법과 유사하다. UDA 테이블을 통한 컴포넌트 식별방법은 유스케이스의 기능을 실행하기 위해서 참조하는 클래스가 어떠한 유형의 메소드(생성, 삭제, 수정, 조회)를 사용하는지를 분석한다. 이 때 이웃해 있는 클래스와 유스케이스를 중심으로 유스케이스가 참조하는 클래스의 메소드가 C(생성) D(삭제) W(수정) R(참조)의 관계에 대하여 가중치를 부여하고 친화성 분석에 의하여 컴포넌트를 식별한다. 그러나 마르미 III는 유스케이스와 클래스 간의 수직적 관계만을 분석하여 컴포넌트를 식별하기 때문에 수평적 관계인 클래스 간의 메소드 호출 유형이나 방향성을 고려하지 않고 있으므로 정확하게 컴포넌트를 식별할 수 없다. 또한 임의의 유스케이스가 기능을 실행하기 위해서 클래스를 참조만 한다면 클래스와 유스케이스가 그룹화될 조건이 제시되지 않으므로 참조관계만을 가진 유스케이스는 어느 후보 컴포넌트로 들어가도 상관없다고 제시하고 있다. 그러나 시스템 컴포넌트를 식별한 후 비즈니스 컴포넌트를 식별한다면 이러한 문제는 해결되는데 마르미 III 역시 시스템 컴포넌트의 개념이 정의되지 않고 전체 도메인에서 컴포넌트를 식별한다.

3. 분석 클래스 간의 종속적 특성에 의한 비즈니스 컴포넌트 식별

3.1 클래스 간의 종속적 특성

클래스 간에 종속성이 존재한다는 것은 클래스 C2와 C1에 대하여 <C2, C1> ∈ R의 관계가 존재한다면 클래스 C2에 대한 객체가 클래스 C1의 객체에게 메시지를 보냄으로 이루어지고 C2 객체의 메시지 호출에 의하여 C1의 객체가 영향을 받으므로 C1 객체는 C2 객체에 의존 또는 종속되어 있다고 정의할 수 있다. 본 절은 이러한 종속적 특성을 클래스 간의 구조적 관계, 메시지 호출의 유형 그리고 메시지 호출의 방향에 따라 분리하여 제시한다.

1) 클래스 간의 구조적 관계에 의한 종속성

클래스들의 구조적 관계는 포함관계, 상속관계 그리고 연관관계가 존재하고 포함관계와 상속관계는 클래스 간의 수직적 관계이고 연관관계는 클래스 간의 수평적 관계이다.

① 클래스 간의 수직적 관계의 종속성

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. 따라서 보다 독립적이어야 하고 컴포넌트 간의 종속성이 최소한이어서 서로 영향을 적게 받아야 하는 컴포넌트의 특성상 이러한 관계의 클래스들은 분리되어질 수 없고 반드시 하나의 컴포넌트 안에 포함되어져야 한다.

② 클래스 간의 수평적 관계의 종속성

클래스들의 관계가 연관관계라면, 즉, A클래스와 B 클래스 사이에 연관관계가 존재한다면 A 클래스와 B 클래스 사이에 메시지 호출에 의한 수평적 관계이다. 즉, 클래스 A와 B에 대하여 <A, B> ∈ R의 연관관계가 존재한다면 클래스 A와 B의 연관관계는 클래스 A에 대한 객체가 클래스 B의 객체를 생성, 삭제, 수정 그리고 조회하는 메시지를 보냄으로 이루어진다. 연관관계는 클래스들 간의 관계가 포함관계나 상속관계와 같이 클래스의 영향이 다른 클래스에 그대로 전파되는 것이 아니라 메시지 호출 유형에 따라서 클래스와 객체에 대한 수정 영향의 강도가 달라진다. 따라서 클래스의 구조적 특성에 의한 클래스 간의 종속의 강도는 포함관계 > 상속관계 > 연관

관계 순이고 컴포넌트의 특성상 포함관계나 상속의 수직적 관계의 클래스들은 분리되어질 수 없으므로 하나의 컴포넌트 안에 포함되어져야 하지만 연관관계에 있는 수평적 관계의 클래스들은 메소드의 호출 유형에 따라서 종속의 관계가 달라지기 때문에 연관관계의 클래스들은 분리되어져 각각 다른 컴포넌트로 포함되어질 수도 있고 아니면 하나의 컴포넌트 안에 모두가 포함되어질 수도 있다.

2) 메시지 호출 유형에 따른 클래스 간의 종속성

클래스 간의 메시지 호출의 유형은 다음과 같이 분류할 수 있다. 즉, <A, B> ∈ R의 관계에서 메시지의 유형은 다음과 같다.

① A 객체가 B 객체를 생성하는 메시지를 호출할 경우
② A 객체가 B 객체를 삭제하는 메시지를 호출할 경우
③ A 객체가 B 객체를 수정하는 메시지를 호출할 경우
④ A 객체가 B 객체를 참조하는 메시지를 호출할 경우

①과 ②의 경우는 A 객체에 의해서 B 객체의 구조가 변경되므로 B 객체를 참조하여 기능을 수행하는 다른 객체들은 전체적으로 구조적 측면에서 영향을 받는다. 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 객체를 참조하여 기능을 수행하는 다른 객체들은 전혀 영향을 받지 않는다.

3) 클래스 간의 메시지 방향에 따른 종속성

A 객체가 B 객체에게 B객체를 생성하는 메시지를 보내는 경우에는 A 객체에 의해서 B객체가 종속되는 관계이고 B 객체의 생성은 B 객체를 조작하는 다른 객체들에게 구조적으로 영향을 미친다. 그러나 B 객체가 A 객체에게 A 객체의 자료를 참조하는 메시지를 보낸다면 A 객체는 B 객체의 참조라는 메시지에 의해서 종속되는 관계이나 B 객체는 단지 A 객체의 데이터를 단지 참조만하기 때문에 A 객체는 전혀 영향을 받지 않는다. 따라서 객체들 간의 메시지 호출 방향에 의해서 종속되는 관계가 달라지므로 반드시 메시지 호출의 방향을 고려하여야 한다.

3.2 컴포넌트 식별 기준

컴포넌트의 특성을 부여한 컴포넌트의 식별기준을 본 논문의 3.1에 의하여 다음과 같이 정의한다.

정의 1. 클래스 간의 관계가 포함관계나 상속관계라면 그들 사이에 강한 종속관계가 존재하므로 하나의 컴포넌트 안에 포함되어져야 한다.

정의 2. 클래스 간의 관계가 연관관계라면 각각 다른 컴포넌트에 분리되어져 포함될 수 있다.

정의 3. 클래스 간의 관계가 연관관계이지만 메시지 호출의 유형이 생성과 삭제 메시지라면 그들 사이는 객체의 구조를 바꾸는 강한 종속관계가 존재하므로 하나의 컴포넌트 안에 포함되어져야 한다.

정의 4. 클래스 간의 관계가 연관관계이고 메시지 호출의 유형이 참조의 메시지라면 메시지 호출을 받는 객체는 전혀 영향을 받지 않으므로 그들 각각이 다른 컴포넌트로 포함된다 하더라도 각각의 컴포넌트는 독립성을 만족한다.

3.3 컴포넌트 식별 방법

다음은 3.2절에서 제시한 컴포넌트 식별 기준에 의하여 식별된 시스템 컴포넌트를 중심으로 비즈니스 컴포넌트를 식별하는 방법과 절차를 제시한다.

절차1. 시스템 컴포넌트를 식별한다.

1) 시스템의 유스케이스를 식별한다.
2) 식별된 각각의 유스케이스에 대하여 유스케이스 실체화 분석을 한다.
3) 식별된 유스케이스 중 서로 밀접한 관계의 유스케이스들은 하나의 유스케이스로 그룹화하고 식별된 각각의 유스케이스와 유스케이스 실체화 과정에 의해서 도출된 객체와의 관계성을 고려하여 시스템 컴포넌트를 식별한다.

절차2. 식별된 시스템 컴포넌트에 대한 클래스 다이어그램을 완성한다.

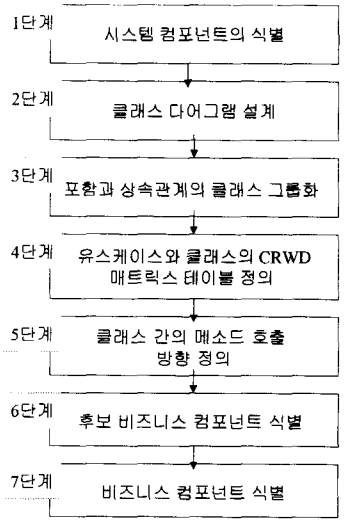


그림 1. 컴포넌트 식별 방법

절차 3. 3.2절의 정의 1에 의하여 클래스 다이어그램의 클래스들 중에 포함관계와 상속관계의 클래스들은 강한 종속관계가 존재함으로 하나의 클래스로 그룹화 한다.

절차 4. 시스템의 인터페이스인 유스케이스의 기능을 실행하기 위하여 객체를 조작하기 위한 연산은 객체의 생성, 삭제, 수정 그리고 참조 등으로 나뉘어지므로 시스템 컴포넌트가 포함하는 유스케이스, 즉 시스템 인터페이스가 각 클래스에 대하여 어떠한 메소드를 실행하는지를 나타내는 CRWD 메트릭스 테이블을 만든다. (C: 생성,R: 참조,W: 수정,D: 삭제)

절차 5. 정의 2, 3, 4에 제시된 특성을 이용하여 비즈니스 컴포넌트를 식별하기 위해서 절차 3의 CRWD 메트릭스 테이블에서 제시된 클래스들 사이의 메시지 호출관계를 정의한다.

절차 6. 정의 3에 의하여 메시지 호출관계에서 생성과 삭제의 관계인 클래스들은 하나의 클래스로 그룹화 한다. 식별된 각각의 클래스들은 하나의 후보비즈니스 컴포넌트로 식별된다.

절차 7. 식별된 후보 비즈니스 컴포넌트들을 다시 상세 정제한 후 완전한 비즈니스 컴포넌트를 식별한다.

4. 사례 연구 및 비교 분석

4.1 클래스 간의 종속적 특성에 의한 비즈니스 컴포넌트 식별

다음은 본 논문에서 제시한 방법을 치과병원 관리 사례를 통하여 그 효과성을 검증한다. 먼저 시스템 컴포넌트를 식별하고 식별된 시스템 컴포넌트 중 진료 시스템 컴포넌트를 중심으로 비즈니스 컴포넌트를 식별한다.

절차1. 유스케이스와 유스케이스 실체화에 의한 시스템 컴포넌트의 식별.

1) 치과병원 시스템의 유스케이스를 제시한다.

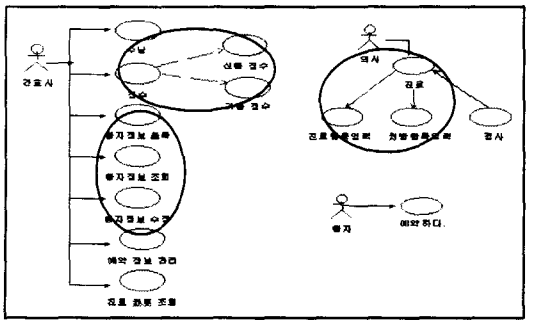


그림 2. 유스케이스 다이어그램

2) 식별된 유스케이스 각각에 대하여 RUP 분석 프로세스의 실체화 분석을 진행한다.

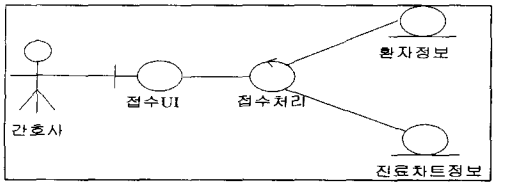


그림 3. 유스케이스의 실체화 분석

3) 포함관계의 유스케이스는 하나의 유스케이스로 그룹화하고 각 유스케이스와 실체화 분석에 의한 객체들의 관계에 의하여 시스템 컴포넌트를 식별한다.

액터	유스케이스	엔티티 객체											
		예약	접수	환자정보	진료차트	처방전	진료내역	신료비	검사항목	검사결과	약품정보	보험	비보험
간호사	예약을 하다.	V		V									
관자	예약을 관리하다.	V		V									
간호사	접수를 하다.		V	V	V								
	환자 정보를 관리하다.			V	V								
	수납하다.			V	V		V	V					
	진료차트를 조회하다.			V	V	V	V	V	V	V			
의사	진료를 하다.			V	V	V	V	V			V	V	V
	검사를 하다.			V	V			V	V				

그림 4. 유스케이스와 객체와의 관계

절차 2. 식별된 시스템 컴포넌트에 대한 클래스 다이어그램을 완성한다.

절차1에서 식별된 치과병원 시스템에서 식별된 시스템 컴포넌트는 접수, 예약, 진료이다. 진료시스템 컴포넌트를 중심으로 비즈니스 컴포넌트를 절차3부터 식별한다.

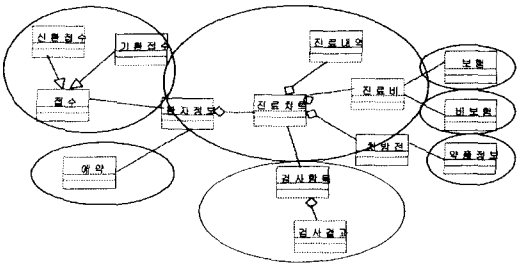


그림 5. 시스템 컴포넌트내의 클래스

절차3. 클래스 다이어그램에서 포함관계와 상속관계의 클래스들은 하나의 클래스로 그룹화 한다.

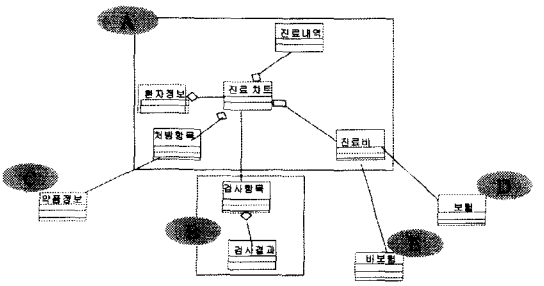


그림 6. 클래스의 그룹화

절차4. 시스템 컴포넌트가 포함하는 유스케이스를 중심으로 CRWD 메트릭스 테이블을 만든다. (C : 생성, R : 참조, W : 수정, D : 삭제)

인터페이스 / 클래스	A					B		C	D	E
	환자정보	진료차트	처방전	진료내역정보	진료비	검사항목	검사결과	약품정보	보험정보	비보험정보
환자정보등록	C	C								
환자정보조회	R									
환자정보수정	W									
환자정보삭제	D	D								
진료를 하다.	R	R	C	C	C	R		R	R	R
진료차트조회	R	R	R	R	R	R	R			
검사를하다.	R	R				C	C			
수납하다.	R	R		R	R					
처방전 조회	R	R	R							

그림 7. 유스케이스와 클래스간의 CRWD 메트릭스 테이블

절차5. CRWD 메트릭스 테이블에서 제시된 클래스들 사이의 메시지 호출관계를 정의한다.

절차 6. 메시지 호출관계에서 생성과 삭제의 관계인 클래스들은 하나의 클래스로 그룹화 한다. 식별된 각각의 클래스들은 하나의 비즈니스 컴포넌트로 식별된다.

인터페이스 / 클래스	A					B		C	D	E
	환자정보	진료차트	처방전	진료내역정보	진료비	검사항목	검사결과	약품정보	보험정보	비보험정보
환자정보등록	C	C								
환자정보조회	R									
환자정보수정	W									
환자정보삭제	D	D								
진료를 하다.	R	R	C	C	C	R		R	R	R
진료차트조회	R	R	R	R	R	R	R			
검사를하다.	R	R				C	C			
수납하다.	R	R		R	R					
처방전 조회	R	R	R							

그림 8. 클래스 간의 메시지 호출

절차 4의 CRWD 메트릭스 테이블에서 제시된 클래스들 사이의 메시지 호출관계에 의하여 진료 챠트에 의해서 검사항목이 생성(Create)되므로 진료챠트 클래스인 A 클래스와 검사항목과 검사결과 클래스인 B 클래스를 그룹화한 진료 컴포넌트가 식별된다. 나머지 클래스들은 참조의 관계에 있으므로 그룹화하지 않는다. 따라서 후보 비즈니스 컴포넌트는 진료, 약품정보, 비보험, 보험이고 식별 결과에 대한 후보 비즈니스 컴포넌트에 대한 다이어그램은 다음과 같다.

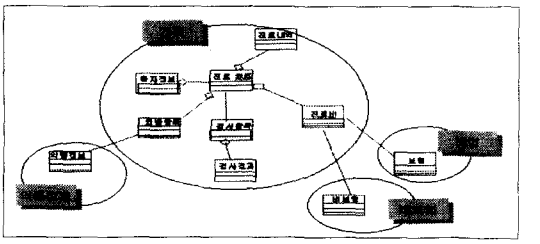


그림 9. 후보 비즈니스 컴포넌트

절차 7. 후보 컴포넌트를 다시 분석하여 정제한 후 완전한 비즈니스 컴포넌트를 식별한다.

약품정보, 비보험, 보험 컴포넌트들은 진료 컴포넌트와의 메시지 호출 관계가 전부 참조의 관계이므로 약품정보, 비보험, 보험 등은 그룹화 되어지지 않고 분리된 컴포넌트로 존재하는 것이 3.2절의 컴포넌트 식별 기준 정의4에 의하여 타당하다. 따라서 식별된 비즈니스 컴포넌트는 진료, 약품정보, 비보험, 보험 컴포넌트이다.

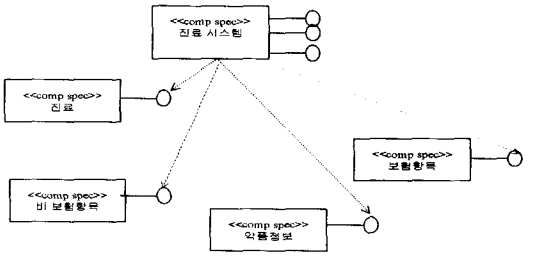


그림 10. 식별된 비즈니스 컴포넌트

4.2 기대효과 및 비교평가

다음은 본 논문에서 제안한 분석 클래스 간의 종속적 특성을 적용한 시스템 컴포넌트 기반의 비즈니스 컴포넌트 식별 방법의 기대효과를 제시하고 기존의 컴포넌트 식별 방법들과 비교평가 한다.

기대효과는 다음과 같다.

1) 시스템 컴포넌트를 중심으로 비즈니스 컴포넌트를 식별하기 때문에 요구사항 분석단계의 유스케이스로부터 비즈니스 컴포넌트까지 추적이 가능하므로 요구사항 변경이나 소프트웨어의 수정이 일어났을 경우에 효과적으로 관리할 수 있다.

2) 비즈니스 컴포넌트를 식별하기 위하여 전체 도메인을 중심으로 식별하는 것이 아니라 시스템의 부분 집합이 되는 시스템 컴포넌트를 중심으로 비즈니스 컴포넌트를 식별하므로 비즈니스 컴포넌트를 더욱 효과적으로 식별할 수 있다.

3) 비즈니스 컴포넌트를 식별하기 위하여 유스케이스와 클래스와의 관계인 수직적 관계, 클래스와 클래스와의 관계인 수평적 관계를 모두 다 적용하여 비즈니스 컴포넌트를 식별하므로 더욱 더 정확하게 비즈니스 컴포넌트를 식별할 수 있다.

4) 개발자의 직관과 경험에 의하여 컴포넌트를 식별하는 문제점을 보완하여 컴포넌트를 식별하기 위한 기준을 설정하였으므로 비즈니스 컴포넌트를 식별하는 것이 더욱 용이하다.

다음은 기존의 컴포넌트 식별 방법들과 본 논문에서 제안한 방법을 비교 평가하고자 하고 그 결과는 표 1과 같다.

5. 결 론

본 논문에서는 기존 방법론들의 컴포넌트 식별에 관한 문제점을 보완하여 분석 클래스들의 종속적 특성을 적용한 시스템 컴포넌트 기반의 비즈니스 컴포넌트 식별 방법을 제안하였다. 본 논문에서 제안한 시스템 컴포넌트 기반의 비즈니스 컴포넌트 식별 방법은 클래스 간의 구조적 관계, 메시지 호출의 유형 그리고 메시지 호출의 방향에 따른 분석 클래스 간의 종속적 특성을 적용하여 보다 독립적인 비즈니스 컴포넌트를 효율적으로 식별하였다. 또한 본 논문에서 제시한 방법을 치과병원관리 시스템의 사례에 적용하여 그 결과를 제시함으로 그 효율성을 검증 하였다.

기존의 컴포넌트 식별 방법들은 비즈니스 컴포넌트를 식별함에 있어서 개발자의 직관과 경험을 토대로 컴포넌트를 식별하도록 함으로 개발자의 시간과 노력을 많이 소모하는 어려움이 존재하였다. 그러나 본 논문에서는 비즈니스 컴포넌트를 바로 식별하는 것이 아니라 시스템 컴포넌트를 식별한 후 비즈니스 컴포넌트를 식별하도록 하고 비즈니스 컴포넌트를 식별함에도 클래스 간의 구조적 관계, 클래스 간의 메시지 호출의 유형 그리고 메시지 호출의 방향에 따른 분석 클래스 간의 종속적 특성을 적용하여 비즈니스 컴포넌트를 식별하는 기준과 방법을 제시하였으므로 평이한 개발자라 하더라도 보다 독립적인 컴포넌트를 효율적으로 식별할 수 있다.

표 1. 컴포넌트 식별 방법 비교 평가

비교항목 \ 식별방법	RUP	CBD96 (UML Components)	마르미 III	제안한 방법
시스템 컴포넌트 기반의 비즈니스 컴포넌트 식별	×	×	×	○
메소드 호출 유형을 적용	×	×	○	○
메소드의 호출 방향 적용	×	×	×	○
시스템 컴포넌트 식별 방법 제안	×	△	×	○
개발자의 노력, 비용 절감	×	△	△	○
클래스와 유스케이스간의 관계 적용	×	×	○	○
컴포넌트 식별 결과의 타당성	×	△	△	○

참 고 문 헌

[1] Desmond Francis Dsouza, Alan Cameran wills, *Objects, Component, and Frameworks with UML: the Catalysis approach*, Addison Wesley, 1999.

[2] 조은숙, "UML 기반의 컴포넌트 모델링 기법", 숭실대학교 박사학위 논문, 2000.

[3] George T. Heineman, William T. Council, *Component Based Software Engineering : Putting the Pieces Together*, Addison-Wesley, 2001.

[4] John Cheesman, John Daniels, *UML Components: A Simple Process for Specifying Component-Based Software*, Addison-Wesley, 2001.

[5] Clemens Szyperski, Dominik Gruntz, Stephan Murer, *Component Software: Beyond Object-Oriented Programming*, 2nd Edition, Addison-Wesley, 2002.

[6] John Dodd, "Identifying & Scoping CBD96 Components", Texas Instruments Inc., 1999.

[7] Ivar Jacopson, Grady Booch, James Rumbaugh, *The Unified Software Development Process*, Addison Wesley, 1999.

[8] 최미숙, 윤용익, 박재년, "RUP 기반의 컴포넌트 식별 방법에 관한 연구", 한국정보처리학회 논문지, 제9-D권, 1호, 2002.

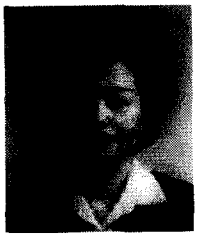
[9] Lee Sang Duck, Yang Young Jong, Cho Eun Sook, Soo Dong Kim, "COMO : A UML based Component Development Methodology", Proceeding of IEEE APSEC, pp.54-61, 1999.

[10] 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.

[11] ETRI, "컴포넌트 개발방법론 마르미 III", Technical Report, 2002.

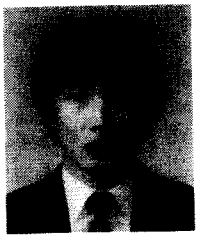
최 미 숙

1990년 전북대학교 졸업(이학사)
1994년 숙명여자대학교 일반대학원 컴퓨터과학과(이학석사)
2002년 숙명여자대학교 일반대학원 컴퓨터과학과(이학박사)
1995년~1999년 나주대학 소프트웨어 개발과 전임교수
2003년~현재 우석대학교 컴퓨터공학과 강사
관심분야 : 소프트웨어 개발 방법론, CBD 방법론, 소프트웨어 아키텍처



조 은 숙

1993년 동의대학교 자연과학대학 전산통계학과 졸업(이학사)
1996년 숭실대학교 공과대학 컴퓨터학과 졸업(공학석사)
2000년 숭실대학교 공과대학 컴퓨터학과 졸업(공학박사)
2002년~2003년 한국전자통신연구원 초빙연구원
2000년~현재 동덕여자대학교 정보학부 강의전임교수
관심분야 : 소프트웨어 모델링, CBD 방법론, 소프트웨어 아키텍처, 웹서비스 컴퓨팅



하 종 성

1984년 서울대학교 컴퓨터공학과(학사)
1986년 한국과학기술원 전산학과(석사)
1996년 한국과학기술원 전산학과(박사)
1986년~1989년 (주)현대전자산업 근무
1990년~현재 우석대학교 컴퓨터공학과 교수
2001년 미국 조지워싱턴대학교 방문교수
관심분야 : 응용계산기하학, 컴퓨터그래픽스, CAD/CAM 등임