論文
3-3-12

컴포넌트 식별 방법에 관한 비교 연구
(A Comparison Study of Methods about Component Identification)

최 미 숙*
(Mi-Sook Choi)

요 약

컴포넌트에 의한 시스템 구축은 기능 중심의 독립적인 컴포넌트를 기반으로 새로운 소프트웨어를 구축한다. 컴포넌트를 중심으로 한 소프트웨어의 개발은 품질향상, 빠른 개발과 유지보수의 효율성과 같은 소프트웨어의 재사용에 의해서 얻을 수 있는 이익을 얻을 수 있다. 컴포넌트 기반의 시스템에서 재사용성을 높일 수 있는 고 품질 컴포넌트의 효율적인 식별과 구축은 가장 중요한 목표이자 성공 요소이다.

따라서 컴포넌트 개발자는 현재 각 컴포넌트 개발 방법론들의 컴포넌트 식별에 대한 장점과 단점을 파악하여 개발하고자 하는 시스템에 적합한 방법론을 선택하는 것이 필요하고 또한 여러 방법론의 단점을 보완하여 독립적인 컴포넌트를 효율적으로 식별할 수 있는 새로운 방법론을 개발하는 것이 필요하다. 그러기 위해서는 다양한 방법론에 대한 비교 분석이 절실히 요청된다.

본 논문은 최근에 산업계에서 널리 쓰이고 있는 대표적인 컴포넌트 기반 소프트웨어 개발 방법론인 Rational사의 RUP(Rational Unified Process)[1,9], Computer Associates사의 CBD96[2,3]과, Compuware사의 UNIFACE[5] 그리고 Cheesman과 Daniels가 CBD96의 컴포넌트 개발 방법을 확장하여 제안한 UML Components 방법론[4] 등의 컴포넌트 식별 방법을 설명하고 그들의 문제점을 기술한다. 그리고 컴포넌트 식별 방법들을 총체적인 측면에서 비교 분석한다.

ABSTRACT

System developments by components are accomplished by creating new software based on independent components having respective function. Generally, component-based software developments are expected to obtain profits caused by reuse of software, such as improvement of quality, rapid development, and efficiency of maintenance. In a component-based system, the most important goal and also key to success is to identify and construct high quality components that may increase reusability.

Therefore, it is necessary for component developers to understand strong and weak pointsof existing component development methodologies in the aspect of identification of component, and to select the most appropriate methodology for the system to be constructed. It is also necessary for component developers to develop a new methodology enabling effective identification of independent components through modification and/or complementation of existing methodologies. The modification and complementation needs comparison and analysis of various existing methodologies.

Accordingly, the this paper is to provide explanation for some widely used methodologies representing the existing component-based software development methodologies such as RUP(Rational Unified Process) of Rational company [1,9], CBD96 of Computer Associates company [2,3], UNIFACE of Compuware company [5], and UML components methodologyderived from extending of the component developing method of CBD96 by Cheesman and Daniels [4]. The this paper is also to point out respective problems of the representative existing methodologies. Furthermore, component identification methodologies are compared and analyzedon the whole through this paper.

* 정회원 : 숙명여자대학교 정보과학부 교수

논문접수 : 2002. 2. 20.
심사완료 : 2002. 3. 14.

과학기술부 지원2001년도 숙명여대 KISTEP과제에 의해서 수행된 연구과제임

1. 서론

환경이 복잡하고 빠르게 변화함에 따라서, 구축될 시스템 또한 제한된 기간에 저렴한 비용으로 원하는 시스템을 구축 해야 한다. 따라서 컴포넌트를 기반으로 한 시스템 구축은 필수적인 것이다. 이러한 컴포넌트 중심의 시스템 구축은 기능 중심의 독립적인 부품 단위인 컴포넌트를 조립하여 새로운 소프트웨어를 구축한다. 컴포넌트 기반의 소프트웨어의 개발은 품질향상, 빠른 개발과 유지보수의 효율성과 같은 소프트웨어 재사용에 의해서 얻을 수 있는 이익을 얻을 수 있다. 따라서 이러한 컴포넌트 기반의 시스템 개발 중 가장 중요한 작업은 재사용성을 높일 수 있는 고 품질 컴포넌트의 효율적인 식별과 구축이다. 즉, 컴포넌트 식별을 위한 가장 중요한 작업은 컴포넌트를 식별하기 위한 객체를 어떻게 효율적으로 식별할 것인가?, 보다 독립적인 컴포넌트를 식별하기 위해서 응집성과 결합성이 좋은 독립적인 컴포넌트를 어떻게 추출할 것인가?, 기능적 재사용을 위한 컴포넌트를 어떻게 효율적으로 식별할 것인가?, 유지보수단계의 요구사항 변경의 용이성을 위하여 요구사항 분석단계의 산출물부터 설계단계의 산출물들간의 추적이 가능한가?, 또한 현업사용자의 능동적인 참여 환경과 평균적인 시스템 분석가가 어려움이나 모호함이 없이 체계적으로 컴포넌트를 식별할 수 있는 가? 이다.

따라서 본 논문은 컴포넌트 기반 소프트웨어 개발 방법론인 Rational사의 RUP(Rational Unified Process)[1,9], Computer Associates사의CBD96[2,3], Cheesman과 Daniels의 UML Components[4] 그리고 Compuware사의 UNIFACE[5] 등의 컴포넌트 식별 방법을 설명하고 위에서 제시한 중요한 작업을 중심으로 그들의 문제점을 기술한다. 그리고 컴포넌트 식별 방법들을 총체적인 측면에서 비교 분석한다.

본 논문은 2장에서 컴포넌트 식별 방법을 기술하고, 3장에서 각 방법론의 문제점을 제시하고, 4장에서 기존 방법론들의 컴포넌트 식별 방법을 총체적으로 비교 분석하고 5장에서 결론과 향후 연구과제를 기술 한다.

2. 컴포넌트 식별 방법

2.1 RUP에서의 컴포넌트 식별 방법

RUP(Rational Unified Process)는 Booch, Rumbaugh, Jacopson 방법론을 통합한 방법론이다. RUP에서의 컴포넌트 식별 방법은 사용자의 요구사항 분석을 위하여 유스케이스 분석을 한 후 추출된 유스케이스를 기반으로 하여 정적 모델링과 동적 모델링을 통해서 시스템 아키텍처를 정의한 후에 컴포넌트를 식별한다.

그 구체적인 절차는 다음과 같다.

1) 유스케이스 분석

시스템의 액터와 유스케이스를 식별한 후 각 유스케이스당 이벤트 흐름을 정의한다.

2) 객체 분석

추출된 유스케이스의 이벤트 흐름을 기반으로 유스케이스 실체화 과정을 통해 객체를 분석한다.

3) 동적 모델링

인터랙션 다이어그램(Interaction Diagram)을 통해 추출된 객체들의 상호작용을 분석함으로 추출된 객체를 정제하고 객체들의 오퍼레이션을 추출한다.

4) 정적 모델링

추출된 객체와 오퍼레이션을 통해서 클래스 다이어그램을 완성한다.

5) 아키텍처 확정

시스템의 수직적 아키텍처와 수평적 아키텍처를 확정 한다.

6) 컴포넌트 식별

① 프로세스 설계를 통한 컴포넌트 식별(Outside-In)
식별된 프로세스를 기준으로 하나 혹은 그 이상의 프로세스를 하나의 컴포넌트로 작성한다.

② 클래스 간의 결합 정도를 기준으로 컴포넌트 식별(Inside-Out)

상호 협력 다이어그램(Collaboration Diagram), 클래스 다이어그램(Class_ Diagram)을 참조하여 밀접하게 협력하는 클래스 그룹을 식별한다.

③ 아키텍쳐 레벨에서 식별된 서브시스템 위주의 컴포넌트 식별 아키텍쳐가 식별한 서브 시스템을 기준으로 컴포넌트를 식별하고 설계측면에서 구성한 서브 시스템은 구현부로 옮아가는 과정에서 컴포넌트로 작성된다.

2.2 CBD96에서의 컴포넌트 식별 방법

CBD96은 Computer Associates 사의 개발 도구인 COOL을 이용한 컴포넌트 개발 아키텍쳐 모델이면서 개발 방법론이다. 컴포넌트의 식별은 시스템의 전체 도메인을 중심으로 타입 다이어그램과 비즈니스 타입 다이어그램을 추출하고 추출된 비즈니스 타입 다이어그램의 타입 중 핵심 타입(Core Type)을 중심으로 관련된 타입을 그룹화하여 비즈니스 컴포넌트를 추출한다. 이렇게 비즈니스 컴포넌트를 추출한 후 인터랙션 다이어그램(Interation Diagram)을 통해 보다 독립적인 비즈니스 컴포넌트를 추출하도록 정제하는 과정을 거친 후에 비즈니스 컴포넌트를 조합하여 컴포넌트 아키텍쳐를 정의 한다.

그 구체적인 절차는 다음과 같다.

1) 개념(Concept) 분석

프로젝트 범위 내에서, 현업 사람들에 의해서 인식되는 기본적인 컨셉을 식별해 낸다.

2) 유스케이스 분석

시스템의 액터와 유스케이스를 식별한 후 각 유스케이스당 이벤트 흐름을 정의한다.

3) 비즈니스 타입 모델(Business Type Model) 정의

비즈니스 타입 모델링은 타입 다이아그램을 복사하여 행해지며 타입 다이아그램의 타입들 중 프로젝트 범위에서 벗어나는 것, 중복 되는 것, 애매모호한 것들의 속성과 연관관계는 삭제 한다.

4) 핵심 타입(Core Type) 식별

비즈니스 타입 다이어그램의 타입들 중에서 핵심 타입을 가려내고 그 외 모든 타입들은 일부 핵심 비즈니스 타입의 상세 타입(Detailing type)으로 지정한다.

5) 인터페이스 책임 범위 결정

한 개의 핵심 비즈니스 타입 당 한 개의 인터페이스를 추가 한다. 즉, 한 개의 핵심 비즈니스 타입 당 한 개의 인터페이스가 있다고 보며 한 인터페이스당 한 개의 비즈니스 컴포넌트를 생각한다.

6) 인터페이스 의존성 결정

이 스텝은 인터페이스간의 사용 의존성(Usage Dependencies)을 조사하여 인터페이스 의존성을 줄이거나, 같은 컴포넌트에 대해 상호-의존적인 인터페이스들을 그룹으로 묶는 것이다.

7) 컴포넌트 아키텍쳐 정의

식별된 비즈니스 컴포넌트를 아키텍쳐 스타일의 형태로 아키텍쳐를 정의한다.

아키텍쳐의 스타일로는 방사형(Radical) 스타일, 계층형(Hierarchical) 스타일, 네트워크(Network) 스타일이 있다.

2.3 UML Components에서의 컴포넌트 식별

CBD96의 컴포넌트 개발 방법을 확장하여 Cheesman과 Daniels가 제안한 UML Components 방법은 CBD96에서의 컴포넌트 식별 방법과 다르게 시스템 컴포넌트와 비즈니스 컴포넌트의 개념을 명확히 분리하여 적용하고 있다. 컴포넌트의 식별은 시스템의 전체 도메인을 중심으로 타입 다이어그램과 비즈니스 타입 다이어그램을 추출하고 추출된핵심 타입(Core Type)을 중심으로 관련된 타입을 그룹화하여 비즈니스 컴포넌트를 추출한다. 동시에 요구사항 분석을 통하여 추출된 유스케이스에 의해 시스템 컴포넌트를 추출한다. 그 이후에 인터렉션 다이어그램에 의해 추출된 비즈니스 컴포넌트를 정제하고 컴포넌트 아키텍쳐를 정의하기 위하여 추출된 시스템 컴포넌트에 포함될 비즈니스 컴포넌트를 찾아서 조합한다.

그 구체적인 절차는 다음과 같다.

1) 유스케이스 분석

시스템의 액터와 유스케이스를 식별한 후 각 유스케이스당 이벤트 흐름을 정의한다.

2) 시스템 인터페이스와 오퍼레이션 분석

1) 에 의해서 추출된 유스케이스를 중심으로 시스템 인터페이스와 오퍼레이션을 정의한다.

3) 비즈니스 타입 모델 정의

비즈니스 개념 모델을 복사하고 시스템의 입장에서 정제하여 시스템에 의해서 관리 되어야 할 정보를 중심으로 비즈니스 타입 모델을 생성하고 비즈니스 타입 모델의 타입 중 핵심 타입을 식별한다.

4) 비즈니스 인터페이스 식별

비즈니스 타입 모델의 핵심 타입을 중심으로 비즈니스 인터페이스를 식별한다.

즉, 한 개의 핵심 비즈니스 타입 당 한 개의 인터페이스를 추가 한다.

5) 2)와 3)에 의해서 초기 컴포넌트 명세와 아키텍쳐 생성

추출된 시스템 인터페이스와 비즈니스 인터페이스를 중심으로 컴포넌트의 명세를 작성하고 초기 컴포넌트 아키텍쳐를 정의 한다.

6) 컴포넌트 상호 작용분석에 의해 컴포넌트 아키텍쳐 정련

시스템 인터페이스와 비즈니스 인터페이스들에 대한 오퍼레이션의 상호작용을 분석, 식별하고 상호작용분석에 의해서 추출된 인터페이스들의 의존성과 참조무결성을 고려하여 초기 컴포넌트 아키텍쳐를 정련한다.

2.4 UNIFACE에서의 컴포넌트의 식별

Compuware사에서 개발한 UNIFACE에 대한 컴포넌트 식별 방법은 요구 사항 분석을 통하여 상위 유스케이스를 식별하고 각 상위 유스케이스별로 비즈니스 기능(Business Function Modeling)모델링을 통해서 기본적인 유스케이스(Elementary Use Cases)를 추출 한다. 그 이후에 객체 모델링(Object Modeling)과 행위 모델링(Behavioral Modeling)을 통해서 클래스 다이어그램을 완성 한다. 객체들의 상호작용 분석에 의해서 밀접하게 관련된 클래스들을 그룹화하여 패키지를 추출하여 비즈니스 컴포넌트를 식별 한다.

그 구체적인 절차는 다음과 같다.

1) 상위 유스케이스 정의

시스템의 상위레벨의 비즈니스 기능을 도출한다.

2) 기본적인 유스케이스 정의 : 비즈니스 기능 모델링

추출된 상위 유스케이스가 완성되기 위하여 각 액터가 수행해야 하는 기능인 기본적 유스케이스를 분석한다.

3) 객체 분석 : 객체 모델링

시스템의 객체를 추출한 후 추출된 객체를 중심으로 클래스 다이어그램을 완성하고 시스템을 좀 더 작은 관리 단위로 나눈 패키지 다이어그램을 추출된 클래스를 중심으로 분석 한다.

4) 객체들의 상호작용 분석 : 행위 모델링

추출된 객체들간에 상호작용을 분석함으로 클래스의 메소드를 추출하고 추출된 클래스 다이어그램과 패키지 다이어그램을 정제한다.

5) 컴포넌트 식별 : 컴포넌트 모델링

어플리케이션을 생성하기 위하여 사용되어질 수 있는 소프트웨어 오퍼레이션들의 독립적인 배포단위가 컴포넌트이므로 오퍼레이션들을 이러한 기준에 맞추어 클러스터링함으로 컴포넌트를 식별한다. 즉, 컴포넌트 패키지 다이어그램의 각각의 패키지는 비즈니스 컴포넌트로 추출된다.

3. 기존 개발 방법의 문제점

3.1 CBD96에서의 컴포넌트 식별 방법

컴포넌트를 식별하기 위하여 절차와 지침이 다른 방법들에 비해서 좀 더 체계적이고 구체적으로 제시

되고 있는 장점이 있지만 몇 가지의 문제점이 존재한다. 그 문제점은 **첫째**, 현업 담당자는 자기의 업무에 대해서 의미 있는 개념을 추출하고 그들 사이의 연관관계를 나타냄으로 타입 다이어그램을 그리는데 소프트웨어에 대한 아무런 개념이 없는 현업 담당자는 어떠한 단어가 의미가 있는지 없는지 구별하기가 쉽지 않고 그들 사이에 관계성을 도출하기가 쉽지 않다. **둘째**, 추출된 타입 다이어그램을 기반으로 하여 분석가는 시스템 설계 측면에서 불필요한 타입을 없애고 필요한 타입을 추가하여 비즈니스 타입 다이어그램을 추출한다. 그러나현업 담당자에 의해서 추출된 타입 다이어그램이 완전히 잘못 되어졌다면 분석가에 의해서 올바른 비즈니스 타입 모델을 추출하기가 쉽지 않다. 또한 비즈니스 타입 모델의 타입을 도출할 때 타입 추출의 구체적인 지침 없이 도출한다는 것은 경험이 많은 시스템 분석가의 직관에 의해서 객체를 추출해야 하는 모순이 있다. **셋째**, 타입 다이어그램이나 비즈니스 타입 다이어그램을 전체 시스템을 중심으로 도출하기 때문에 모델을 도출하기가 쉽지 않고 요구사항 변경시 요구사항 분석 단계부터 컴포넌트 식별단계 까지 체계적으로 추적이 가능하지 않다. **넷째**, 시스템 서비스 측면의 기능을 재사용할 수 있는 시스템 컴포넌트를 식별하기 위한 지침과 절차가 없고 비즈니스 컴포넌트 식별 방법만을 중심으로 컴포넌트 추출이 이루어 진다. 다섯째, 비즈니스 컴포넌트를 식별하는데 있어서 핵심타입을 중심으로 의존성을 고려하여 비즈니스 컴포넌트를 식별한다. 그러나 핵심 타입을 추출하기가 쉽지 않으므로 분석가의 직관과 경험에 의하여 비즈니스 컴포넌트를 추출하여야 하는 문제점이 존재한다.

3.2 UML Components에서의 컴포넌트 식별

Cheesman과Daniels가 제안한 UML Components 방법은 CBD96의 문제점들을 가지고 있지만 시스템 서비스 측면의 기능적 재사용이 가능하도록 시스템 컴포넌트의 정의가 명확히 명시되어 있고 시스템 컴포넌트와 비즈니스 컴포넌트의 정의가 명확히 분리되어져 있다. 그러나 시스템 컴포넌트를 추출하기 위하여 유스케이스를 어떻게 그룹화할 것인지 명확한 기준이 정의 되지 않아 시스템 컴포넌트를 추출하기가 쉽지 않다. 또한 시스템 컴포넌트와 비즈니스 컴포넌트가 명확히 분리 되어서 추출되어지기 때문에 비즈니스 컴포넌트를 조합하여 시스템 컴포넌트를 완성하기 위해서는 다시 시스템 컴포넌트에 포함된 인터페이스인 각 유스케이스를 분석하여 어떠한 타입이 이 시스템 컴포넌트에 포함될 것인지를 추출하고 추출된 타입이 어느 비즈니스 컴포넌트에 포함되는지를 다시 한번 고려하여야 하는 번거로움이 있다.

3.3 RUP에서의 컴포넌트 식별 방법

표준화된 UML을 산출물로 정의하고 사용되는 개발 방법론이다. 그러나 CBD96에서 제시한 컴포넌트 식별 방법의 문제점 중 첫째, 둘째, 셋째의 문제점을 많이 보완하고 있는 개발 방법론이다. 즉, RUP는 요구사항 분석 단계의 산출물로 유스케이스 모델을 산출하고 분석 단계에서 각 유스케이스의 이벤트 플로우를 보고 인터페이스 객체, 컨트롤 객체, 엔티티 객체 추출을 통하여 CBD96 보다는 좀더 체계적이고 효율적으로 객체를 도출해 나간다. 클래스 다이어그램을 도출할 경우에도 인터랙션 다이어그램을 통해서 효율적으로 도출한다. 또한 유사한 유스케이스를 그룹화 하여 시스템 컴포넌트인 패키지 다이어그램을 도출하고 패키지 다이어그램을 중심으로 클래스 다이어그램을 산출한다. 이는 시스템을 의미적으로 분할하여 클래스 다이어그램을 도출하므로 이해하기가 쉽고 분산환경에서 전개(Deploy)할 경우에 분할이 쉬우며 요구사항 변경시 요구사항 분석단계부터 설계단계까지 각 산출물을 추적할 수가 있어 요구사항 변경을 쉽게 수용할 수 있다. 그러나 이러한 장점을 보유하고 있음에도 컴포넌트를 추출하기 위하여 널리 사용되지 못하고 있는 문제점은 컴포넌트 식별단계에서 컴포넌트를 식별하기 위한 방법만 제시하고 있지 제시한 방법을 사용해서 컴포넌트를 식별하기 위한 구체적인 지침과 절차를 제시하고 있지 않으므로 시스템 분석가의 직관과 경험에 의하여 컴포넌트를 식별할 수 밖에 없다. 또한 시스템 컴포넌트인 패키지 다이어그램이나 서브 시스템을 식별하기 위한 기준이 명확히 제시 되어있지 않아 수직적 아키텍처를 설계하기가 쉽지 않고 프리젠테이션 층(Presentation Layer), 비즈니스 로직 층(Business Logic Layer), 미들웨어 층(Middle Layer)등, 수평적

아키텍처 또한 설계하기가 쉽지 않다. 프리젠테이션 층은 환경에 따라서 자주 바뀔 수 있으므로 엄격한 의미에서 재사용이 빈번하지 않음에도 불구하고 컴포넌트를 식별할 때 수평적 계층 모두를 고려 한다.

3.4 UNIFACE에서의 컴포넌트의 식별

Compuware의UNIFACE는 거의 모든 모델링을 UML의 모델과 표기법을 그대로 사용하고 있어서 CBD96이나 UML Components의 문제점을 많이 보완하고 있는 방법론이다. 그러나 첫째, 추출된 클래스를 중심으로 서로 연관된 클래스를 그룹화하여 패키지를 추출하는데 명확한 기준이 설정되어 있지 않고 둘째, 객체를 추출하는데 있어서도 전체 도메인을 중심으로 객체를 도출하므로 CBD96이나 UML Components가 가지고 있는 문제점을 가지고 있다. 셋째, 클래스들과 행위 모델링에 의하여 추출된 오퍼레이션을 기준으로 유사한 기능을 그룹화하여 비즈니스 컴포넌트를 식별하는데 유사한 기능을 어떻게 그룹화 해야 하는가에 대한 상세하고 명확한 지침이 설정되어 있지 않아서 분석가의 직관과 경험에 의하여 컴포넌트를 식별할 수 밖에 없다.

넷째, 비즈니스 컴포넌트들 사이의 중복되는 객체들에 대한 참조의 무결성을 어떻게 해야 하는지에 대한 지침이 존재하지 않다. 다섯째, 시스템의 기능을 재사용할 수 있는 시스템 컴포넌트에 대한 추출 방법이 존재하지 않고 비즈니스 컴포넌트 추출에만 중점을 두고 있다. 따라서 비즈니스 컴포넌트를 조합하여 시스템 컴포넌트를 완성하기 위해서는 다시 시스템 컴포넌트에 포함된 인터페이스인 각 유스케이스를 분석하여 어떠한 클래스가 이 시스템 컴포넌트에 포함될 것인지를 추출하고 추출된 클래스가 어느 비즈니스 컴포넌트에 포함되는지를 다시 한번 고려하여야 하는 번거로움이 있다.

4. 기존 컴포넌트 식별 방법의 총괄 비교 분석

3장에서 제시한 문제점을 기준으로 하여 기존 컴포넌트 식별 방법을 다음 <표 1>에서 총체적으로 비교 분석하고자 한다.

< 표 1 > 총체적인 비교 분석표

<Table 1> The comparison and analysis table by component identification methodology

식별 방법 / 비교항목	RUP	CBD96	UML Components	UNIFACE
컴포넌트의 식별 기준	Usecase, Class	Entity Class	Entity Class	Operation
컴포넌트 추출 기법	프로세스, 클래스간의 결합도, 서브시스템	독립적이면서 영구적인 엔티티 클래스 기준	독립적이면서 영구적인 엔티티 클래스 기준	유사한 기능을 그룹화하기 위한 오퍼레이션 기준
주요 결정 인자	Usecase, Class	Entity Class	Entity Class	Class, Operation
컴포넌트 식별 절차와 지침	△	O	O	O
요구사항 분석 단계 부터의 추적성	O	△	O	△
객체 추출 용이성	O	X	X	O
시스템 컴포넌트 정의	O	X	O	X
비즈니스 컴포넌트 식별 방법 정의	△	O	O	O
컴포넌트 사이의 중복객체 고려	X	O	O	X
시스템 분석가의 컴포넌트 추출 용이성	X	X	X	X
컴포넌트 식별의 상세절차와 상세 지침	X	△	△	X

O: 지원됨, △: 어느 정도 지원됨, X: 지원 않됨

5. 결론 및 향후 연구과제

본 논문은 컴포넌트 기반 소프트웨어 개발 방법론인 Rational사의 RUP, Computer Associates사의 CBD96, Cheesman과 Daniels의 UML Components 그리고 Compuware사의 UNIFACE 등의 컴포넌트 식별 방법을 설명하고 각 방법론의 문제점을 기술하고 컴포넌트 식별 방법의 가장 핵심 요소를 기준으로 총체적인 관점에서 비교 분석하였다. 따라서 본 논문은 각 방법론들의 문제점을 보완하여 개발하고자 하는 시스템에 적합하고 효율적인 새로운 컴포넌트 식별 방법론을 개발하기 위한 지침을 마련하였다.

또한 컴포넌트를 식별하기 위한 각 방법론을 비교 분석한 결과 현재 방법론 중 UML Components 방법이 가장 효과적인 방법으로 분석 되었고 개선해야 될 점은 대 다수 컴포넌트 방법들이 기능적 단위로 재 사용이 가능한 시스템 컴포넌트가 존재함에도 불구하고 비즈니스 컴포넌트 식별 방법에만 중점을 두어 컴포넌트를 추출하고 있었고 비즈니스 컴포넌트를 식별하는 방법도 시스템 분석가의 직관과 경험에 의존해야 하는 방법으로 비즈니스 컴포넌트를 추출하기가 용이하지 않았다. 또한 컴포넌트의 근원이 되는 객체를 추출하는 부분이 비효율적이었고 대체로 컴포넌트를 추출하기 위한 방법이 상세히 제시되지 않았다. 그러나 비교한 4가지 방법론들은 문제점만을 가지고 있는 것이 아니라 각각의 고유한 방법론들 나름대로 장단점을 가지고 있었다.

따라서 향후 연구과제로는 본 논문에서 제시하고 있는 기존의 컴포넌트 방법론들의 단점을 보완하여 즉, 효율적인 객체의 추출, 기능적 재사용을 위한 시스템 컴포넌트의 포함과 효율적인 시스템 컴포넌트의 추출, 그리고 분석가의 직관과 경험을 배제한 비즈니스 컴포넌트의 효율적인 추출의 측면을 중심으로 평이한 시스템 분석가라도 효과적으로 컴포넌트를 추출할 수 있는 새로운 개발 방법을 제안하는 것이다.

※ 참고문헌

[1] Ivar Jacobson, Grady Booch, James Rumbaugh, 'The Unified Software Development Process', Addison-Wesley, 1999.

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

[3] Cool software korea, "CBD Project Guide using Cool: Joe ver 1.0", Cool software korea, 2000

[4] John Cheesman, John Daniels, 'UML Components', pp.67-120, Addison-Wesley, 2001

[5] Compuware corp., "UNIFACE Development Methodology: UNIFACE V7.2", Compuware corp., 1998

[6] Seok-Jin Yoon, Gyu-Sang Shin, "Extraction Component from Object-Oriented Programs", Proceedings of the 14th KIPS Fall Conference, 2000

[7] 조진희, 이우진, 김민정, 신규상, "CBD 방법론의 컴포넌트 식별 방법의 비교", 한국정보처리학회지, 제7권 제2호, pp.515-518, 2000.

[8] Desmond Francis Dsouza, Alan Cameran wills, 'Objetcs, Component, and Frameworks with UML: the Catalysis approach', Addison Wesley, 1999

[9] Misook Choi, Kyunghee Kim, Seojung Lee, Yongik Yoon, Jaenyun Park, "Change Impact Analysis based on UML", CIMCA International Conference, Proceeding, 2001

[10] Misook Choi, Hyunhee Koh, Yongik Yoon, Jaenyun Park, "Component Identification based on Usecase", ICIS '01 International Conference, 2001

[11] Misook Choi, Hyunhee Koh, Yongik Yoon, Jaenyun Park, "Algorithm and Graph for Change Management by Software Architecture ", ICIS '01 International Conference, 2001

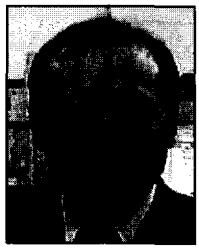
최 미 숙

1990년 전북대학교 졸업(학사)
1994년 숙명여자대학교 대학원 전산학과(이학석사)
1995년~1999년 나주대학 소프트웨어개발과 전임강사
1997년~현재 숙명여자대학교 대학원 전산학과 박사과정 수료
관심분야 : 컴포넌트 모델링, 컴포넌트 메트릭, 컴포넌트 테스팅

윤 용 익

1983년 동국대학교 졸업(학사)
1985년 한국과학기술원 전산학과(이학석사)
1994년 한국과학기술원 전산학과(이학박사)
1985년~1997년 한국 전자통신연구소 책임 연구원
1997년~현재 숙명여자대학교 정보과학부 교수
관심분야 : 정보통신, 멀티미디어통신, 분산시스템, 분산 미들웨어 시스템, 분산 데이터베이스 시스템, 실시간 OS/DBMS, 실시간 처리시스템

박 재 년



1966년 고려대학교 졸업(학사)
1969년 고려대학교 이학석사
1981년 고려대학교 이학박사
1979년~1983년 전남대학교 전산통계학과 교수
1983년~현재 숙명여자대학교 정보과학부 교수
관심분야 : 시스템 개발방법론, 품질 평가, 메타 DB, 시뮬레이션