2000년 한국정보처리학회 추계 학술발표논문집 제 7 권 제 2 호

CBD 방법론의 컴포넌트 식별 방법의 비교

조진희[0] 이우진 김민정 신규상
한국전자통신연구원 소프트웨어공학연구부 컴포넌트공학연구팀
{chojh, woojin, minjkim, gsshin}@etri.re.kr

A Comparison of Methods for Identifying Components in CBD Methodologies

Jin-Hee Cho[0] Min-Jung Kim Woo-Jin Lee Gyu-Sang Shin
Component Engineering Team, Software Engineering Department,
ETRI-CSTL

요 약

컴포넌트 소프트웨어 기술은 재사용성, 적시성, 유지 보수성 등이 업체의 경쟁력으로 대두되고 있는 정보기술 업계에서 점차 각광을 받고 있다. 현재 다양한 컴포넌트 기반 개발 방법론 및 지원 도구들이 제공되고 있지만 각 방법론별 고유의 컴포넌트의 식별 방법을 기반으로 하고 있으며 포괄적인 컴포넌트 식별 방법을 제공하는 방법론은 없다고 볼 수 있다. 이 논문에서는 현재 산업계에서 쓰이고 있는 여러 방법론과 한국전자통신연구원에서 개발하고 있는 컴포넌트 기반 개발방법론(FOCUS)에서 제안하고 있는 컴포넌트 식별 방법을 비교 분석하여, FOCUS 에서는 보다 포괄적인 컴포넌트 식별 방법을 제공하고자 한다.

1. 서론

소프트웨어 산업이 급속하게 발전해감에 따라 정보기술 업체간 경쟁이 더욱 심화되어 소프트웨어 재사용성, 적시성(time to market), 유지 보수성 등이 업체의 생명력으로 대두되면서 컴포넌트 소프트웨어 기술이 점차 각광을 받고 있다. 컴포넌트 소프트웨어 기술은 컴포넌트 단위로 독립적인 개발이 가능하므로 병행적/단계적 소프트웨어 개발을 통해 개발 기간을 단축시킬 수 있다. 컴포넌트의 수행은 컴포넌트 플랫폼 아키텍처상에서 이루어지므로 아키텍처가 제공하는 트랜잭션, 보안성, 지속성 등의 다양한 서비스를 이용할 수 있으며 또한 컴포넌트 단위로 수행되므로 컴포넌트의 대체 및 유지보수가 수월한 장점이 있다.

최근에 산업계에서 쓰이고 있는 대표적인 컴포넌트 기반 소프트웨어 개발 방법론으로는 Rational 의 RUP(Rational Unified Process)[1], Computer Associates 사의 CBD96[2], Compuware 사의 UNIFACE[3], PrincetonSoftech 사의 Select[4] 등이 있으며, 현재 한국전자통신연구원도 FOCUS(Family Oriented Component System Methodology)라는 컴포넌트 기반 개발방법론을 개발하고 있다.[5] 이러한 개발 방법론들은 고유의 컴포넌트 식별 방법이 있으며 컴포넌트 개발을 밀접하게 지원하는 도구를 또한 가지고 있다.

이 논문에서는 이러한 여러 방법론에서의 컴포넌트 식별 방법을 다음과 같은 3 부류로 나누어 비교하였다. 첫 번째는 객체 모델링을 한 후 프로세스를 식별하여 컴포넌트를 식별하는 상향식(bottom-up) 접근법(RUP)이며, 두 번째는 시스템의 개념적인 객체를 식별한 후 주요 객체를 컴포넌트로 매핑하는 하향식(top-down) 접근법(CBD96), 그리고 마지막으로 객체 모델과 유스케이스 모델의 연관성을 테이블화 하여 관련이 많은 객체를 그루핑하여 컴포넌트를 식별하는 접근법(FOCUS)이 있다.

이 논문의 구성은 다음과 같다. 우선 제 2 장에서는 Rational 의 RUP 에서의 컴포넌트 식별방법에 대해 특징을 설명하고 제 3 장에서는 Computer Associates 사의 CBD96 에서 컴포넌트를 식별하는 방법을, 제 4 장에서는 FOCUS 방법론에서 컴포넌트를 식별하는 방법을 설명하고, 마지막으로 제 5 장에서는 결론 및 향후 연구방향을 제시한다.

2. RUP 에서의 컴포넌트 식별 방법

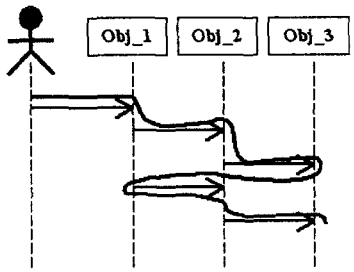
RUP 는 Rational software 사에서 Booch, Rumbough, Jacobson 방법론의 모델링 기법을 통합한 방법론이다. RUP 는 Inception, Elaboration, Construction, Transition 의 4 단계로 구성되어 있으며, 각 단계에서는 요구획득, 분석, 설계, 구현, 테스트가 iteration 이라는 단위로 반복적으로

실행된다. 특징으로는 객체지향(OO), 반복적 개발(iterative), 유스케이스 기반(Use-Case driven), 아키텍처 중심(Architecture centered)적인 개발 등이다.

RUP 에서 컴포넌트를 식별하는 방법은 크게 3 가지로 나누어 볼 수 있다. 첫 번째는 프로세스 설계를 통해 컴포넌트를 식별하는 것(outside-in)이고, 두 번째는 클래스 간의 결합 정도를 기준으로 한 컴포넌트 식별 방법이며, 세 번째는 아키텍처 레벨에서 식별된 서브 시스템을 기준으로 컴포넌트를 식별하는 방법이다.

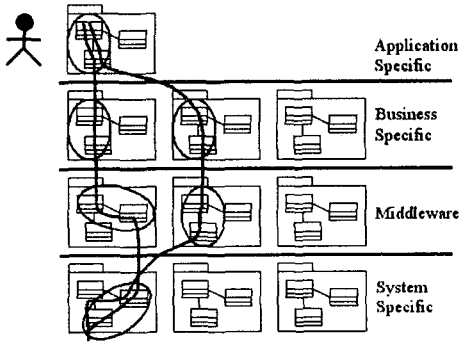
2.1 프로세스 식별을 통한 컴포넌트 식별

이 방법에서는 컴포넌트를 식별하기 위해 우선 각 액터(actor)가 발생시키는 이벤트를 시작점으로 하나의 작업 쓰레드(thread)를 식별한다. 이때 (그림 1)과 같이 순차도(sequence diagram)를 참고하여 작업을 수행한다.



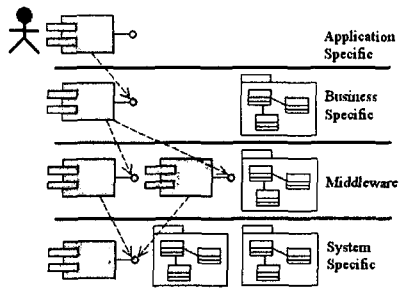
(그림 1) 작업 쓰레드 식별

그리고 나서 액터가 의도한 결과를 얻을 때까지의 진행되는 작업 쓰레드를 시스템의 계층(layering) 원칙에 어긋나지 않는 범위 내에서 그림 2 에서 와 같이 여러 개의 프로세스로 나눈다.



(그림 2) 식별된 프로세스

만약 활동도(activity diagram)를 작성했을 경우에는 활동도 상에서 병렬로 수행 가능한 부분은 쓰레드로 나누어 하나의 프로세스를 구성한다. 이렇게 식별된 프로세스를 기준으로 (그림 3)에서와 같이 하나 혹은 그 이상의 프로세스를 하나의 컴포넌트로 식별한다. 즉, 그 프로세스를 구성하는 각 클래스가 하나의 컴포넌트를 구성하게 된다.

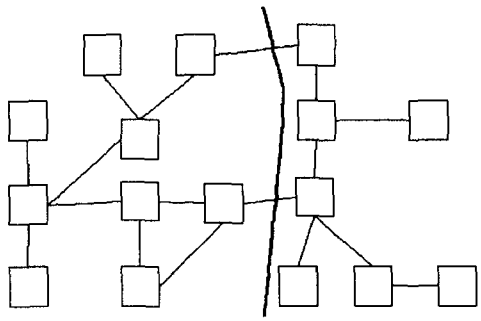


(그림 3) 식별된 컴포넌트

그리고 나서 부가적으로 필요한 클래스들을 그 컴포넌트에 할당한다.

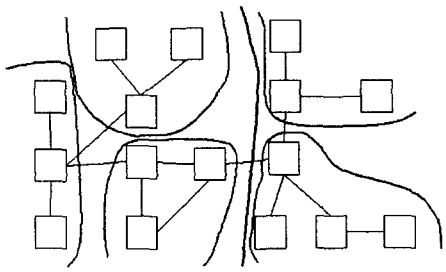
2.2 클래스 간의 결합 정보를 이용한 컴포넌트 식별

이 방법은 우선, 협력도(collaboration diagram)나 클래스도를 참조하여 서로 밀접하게 협력하는 클래스 그룹을 식별한다. (그림 4)는 그 예이다.



(그림 4) 식별된 상호 협력 클래스그룹

일단 클래스 그룹을 식별하고 나면, 이중에서 서로 상호 동작이 없는 클래스들은 분리를 한다. 그리고 나서 최소한의 프로세스로 필요한 서비스들을 해결할 수 있을 때까지 계속해서 위의 두 과정을 반복한 후, 프로세스를 기준으로 컴포넌트를 식별한다. 그 결과는 (그림 5)와 같이 표현될 수 있다.



2.3 서브시스템 위주의 컴포넌트 식별

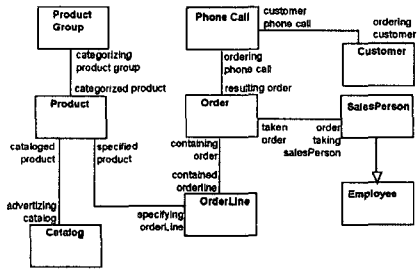
이 방법은 아키텍터(architect)가 시스템을 설계하는 시점에서 구성한 서브시스템을 구현하는 과정에서 컴포넌트로 작성하는 방법이다.

3. CBD96 에서의 컴포넌트 식별 방법

CBD96 은 Computer Associates 사의 개발 도구인 COOL 시리즈를 이용한 컴포넌트 개발 아키텍처 모델이면서, 개발 방법론이라고도 할 수 있다.

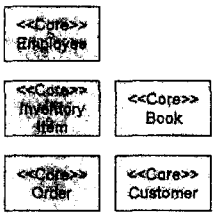
이 방법론에서는 컴포넌트를 모델링할 때 객체 모델링을 어느 정도 진행한 후에 컴포넌트를 식별하는 것이 아니라, 시스템에서 지속적으로 유지 관리를 해야 하는 데이터를 중심으로 컴포넌트를 식별한다. 그 구체적인 절차는 다음과 같다.

우선, (그림 6)에서 같이 비즈니스 타입 모델링을 통해 시스템의 타입 다이어그램을 작성한다.



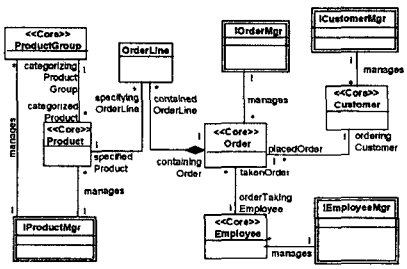
(그림 6) 타입 다이어그램

여기서 타입은 클래스에서 오퍼레이션을 제외한 개념적인 클래스라고 생각할 수 있다. 비즈니스 타입 모델링이 완성되고 나면, 시스템에 의해 지속적으로 정보를 유지 관리 해야 하는 타입을 core 타입으로 선정한다. (그림 7)은 (그림 6)으로부터 5 개의 core 타입을 식별한 것을 나타내고 있다.



(그림 7) Core 타입

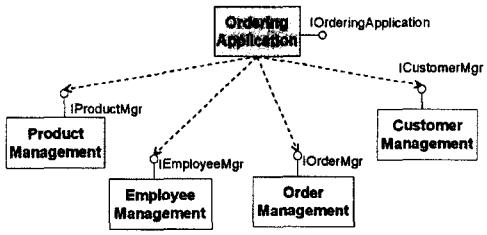
그리고 나서, (그림 8)에서와 같이 core 타입 당 하나의 인터페이스를 추가하고, 그 인터페이스 별로 하나의 컴포넌트를 정의한다.



(그림 8) 인터페이스가 추가된 타입 다이어그램

이때 하나의 core 타입에 하나의 인터페이스가 추가되는 것이 일반적이지만 예외의 경우가 있다. 그 예를 들면, product 와 product group 이라는 core 타입이 있을 경우 product group 은 공통 특성을 갖는 product 의 인스턴스와 함께 있을 때만 비즈니스 상의 의미를 지니게 되므로 이 두 core 타입이 하나의 인터페이스를 공유하도록 설정하게 된다.

그리고 나서 아키텍처 스타일을 정하여 그에 맞는 아키텍처를 구성한다. Radical 구조를 선택할 경우, 그림 9 와 같이 다른 컴포넌트를 접근하기위한 통로 역할이나 제어를 하는 새로운 컴포넌트를 하나 추가하고, 이것이 다른 비즈니스 컴포넌트로의 의존성을 가지도록 설정한다.



(그림 9) 컴포넌트 다이어그램

아키텍처의 스타일로는 Radical 외에 층을 두어 위에서 아래로의 단 방향 의존성만 가지도록 구성하는 계층적(Hierarchical) 스타일, 그리고 컴포넌트 상호 간에 서로 의존성을 가질 수 있는 네트워크(Network) 스타일이 있으며, 그 외의 여러 스타일로 구성할 수도 있다. 그러나 일반적으로 단순한 형태인 Radial 이나 단 방향 의존성만 있는 네트워크 스타일이 권장되고 있다.

식별된 컴포넌트는 이후에 non-core 타입을 구체적으로 정의하고 나서 엔터프라이즈 아키텍처, 성능, 레거시 시스템과의 연계 등을 고려하여 조정을 하게 된다.

4. FOCUS 방법론에서의 컴포넌트 식별 방법

FOCUS 개발 방법론은 현재 한국전자통신연구원에서 숭실대학교와 함께 개발하고 있는 컴포넌트 기반 개발 방법론이다. FOCUS 는 크게 두 부분으로 나눌 수 있는데, 하나는 컴포넌트 자체를 개발하는 절차이고, 다른 하나는 개발된 컴포넌트를 기반으로 하여 시스템을 개발하는 절차이다. FOCUS 에서는 컴포넌트를 개발할 때 다른 방법론들과는 다르게 Family 라는 개념을 도입하여 재사용성이 높은 컴포넌트의 식별에 중점을 두고 있다. 컴포넌트를 개발할 때 그 대상이 되는 응용시스템이 많을수록 개발된 컴포넌트는 재사용성이 높아질 수 있지만, 각 응용시스템별로 가변성이 많아져서 컴포넌트의 개발이 어려워 지거나 구현된 기능이 너무 약할 수 있다. 그 역으로 컴포넌트 개발 대상 응용 시스템이 소수라면 개발된 컴포넌트를 이용하여 개발될 시스템의 수가 적게 되므로 컴포넌트화의 효과가 적다고 볼 수 있다. 컴포넌트를 개발하는 대상 응용 시스템들을 그루핑한 것을 Family 라고 할 수 있으며, 재사용성과 개발 용이성을 고려하여 Family 의 범위를 잘 정할 필요가 있다.

FOCUS 방법론에서 컴포넌트를 식별하는 과정을 개략적으로 설명하면 다음과 같다. 먼저, 컴포넌트 개발 대상이 되는 Family 의 각 멤버인 응용시스템별로 요구사

항을 받아서 Normalize 절차를 거친 후, 유스케이스 모델링을 한다. 두 번째로 유스케이스 모델링과 병행하여 Family 에 대해 객체 모델링을 수행한다. 세 번째로, <표 1>과 같이 유스케이스와 클래스의 친밀도를 보여주는 UDA(Use case Data Access) 표를 작성한다.

<표 1> UDA 표

	C1	C2	C3	C4	C5	C6
U1	C	R	W			R
U2	R	R	C	C		
U3	R	C				C
U4				R	C	
U5				R	D	
U6			R	R		
U7	R	R			R	R
U8	R	R	W	D		

접근 관계는 생성(create), 삭제(delete), 읽기(read),쓰기(write)로 표시한다. 예로 유스케이스 U1 에서는 클래스 C1 을 생성하고, 클래스 C2 및 C6 에 대해 읽기를 하며, 클래스 C3 에 대해 쓰기를 수행한다. 이렇게 작성된 UDA 표는 다시 다음과 같은 절차를 거쳐 조정을 하게 된다.

1. 클래스를 참조(read)만 하는 유스케이스의 제거한다.
2. 여러 유스케이스에서 공통으로 쓰이는 유스케이스(베이스 유스케이스)의 제거한다.
3. 베이스 유스케이스에서 생성하는 클래스(베이스 클래스)의 제거한다.
4. CDRW 에 대해 상수 값을 정한 후, 아래의 계산식 (1)을 이용하여 유스케이스와 클래스 간의 친밀도 값을 계산한다.

A(C) = 10, A(D) = 7, A(W) = 5, A(R) = 3 이라고 하면,

$$M(U_i, C_j) = A(U_i, C_j) / \sum_{1 \le k \le m} A((U_i, C_k)$$

이때, $1 \le i \le n$, $1 \le j \le m$

(1)

이 것을 이용한 결과는 아래 <표 2>와 같다.

<표 2> 친밀도 계산 후의 UDA 표

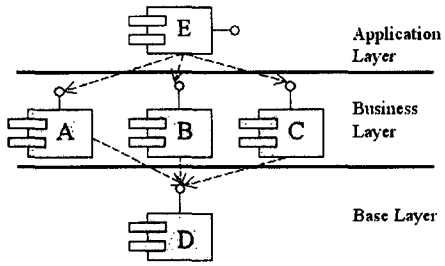
	C1	C2	C3	C4	C5	C6
U1	0.48	0.14	0.24	0	0	0.14
U2	0.12	0.12	0.38	0.38	0	0
U3	0.14	0.43	0	0	0	0.43
U4	0.19	0	0	0.19	0.62	0
U8	0.17	0.17	0.28	0.38	0	0

여기에 임계 값을 0.35 로 설정하고, 유스케이스 및 클래스를 관련 있는 것끼리 클러스터링(clustering)을 하면 다음 (그림 10)과 같이 컴포넌트 후보가 식별된다. 임계 값으로 여러 값을 적용해서 적절한 컴포넌트 후보가 나올 때까지 클러스터링 작업을 반복 한다. 이렇게 후보 컴포넌트가 나오면 추가적인 클래스를 찾아서 컴포넌트를 정제한다.

Component				
A	B		C	D
	C3	C4	C2	C6	C5
	U2	U2	U3	U3	U4
		U8			

(그림 10) 후보 컴포넌트

이후에 (그림 11)에서와 같이 Base 계층, Business 계층, Application 계층으로 구분되도록 컴포넌트를 구조화 하여 컴포넌트 다이어그램을 작성하고 컴포넌트의 인터페이스를 정의한다.



(그림 11) 컴포넌트 다이어그램

이상에서 살펴봤듯이 FOCUS 의 컴포넌트식별 방법은 타 방법론에 비해 다소 복잡하지만, 친밀도의 계산이나 클러스터링 과정에서 고유 알고리즘을 적용하고 있으므로 컴포넌트의 식별에서 어느 정도 자동화할 수 있는 부분이기 때문에 도구가 지원된다면 편리할 것이다.

5. 결론 및 향후 연구 방향

본 논문에서는 현재 산업계에서 쓰이고 있는 여러 방법론 및 한국전자통신연구원에서 개발하고 있는 컴포넌트 기반 개발방법론(FOCUS)에서 제안하고 있는 컴포넌트 식별 방법을 비교 분석하였는데, 위에서 살펴 본 세 가지 부류의 컴포넌트 식별 방법은 각각 고유의 특성이 있다.

향후 연구는 이들 컴포넌트 식별 방법을 개발자들이 쉽게 이해하고 적용할 수 있도록 지침을 만들고, 이를 개발하고 있는 FOCUS 방법론에 반영할 예정이다.

참고문헌

[1] Rational Software Corporation, *Rational Unified Process*, http://www.rational.com/products/rup/index.jsp
[2] Computer Associates, *CBD96*, http://www.sterling.com/cbdedge1/cbd96.htm
[3] CompuWare Corporation, *UNIFACE Development Methodology*, http://www.compuware.com/
[4] PrinceTonSoftech, *Select Perspective*, http://www.princetonsoftech.com/index.asp
[5] 김수동, *컴포넌트 개발 방법론*, 한국전자통신연구원 위탁과제 중간보고서, 2000.