계약관계 중심의 컴포넌트 모델링을 위한 UML의 확장

김경민*, 김태웅, 김태공, 최항묵
인제대학교 전산학과
e-mail:(kmkim, twkim, ktg, hmchoi)@cs.inje.ac.kr

Extension of UML for Components Modeling Focusing on Contract Relationship

Kyung-Min Kim, Tae-Woong Kim, Tae-Gong Kim, Hang-Mook Choi
Dept of Computer Science, Inje University

요 약

컴포넌트 개발은 시스템을 이해하고 분석하기 위한 컴포넌트 모델링 방법이 중요한 부분을 차지하고 재사용성을 높이는 방안으로써 받아들여지고 있다. 컴포넌트는 일반적으로 컴포넌트를 개발하는 사람과 그것을 조립하여 사용하는 사람이 다르며 이에 따른 계약의 내용과 목적이 다르기 때문에 대규모의 컴포넌트 시스템을 개발하는데 있어서 그 역할에 따라 두 가지 유형으로 구분하는 것이 중요하다. 이것이 현실화되기 위해서는 명확한 컴포넌트와 인터페이스 명세가 필요하며 조립자와 개발자 간의 서로 다른 관점에서의 컴포넌트 모델이 필요하다. 이에 본 논문에서는 컴포넌트 조립자와 개발자의 계약관계에 기반하여 조립자 관점의 컴포넌트 모델과 개발자 관점의 컴포넌트 모델을 정의하며 이를 위해 UML을 확장한다. 그리고 이를 적용하여 그 효용성을 검토한다.

1. 서론

소프트웨어 컴포넌트란 하나 이상의 기능을 갖는 독립적인 소프트웨어이며, 조립을 통해 응용프로그램을 작성할 수 있는 부품 형태의 소프트웨어를 말한다[1]. 이런 컴포넌트 개발은 시스템을 이해하고 분석하기 위한 컴포넌트 모델링 방법이 중요한 부분을 차지하고 재사용성을 높이는 방안으로써 받아들여지고 있다. 컴포넌트 모델링 방법은 개발 초기 단계의 부정확한 산출과 불충분한 모델 구축의 문제를 해결할 수 있는 방법으로 간주되고 있으며, 개발 노력과 유지보수 비용을 줄일 수 있다.

이러한 컴포넌트 기반의 모델들은 UML에서 자체 제공하는 모델들뿐 아니라 UML을 기반으로 하는 Catalysis, Unified software development process[2], CBD96[3] 등에서 제공하는 모델들이 있다. 그러나 이러한 모델들에서는 소프트웨어 개발 프로세스의 각 작업에 대하여 체계적이면서 구체적인 지침들을 충분히 제시하지 않고 있다[4]. 게다가 컴포넌트는 일반적으로 컴포넌트를 개발하는 사람과 그것을 조립하여 사용하는 사람이 다르며 이에 따른 계약의 내용과 목적이 다르기 때문에 대규모의 컴포넌트 시스템을 개발하는데 있어서는 그 계약 역할에 따라 두 가지 유형으로 구분하는 것이 중요하다.

여기에서 계약은 명확한 양식을 가지고 약정의 상세한 사항을 기록하며 각 이해당사자의 상대방에 대한 서비스 제공과 같은 책임과 의무를 기술하는 것을 말한다[5]. 소프트웨어 설계의 세계에서는 이미 계약에 의한 설계(Design by Contract) 개념이 객체지향 영역에서는 Bertrand Meyer[6]에 의해서 광범위하게 개발되었고 컴포넌트의 세계에서는 Catalysis[7]에 의해서 개념이 입증되었다.

이것을 기반으로 하여 조립자와 개발자 간의 서로 다른 계약 역할에 따라 두 가지 유형의 컴포넌트 모델링을 위해 UML을 확장한다.

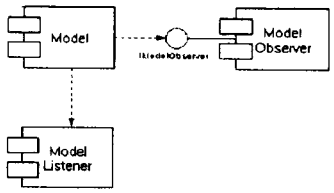
본 논문에서는 조립자 관점의 컴포넌트 모델을 위해 서비스와 조립에 관한 컴포넌트 인터페이스 정보를 표현하고 개발자 관점의 컴포넌트 모델을 위해 컴포넌트 전체 상호작용 관계와 서비스의 내부 비즈니스 로직, 객체 상호작용에 관해 표현한다. 이를 위해 UML을 확장하며 이를 적용하여 그 효용성을 검토한다.

2. 관련연구

2.1 UML에서의 컴포넌트 모델

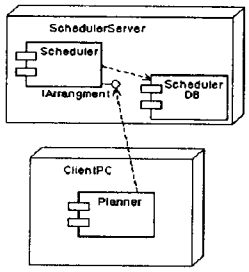
UML은 객체지향 분석과 설계를 위한 모델링 언어로 표기하려는 대상을 다이어그램을 사용하여 나타내고 그 대상에 의미를 부여한다. UML에 의한 컴포넌트 명세는 기존 모델링 기술에 스테레오 타입, 인터페이스 클래스, 실현관계를 추가하고 이를 이용하여 컴포넌트를 기술하게 된다[8].

컴포넌트 다이어그램은 컴포넌트 사이의 의존성과 구조를 모델링 하여 컴포넌트간의 상호작용을 표현할 수 있도록 한다.



(그림1) UML에서의 컴포넌트 다이어그램

배치 다이어그램은 컴포넌트 기반 소프트웨어 동작환경 등을 나타내며, 컴포넌트를 포함한 시스템의 물리적 구조를 나타냄으로써 시스템에 대한 이해를 증진시킨다.



(그림2) UML에서의 배치 다이어그램

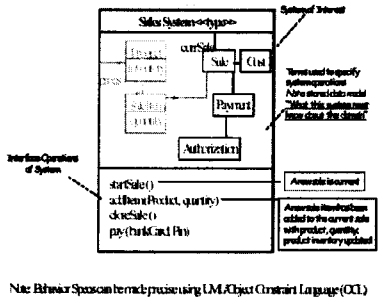
컴포넌트 기반의 개발과정에서 필수적이라 할 수 있는 인터페이스에 대한 표기방법으로 UML에서는 클래스 다이어그램을 사용하여 나타낼 수 있다. 이것은 일반적인 객체에 적용되는 클래스 다이어그램의 변형으로써 클래스 다이어그램에 <<Interface>> 스테레오타입을 추가한다[9].

UML은 소프트웨어의 설계를 위한 언어지만 모든 상황에 대한 의미를 표현하기 위해서는 충분하지 않다. 그리고 컴포넌트의 특성인 인터페이스에 대한 표현이 단순하고 제한된 표현방법을 사용하기 때문에 정확한 컴포넌트의 표현에 한계가 있다.

2.2 Catalysis에서의 컴포넌트 모델

Catalysis는 UML을 사용하여 객체와 컴포넌트 기반 소프트웨어 개발을 지원하는 방법론으로 도메인/비즈니스 레벨, 컴포넌트 명세 레벨, 내부 설계 레벨의 세 레벨로 구성된다[10]. 도메인/비즈니스 레벨에서는 문제 도메인 용어를 설정하고, 비즈니스 프로세스를 이해한다. 컴포넌트 명세 레벨에서는 컴포넌트의 책임, 컴포넌트와 시스템의 인터페이스를 정의하며, 컴포넌트의 오퍼레이션을 명세화 한다. 컴포넌트 구현 레벨에서는 내부 아키텍처를 정의하고, 시스템과 컴포넌트의 내부를 설계한다.

또한 이 세 가지 단계를 통해 정적모델, 행위모델, 상호작용모델을 구성한다[11]. 정적모델은 객체의 속성을 표현하고, 행위모델은 객체의 행위를 표현한다. 상호작용모델은 객체의 협력관계를 표현한다. 이러한 세 모델을 통해 객체에 대한 협력관계가 완성된다. 협력관계에 대한 반복 작업과 보완 작업을 거쳐 소프트웨어 컴포넌트가 정의된다.



(그림3) Catalysis에서의 컴포넌트 정의

Catalysis 방법에서는 컴포넌트라는 단위를 type이란 용어를 사용하여 정의하고[12] 객체에 대한 클래스 또한 타입이란 용어를 사용한다. 즉 객체 타입을 통해 컴포넌트 타입을 정의하는 것이다.

Catalysis는 대부분의 현재의 컴포넌트 개발 방법론과 마찬가지로 추상적이고 일반적인 틀만을 제시하고 있기 때문에 실제로 소프트웨어를 개발하는 것과는 많은 차이가 있다.

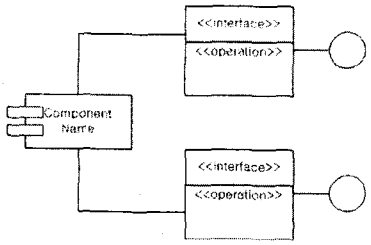
3. UML 확장

기존의 컴포넌트 모델들은 많은 부분으로 구성되어 있지만 이것들 전부가 컴포넌트를 조립하는 사람과 관련을 갖는 것은 아니며, 또한 실제 이 모델을 기반으로 구현하려는 사람에게는 부족한 정보들이 많다. 이에 본 논문에서는 계약 역할에 따라 3.1과 3.2처럼 두 가지 유형으로 UML을 확장한다.

3.1 조립자 관점의 컴포넌트 모델

조립자 관점의 컴포넌트 모델에서는 컴포넌트에서 제공하는 서비스와 이들 간의 조립에 관한 컴포넌트 인터페이스 정보를 표현하기 위해 그림4와 같이 정의한다.

이 모델은 단순히 컴포넌트를 호출해서 사용하는 조립자가 알아야 하는 정보를 나타낸다. 여기에는 호출에 관한 정보로써 컴포넌트 이름과 인터페이스 이름, 해당 인터페이스 내의 오퍼레이션 이름/인자타입/리턴타입 정보들을 포함한다.



(그림4) 조립자 관점의 컴포넌트 모델

이것은 자세한 비즈니스 로직은 숨긴 채 컴포넌트 인터페이스를 통한 서비스를 가능하게 한다.

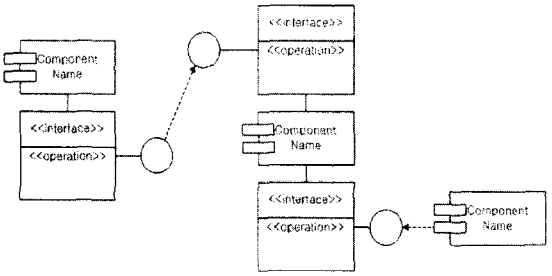
3.2 개발자 관점의 컴포넌트 모델

개발자 관점의 컴포넌트 모델에서는 컴포넌트의 인터페이스 구현물이 자신의 책임을 완수하기 위해 다른 컴포넌트와 어떤 상호작용을 하는지 이에 관한 컴포넌트들 간의 관계들과 서비스의 내부 비즈니스 로직과 그들 객체 간의 상호작용에 관한 정보를 표현한다.

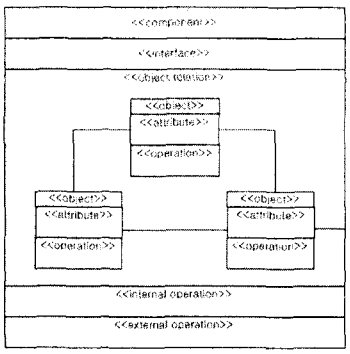
실제 개발자에게 필요한 모든 정보를 포함하기위해 두 단계의 모델로 나타낸다. 그림5와 같이 1단계에서는 개발하려는 컴포넌트의 전체적인 구조와 컴포넌트들 간의 상호작용을 나타내며, 그림6과 같이 2단계에서는 실제 컴포넌트의 인터페이스를 구현하기 위한 오퍼레이션들과 속성들, 그들 객체간의 관계들을 나타낸다. 포함하는 정보들을 다음과 같다.

(1단계) 컴포넌트 이름, 컴포넌트에서 제공하는 인터페이스 이름, 이에 따른 오퍼레이션 이름들, 컴포넌트에서 필요로 하는 인터페이스이름, 이에 따른 오퍼레이션 이름들, 이들 컴포넌트 간의 관계

(2단계) 컴포넌트 이름, 인터페이스 이름, 외부에서 사용이 가능한 오퍼레이션 이름/인자타입/리턴타입, 내부적으로 사용되는 오퍼레이션 이름/인자타입/리턴타입, 속성 정보들, 그들 간의 객체 관계



(그림5) 개발자 관점의 1단계 컴포넌트 상호작용 모델



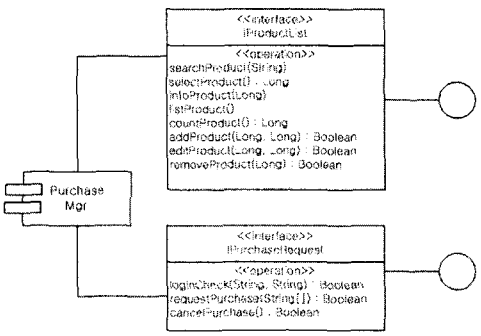
(그림6) 개발자 관점의 2단계 인터페이스 내부 모델

이것은 개발자에게 필요한 모든 정보를 두 단계의 모델로 구체적으로 나타냄으로써 실제 구현 시 모델의 이용 효율을 높인다.

4. 사례연구

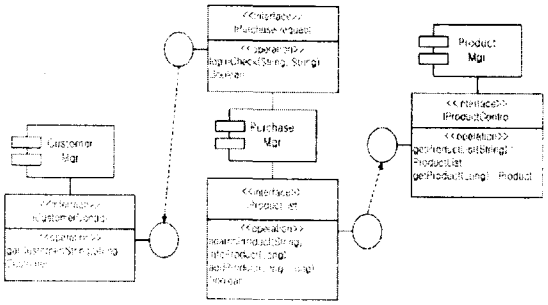
본 연구에서 제안하는 조립자와 개발자 관점의 두 가지 유형의 컴포넌트 모델을 구매발주 시스템 중 구매 컴포넌트에 적용해 보았다.

그림7은 조립자 관점의 컴포넌트 모델을 위해 구매 컴포넌트(PurchaseMgr)에서 제공하는 상품리스트 인터페이스(IProductList)와 구매요청 인터페이스(IProductRequest)의 외부에서 사용이 가능한 오퍼레이션 목록들을 정의한다.

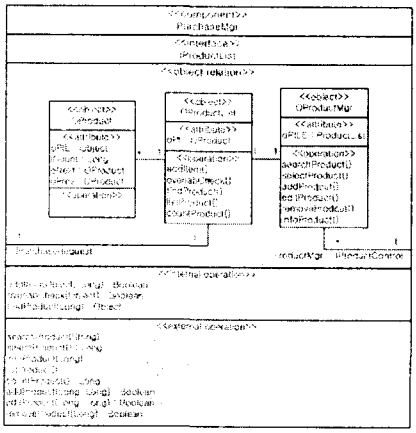


(그림7) PurchaseMgr 컴포넌트의 조립자 관점의 컴포넌트 모델

구매 컴포넌트에 대한 개발자 관점의 컴포넌트 모델은 그림8,9와 같다. 그림8은 개발자 관점의 1단계 컴포넌트 상호작용 모델로써 구매 컴포넌트의 상품리스트 인터페이스와 구매요청 인터페이스가 각각 고객 컴포넌트(CustomerMgr)의 고객관리 인터페이스(ICustomerControl)와 상품 컴포넌트(ProductMgr)의 상품관리 인터페이스(IProductControl)와 관계가 있음을 보여준다.



(그림8) PurchaseMgr 컴포넌트의 개발자 관점의 1단계 컴포넌트 상호작용 모델



(그림9) PurchaseMgr 컴포넌트의 개발자 관점의 2단계 인터페이스 내부 모델

그림9는 그림8에서 구매 컴포넌트의 상품리스트 인터페이스에 대한 2단계 인터페이스 내부 모델을 나타낸다. 실제 구현에 필요한 구체적인 정보들로 외부에서 사용이 가능한 오퍼레이션(external operation)들과 내부적으로 사용되는 오퍼레이션(internal operation)들, 그들 간의 객체 관계(object relation)를 포함한다.

5. 결론

본 연구에서는 컴포넌트 기반 소프트웨어 개발 모델링을 효과적으로 하기위해 계약 역할에 따라 조립자 관점의 컴포넌트 모델과 개발자 관점의 컴포넌트 모델의 두 유형으로 UML을 확장하였다. 또한 이를 적용하여 그 효용성을 검토해 보았다.

이를 통해 조립자는 컴포넌트 내부의 자세한 비즈니스 로직은 숨긴 채 컴포넌트 인터페이스를 통한 서비스가 가능하며, 개발자는 필요한 모든 정보를 두 단계의 모델로 구체적으로 나타냄으로써 실제 구현 시 모델의 이용 효율을 높인다. 또한 이렇게 분리된 두 가지 유형으로 정의함으로써 개발자 계약조건에 대한 변경이 조립자 계약의 변경을 수반하지 않게 되므로 조립자에게 어떠한 영향도 주지 않게 되어 컴포넌트 변경에 쉽게 대응할 수 있게 한다.

향후 컴포넌트 기반 소프트웨어 개발 모델에 좀더 다양한 형태의 정적인 모델과 동적인 모델에 대한 연구가 진행되어야겠다.

참고문헌

[1] Dedmond F.D'Souza, Alan C. Wills, "Object, Component and Frameworks With UML", ADDISON-WESLEY, 1998

[2] Jacobson, I., Booch, G., and Rumbaugh, J., "The Unified Software Development Process", Addison-Wesley, 1999

[3] Sterling, "The CBD96 Standard Versin 2.1", Sterling, 1998

[4] 김수동, "컴포넌트 정의 및 관련 기술 동향", 소프트웨어 공학회지 12권 3호 pp.5-18, 1999

[5] John Cheesman, John Daniels, "UML Components : A Simple Process for Specifying Component-Based Software", Addison-Wesley, 2000

[6] Meyer, B., "Object-Oriented Software Construction (2nd ed.)", Prentice Hall, 1997

[7] D'Souza, D. F., and A. C. Wills., "Object, Components and Frameworks with UML: The Catalysis Approach", Addison-Wesley, 1999

[8] John E.Mann,"Rules for E-Business", Avaliable by webserver from http://www.psgroup.com, 2000

[9] Grady Booch, James Rumbaugh, Ivar Jacobson, "The Unified Modeling Language Reference Manual", Addison-Wesley, 1998

[10] D'Souza, D. and Wills, A. C., "Objects, Frameworks, and Components with UML", Addison-Wesley, 1998

[11] Martin Fowler with Kendall Scott, "UML Distilled Applying the Standard Object Modeling Language", Addison-Wesley, 1997

[12] Wojtek Kozaczynski, "Composite Nature of Component", ICSE99, 1999