컴포넌트 추출을 위한 UML 기반의 역할 모델링에 관한 연구

송 호 영† · 김 정 종††

요 약

컴포넌트 소프트웨어 개발에서 양질의 컴포넌트를 추출하기 위해서는 사용자의 요구사항을 정확하게 표현할 수 있는 체계적인 모델링 방법이 필요하다. 그러나 컴포넌트는 일반적으로 객체 모델링을 기반으로 설계되고 개발되기 때문에 상호작용과 협력의 표현, 상속으로 인한 재사용 문제 등을 해결하는데는 한계가 있으며 복잡한 구조를 가진 대규모 시스템에서 컴포넌트를 추출하기에는 용이하지 않다. 따라서 객체 모델링을 보완하기 위하여 객체 중심이 아닌 객체의 역할을 중심으로 하는 역할 모델링 기법을 활용할 필요가 있다. 역할 모델링은 객체들의 패턴을 추상화하고 복잡한 대규모 시스템을 관계의 분리를 통하여 간단한 모델로 생성할 수 있도록 객체의 역할을 중심으로 모델링하는 기법이다. 본 논문은 UML 기반에서 역할 모델링을 통하여 컴포넌트를 추출하는 방법을 연구한다. UML은 다양한 관점으로 모델을 표현하고 있으므로 다양한 관점의 역할 모델을 표현하여 컴포넌트를 추출하는데 활용할 수 있도록 한다.

A Study on UML-Based Role Modeling for Extracting Components

Ho Young Song† · Jung Jong Kim††

ABSTRACT

Systematic modeling methods, which can correctly represent user requirements, are absolutely required to extract components when developing components. But, components are designed and developed in general based on the concept of object modeling so that they lacks in supporting representing cooperation and interaction as well as reuse. It means extracting components is not easy to be applied to complicated and sophisticated large-scale systems. This paper utilizes a role modeling method based on the role of objects, not on objects themselves, to complement object modeling. The Role modeling can break down a complicated system into simple models through abstracting the pattern of objects based on roles of the objects. In this study, an UML-based method extracting components through role modeling is embodied. UML can represent models from various standpoints so that role models are also viewed from several points of view.

키워드 : 컴포넌트(Component), 역할(Role), 역할 모델링(Role Modeling), UML

1. 서 론

객체지향 소프트웨어 개발 방법은 객체를 중심으로 한 객체 모델링을 기반으로 시스템을 개발한다[1]. 이는 실세계의 현상들을 보다 명확하게 모델링할 수 있기 때문에 개발의 용이성과 분석, 설계, 재사용성, 확장성 등의 많은 장점들을 가지고 있으나 객체나 클래스를 개념으로 한 객체 모델링의 방법으로는 재사용과 생산성을 극대화하기에는 부족한 면이 있다. 또한 복잡한 문제를 가진 대규모의 시스템에는 적합하지 않으며 상호작용과 협력의 표현, 상속으로 인한 재사용 문제 등을 해결하는 데는 한계가 있다[2, 3].

양질의 컴포넌트를 추출하기 위하여 사용자의 요구사항을 정확하게 표현할 수 있는 체계적인 모델링 방법이 필요하며 대규모의 복잡한 시스템에서 컴포넌트를 추출하기 위해서는 시스템의 분해가 필요하다. 그러나 컴포넌트 추출도 객체 모델링을 기반으로 하고 있기 때문에 객체 모델링이 가지고 있는 한계로 인하여 새로운 기법의 컴포넌트 추출 방법이 요구된다.

따라서 객체 모델링 방법을 보완하고 개선할 수 있는 방법으로 객체가 아닌 객체의 역할을 중심으로 하는 새로운 추상화 기법인 역할 모델링 방법이 제안되었다. 역할 모델링은 객체들의 패턴을 추상화하고 복잡한 대규모 시스템을 관계의 분리를 통하여 간단한 모델로 생성할 수 있도록 한다[4, 5].

기존의 역할 모델링에 관한 연구들은 역할 모델을 표현하는데 있어서 아직 표준화된 표현법이 없어 다양한 표현법으로 역할 모델을 표현하고 있다[5-7].

본 논문은 역할 모델링을 통하여 생성된 역할 모델을 UML에서 표현하는 방법을 연구한다. 물론 UML이 객체

※ 본 연구는 2003년도 경남대학교 학술논문게재 연구비 지원으로 이루어졌음.
† 준 회 원 : 창원전문대학 컴퓨터정보처리과 연구교수
†† 종신회원 : 경남대학교 컴퓨터공학부 교수
논문접수 : 2004년 2월 11일, 심사완료 : 2004년 4월 14일

모델링을 위한 표준으로서 다양한 관점으로 시스템을 모델링하며 역할 개념도 반영하고 있지만 이들 역할 개념들이 요구 사항을 충족시키지 못하고 있다. 따라서 본 연구는 다양한 관점으로 역할 모델을 표현할 수 있도록 UML을 기반으로 역할 모델을 표현한다. 또한 이들 역할 모델들을 바탕으로 컴포넌트를 추출한다. 하나의 역할 모델은 클래스 집합이나 하나의 컴포넌트 단위와 대응할 수 있으므로 컴포넌트를 추출하는데 있어 보다 효율적이고 용이하다.

2. 관련 연구

2.1 객체 모델링

객체 모델링 방법은 소프트웨어 공학에서 추구하는 신뢰성 향상, 유지보수의 용이, 성능의 향상 등의 많은 장점들을 제공하고 있으며 실세계의 현상들을 보다 명확하게 모델링할 수 있기 때문에 개발의 용이성과 분석, 설계, 재사용과 확장성 등의 장점을 가지고 있다[1].

객체 모델링 방법이 이런 많은 장점을 가지고 소프트웨어를 개발하는데 있어 각광을 받아오고 있지만, 객체나 클래스 개념으로는 재사용과 생산성을 극대화하기에는 부족한 면이 있으며, 몇 가지 문제점을 나타내고 있다[2, 3, 8].

2.1.1 객체의 문제점

객체의 상태 영역과 행위는 객체가 생성될 때 결정된다. 애트리뷰트와 메소드들은 그것의 생명주기 동안 객체에 추가되거나 혹은 객체로부터 제거되도록 허용되지 않는다. 즉, 객체가 그것의 타입을 동적으로 변경하는 것이 불가능하다. 또한 객체는 객체의 모든 애트리뷰트와 메소드들이 연관관계에서 다른 객체들에게 드러난다. 따라서 특정 애트리뷰트와 오퍼레이션에 대한 가시성을 제한하는 수단이 필요하다.

2.1.2 상호작용과 협력 표현의 문제점

컴포넌트는 작업을 수행하거나 목표를 달성하기 위하여 서로 협력할 필요가 있다. 그러나, 객체 모델링 방법은 객체의 전체 집합 보다는 각 객체에 너무 초점을 맞춘다. 각 객체에 초점을 맞추는 것은 혼란을 가져올 수 있고 양질의 컴포넌트를 생성할 수 없다. 객체는 하나의 추상화 단위인 어플리케이션에서, 적절한 하위집합 혹은 하위시스템을 추출하고 패키지하기가 매우 어렵다. 따라서, 재사용 가능한 컴포넌트를 추출하는데 있어서 주요 문제는 각 객체 레벨에서 상호작용하고 협력하는 개체의 하위집합 수준으로 초점을 이동하는 것이 필요하다.

2.1.3 상속의 문제점

상속은 개발자가 상속하고 오버라이드할 대상을 결정하는 것을 어렵도록 한다. 또한 하위 클래스가 모든 상위 클래스의 특성과 행위를 얻는다는 것을 의미하기 때문에 시스템의 분해를 어렵게 한다.

2.1.4 복잡한 관계 구조로 인한 문제점

객체 모델링은 시스템 전체가 객체를 중심으로 결합되어 있으므로 복잡한 구조를 가진 대규모 시스템을 모델링하는 것은 쉽지가 않을 뿐만 아니라 이해하기도 어렵다. 또한, 복잡한 관계 구조로 인하여 추가적인 요구 사항에 대처가 쉽지 않다.

이런 여러 문제점들로 인해 새로운 추상화 방법으로서 객체 중심이 아닌 역할을 중심으로 한 모델링 방법이 필요하다.

2.2 역 할

클래스의 개념처럼 역할은 객체 집합의 기술이다. 그러나 클래스와 다른 점은 클래스는 일반 특성들을 나타내는 객체 집합을 기술하지만 역할은 객체 패턴에서 같은 위치에 있는 객체 집합을 기술한다. 역할은 특정 문맥에 참여하는 개체의 행동이다. 즉, 표현된 하나의 추상개념이다. 역할은 하나의 개체에 대하여 관점에 따라 서로 다른 역할을 가질 수 있다[5, 9].

한 객체가 특정 역할을 수행할 때 역할이 표현되고 고객들은 객체와 교류하면서 객체가 수행하는 역할에 따른 특정한 행동을 기대한다. 또한, 역할은 객체의 속성과 오퍼레이션에 의해 행위를 기술한다. 역할에 의하여 정의된 행위는 객체가 어떻게 행동할지를 명확하게 하기 위하여 이용된다.

다음은 역할 개념에 대한 특성이다[3, 8].

- 객체의 다양한 역할은 공통적인 구조와 행위를 공유할 수 있다.
- 역할은 특별한 환경에 대한 접근을 제한한다.
- 역할들은 동적으로 추가되거나 삭제될 수 있다.
- 역할들은 독립적으로 획득될 수 있고 제거될 수 있다.
- 역할의 몇몇 인스턴스들이 동시에 하나의 객체에 대해 존재할 수 있다.

역할의 사용은 객체들이 변화와 확장에 잘 적응할 수 있도록 하고, 관련 속성들의 집합으로 그룹화하여 더 조직적으로 만들기 때문에 재사용성을 향상시킨다.

2.3 역할 모델

역할 모델은 객체 모델에서 객체의 패턴을 추상화한 모델이며 객체 구조가 적절한 역할들을 수행함으로서 주어진 관계 영역을 완성하는 방법을 기술한다[5, 9, 10]. 역할 모델은 관계의 분리를 지원하고 하나의 결합(coherent) 모델에서 실세계 현상의 정적·동적 속성들을 기술한다. 이 관계의 분리는 상호작용하는 객체에 대해 대규모의 복잡한 구조를 가진 시스템에서 복잡한 현상을 많은 하위 현상으로 분할되는 것을 말하며 각 하위 현상은 자신의 역할 모델에 의해 기술된다.

현상은 많은 협력하는 객체들에 의해 기술된다. 하위 현

상은 그들의 관계 영역에 의해 명시되고, 하위 현상을 기술하는 객체들은 객체의 패턴에서 구성되며, 패턴에서 같은 위치를 가지는 모든 객체들은 역할에 의해 표현된다.

역할 모델은 다른 종류의 현상을 식별하고, 객체의 이점을 간직한 채 그대로 분리하는 것을 허용하기 때문에 객체지향 패러다임의 강력한 확장이다.

2.4 UML

UML(Unified Modeling Language)은 Rumbaugh의 OMT 방법론, Booch의 Booch방법론, Jacobson의 OOSE 방법론과 기타 다른 전문가들의 주장을 통합하여 만든 모델링 개념의 공통집합으로 객체지향 분석 및 설계 방법론의 표준 지정을 목표로 제안되었으며 OMG에 의해 객체지향 모델링 언어의 산업 표준으로 승인되어 현재 널리 사용되고 있다[11, 12]. 즉, UML은 객체지향 분석과 설계 영역에서 사용되는 표준화된 모델링 언어이다. UML은 배우기 쉽고 확장하기도 쉬우며 다양한 표기법을 지니고 있어서 여러 방법론의 특징을 잘 표현할 수 있다. 또한 표준화된 언어이기 때문에 분석, 설계, 구현 과정에서 발생하는 개발자간의 의사소통의 불일치를 해소할 수 있으며 모델링에 대한 표현력이 강하고 비교적 모순이 적은 논리적인 표기법을 가진 언어이다.

3. 역할 모델링

역할 모델링은 역할을 중심으로 객체들의 관계에 따른 새로운 추상화 방법으로서 객체가 수행하는 역할을 기반으로 모델링하는 방법이다.

본 연구에서는 역할 모델링을 통하여 생성된 역할 모델을 다양한 관점으로 표현할 수 있도록 UML을 기반으로 역할 모델을 표현한다. 이 UML 기반의 역할 모델링은 객체 모델링의 한계를 극복할 수 있으며, 역할 모델링의 표준으로 제공될 수 있다.

3.1 역할 모델링 단계

UML에서 역할 모델링을 통하여 다양한 관점의 역할 모델을 생성하고 컴포넌트를 추출하기 위하여 9단계를 제안한다. 각 단계는 실행 가능한 일로 끝나며 역할 모델 상에 관점이다. 이들 관점에 대한 상대적 중요성은 모델의 목적에 달려있다. 단계들은 역할 모델이 충분히 정의될 때까지 반복하여 수행된다.

다음은 역할 모델링의 단계이다.

① 관계 영역을 결정 : 고려중인 문제에 대하여 자유로운 형식으로 기술한다. 즉, 사용자와 개발자간의 의사소통을 통하여 요구사항을 분석하고 필요한 사항들을 기술한다.

② 문제를 이해하고 객체를 식별 : 주어진 관계 영역에 대하여 객체들과 객체들의 성질(속성)을 식별한다. 그리고 객체들의 연관관계를 설정하여 객체 모델을 작성한다.

③ 객체 패턴을 식별 : 생성한 객체모델에서 시나리오를 바탕으로 객체 패턴을 식별한다.

④ 역할들을 식별 : 식별한 객체 패턴을 바탕으로 역할들과 역할들의 성질(속성)들을 식별한다. 즉, 객체들의 작업과 책임을 분리하고 인식하여 객체들을 시스템에서 수행하는 역할들로 기술한다. 인식한 역할들을 기반으로 역할들의 관계를 설정하여 역할 클래스 다이어그램을 생성한다. 또한 역할 유스 케이스 다이어그램을 생성한다.

⑤ 메시지 순서를 결정 : 역할들에 의해 수행된 활동들을 보여주는 시나리오를 생성한다. 생성된 시나리오는 역할들 사이 흐르는 메시지의 순서를 나타내며 이를 바탕으로 역할 순차 다이어그램을 생성한다.

⑥ 협력 구조를 결정 : 협력하는 역할들의 구조를 보여준다. 이들 구조를 역할 협력 다이어그램으로 나타낸다.

⑦ 역할 행위를 결정 : 주요 역할들에 대하여 메시지들에 의해 유도되는 매소드들을 기술한다.

⑧ 인터페이스 명세 : 생성한 여러 다이어그램을 바탕으로 컴포넌트를 위한 인터페이스를 결정하고 명세한다.

⑨ 컴포넌트 명세 : 컴포넌트에 인터페이스를 연결하고 실체화하여 컴포넌트를 명세하며 역할 컴포넌트 다이어그램을 생성한다.

3.2 역할 모델의 표현

모델링을 통하여 생성된 모델은 UML 상에서 다양한 관점의 다이어그램들로 나타낼 수 있다. 따라서 본 연구에서는 UML 상에서 역할 모델에 대한 다양한 관점의 다이어그램을 표현한다.

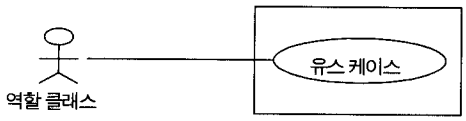
3.2.1 역할 클래스 다이어그램

일반적인 역할 모델에 대한 클래스 다이어그램이다. 다음 (그림 1)은 역할 클래스 다이어그램을 보여준다. 역할 클래스라는 것을 명시하기 위하여 스테레오 타입으로서 <<Role>>으로 표기하며 역할명을 기입한다. 각 역할 클래스는 역할이 가지는 속성과 행위들을 가진다. 또한, 각 역할 클래스들 사이 역할 관계는 연관관계로서 표현한다.

(그림 1) 역할 클래스 다이어그램

3.2.2 역할 유스 케이스 다이어그램

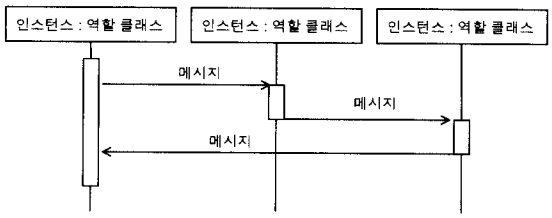
역할 유스 케이스 다이어그램은 액터인 역할 클래스와 유스 케이스 사이 상호작용을 보여준다. 유스 케이스는 역할 클래스가 수행해야 하는 시나리오 집합을 나타낸다. 다음 (그림 2)는 역할 유스 케이스 다이어그램을 나타낸다.



(그림 2) 역할 유스 케이스 다이어그램

3.2.3 역할 순차 다이어그램

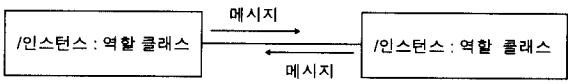
역할 순차 다이어그램은 객체들 사이 상호작용의 시간 순서에 대한 기술이다. 상호작용은 송·수신자 객체에 메시지를 전송하는 이벤트를 나타낸다. 송·수신하는 객체들은 역할 클래스의 인스턴스들로 표현된다. 그러므로 역할 순차 다이어그램은 역할 인스턴스 구조를 통해 흐르는 메시지의 순서를 보여준다. 다음 (그림 3)은 역할 순차 다이어그램을 표현한 것이다.



(그림 3) 역할 순차 다이어그램

3.2.4 역할 협력 다이어그램

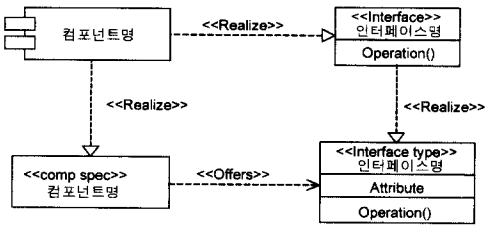
역할 협력 다이어그램은 역할 인스턴스들과 이들 사이 메시지 경로를 보여준다. 즉, 역할 인스턴스들 사이 상호작용하는 관계를 나타낸다. 다음 (그림 4)는 역할 협력 다이어그램을 표현한 것이다. 역할 협력 다이어그램은 협력하는 역할 인스턴스들의 포트를 서로 연결하여 메시지가 어떤 경로를 통하는지에 초점을 둔다. 각 역할 클래스의 인스턴스들 사이 상호작용은 연관관계에 메시지를 첨부하여 나타내며 메시지 흐름은 화살표로서 표현한다.



(그림 4) 역할 협력 다이어그램

3.2.5 역할 컴포넌트 다이어그램

역할 컴포넌트 다이어그램은 컴포넌트, 인터페이스로 구성되어 있으며 각각의 관계를 설정한다. 다음 (그림 5)는 역할 컴포넌트 다이어그램을 나타낸다. 인터페이스는 컴포넌트에 작은 원을 연결하여 나타낼 수 있고 그림과 같이 실체화하여 오퍼레이션들의 집합으로 나타낼 수도 있다. 인터페이스와 컴포넌트에 각각의 명세(인터페이스 타입과 comp spec)를 실체화시킬 수 있다.



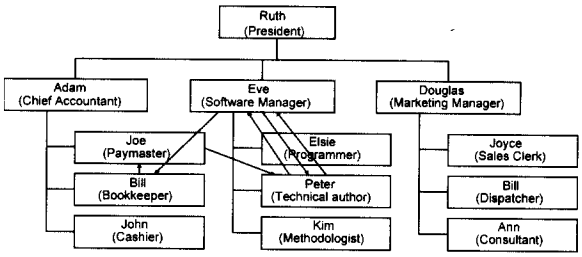
(그림 5) 역할 컴포넌트 다이어그램

4. 사례 연구

사례연구는 회사 사람이 회사 경비로 출장을 갈 때 모델링하는 과정을 보여준다. 먼저, 회사 조직에 대한 객체 모델을 생성하고, 상호작용하는 객체들에 대하여 패턴을 정의하고 추상화하여 UML 기반의 역할 모델에 대한 다양한 다이어그램들을 생성한다.

4.1 객체 모델

다음 (그림 6)은 회사조직을 나타내는 객체 모델이다. 사각형은 객체이고 선은 객체들 사이 관계를 나타낸다. 화살표는 아래 시나리오를 바탕으로 한 흐름으로서 객체의 패턴을 식별할 수 있다.



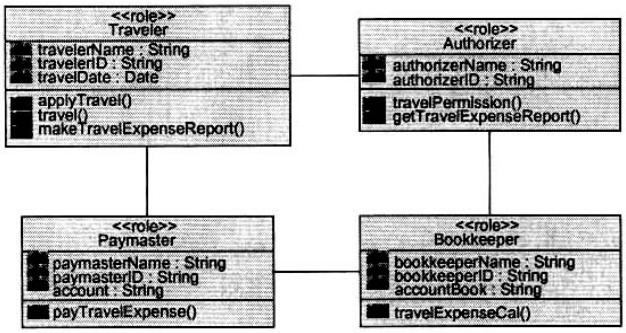
(그림 6) 회사 조직의 객체 모델

Peter라는 사람이 회사 경비로 출장을 갈 때 다음과 같은 시나리오로 처리가 이루어진다.

① Peter는 그의 매니저 Eve에게 출장허가를 요청한다.
② Eve는 예산과 계획을 검사하고 Peter에게 출장을 허가한다.
③ Peter는 티켓을 구매하고 출장을 간다. 그리고 돌아와서 경비 보고서를 작성하고 Eve에게 보고서를 전달한다.
④ Eve는 경비 보고서를 검사하고 허가한 다음 bookkeeper인 Bill에게 허가된 경비 보고서를 전달한다.
⑤ Bill은 금전 장부를 정리하고 Paymaster인 Joe에게 지불해줄 것을 요구한다.
⑥ Joe는 Peter에게 돈을 지불한다.

4.2 역할 클래스 다이어그램

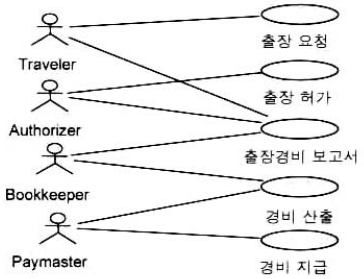
(그림 6)에서 상호작용하는 객체들은 Peter, Eve, Bill, Joe이다. 이들 객체들을 살펴보면 일정한 패턴을 유지한다고 볼 수 있다. 따라서, 이 객체 패턴을 추상화하여 Traveler, Authorizer, Bookkeeper, Paymaster 역할 클래스를 얻을 수 있다. 다음 (그림 7)은 역할 클래스 다이어그램을 보여준다.



(그림 7) 역할 클래스 다이어그램(출장 경비 예)

4.3 역할 유스 케이스 다이어그램

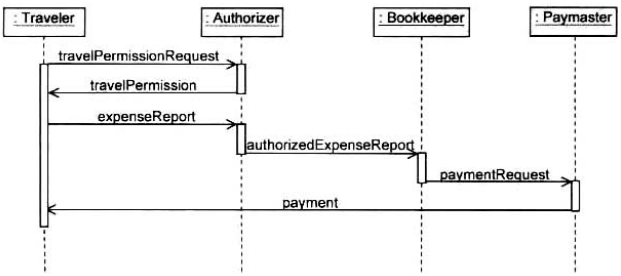
다음 (그림 8)은 역할 유스 케이스 다이어그램을 보여준다. 액터는 역할 클래스들로 구성되며 역할 클래스들은 유스 케이스들과 상호작용한다.



(그림 8) 역할 유스 케이스 다이어그램(출장 경비 예)

4.4 역할 순차 다이어그램

다음 (그림 9)는 각 역할 클래스들에 대한 메시지 순서를 보여준다. 위에서 언급한 시나리오와 (그림 8)을 바탕으로 각 역할 클래스들의 메시지 흐름을 시간 순서로 나타낸다.

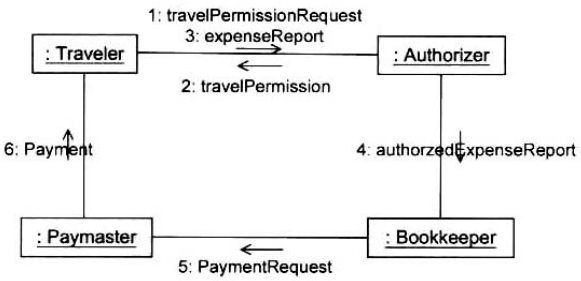


(그림 9) 역할 순차 다이어그램(출장 경비 예)

Traveler는 Authorizer에게 출장 허가를 요청하는 메시지(travelPermissionRequest)를 보내면 Authorizer는 Traveler에게 출장을 허가하는 메시지(travelPermission)를 보낸다. Traveler는 출장을 다녀와서 Authorizer에게 경비 보고서를 제출하는 메시지(expenseReport)를 보낸다. Authorizer는 경비 보고서를 읽어보고 이상이 없으면 승인하여 Bookkeeper에게 메시지(authorizedExpenseReport)를 보낸다. Bookkeeper는 경비 보고서의 경비를 산출하여 Paymaster에게 경비 지불을 요청하는 메시지(paymentRequest)를 보낸다. Paymaster는 경비 지불에 대한 메시지(payment)를 Traveler에게 보낸다.

4.5 역할 협력 다이어그램

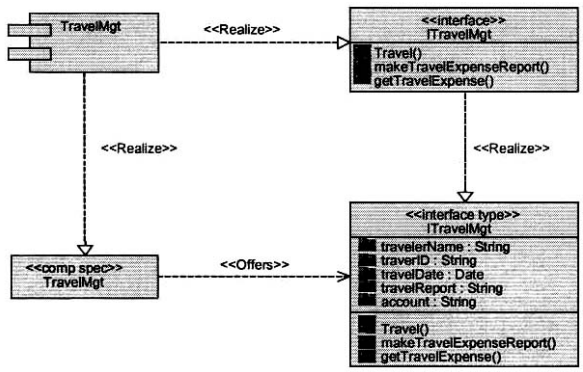
다음 (그림 10)은 각 역할 클래스들 사이 상호작용하는 관계를 나타내는 메시지 경로를 보여준다. (그림 9)의 역할 순차 다이어그램은 메시지 흐름이 시간에 따라 배열되지만 (그림 10)의 역할 협력 다이어그램은 공간에 따라 배열되어 있다.



(그림 10) 역할 협력 다이어그램(출장 경비 예)

4.6 역할 컴포넌트 다이어그램

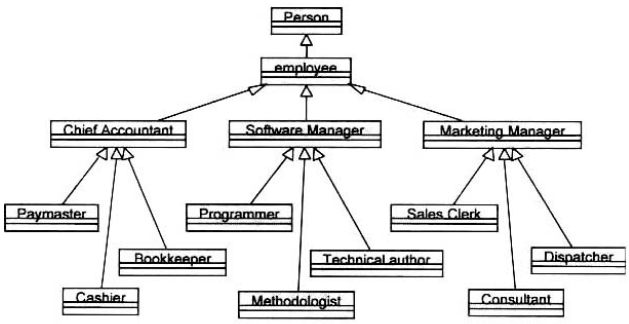
다음 (그림 11)은 역할 컴포넌트 다이어그램을 보여준다. 컴포넌트 TravelMgt는 인터페이스 ITravelMgt를 실체화한다. 또한 컴포넌트 TravelMgt와 인터페이스 ITravelMgt는 각각 <<comp spec>>과 <<interface type>>으로 명세된다.



(그림 11) 역할 컴포넌트 다이어그램(출장 경비 예)

5. 객체 모델링과의 비교

회사조직에 대한 일반적인 클래스 다이어그램은 다음 (그림 12)와 같이 나타낼 수 있다.



(그림 12) 회사 조직에 대한 클래스 다이어그램

이 그림에서 알 수 있듯이 출장 경비 처리를 위하여 관계되는 클래스들을 서로 연결시키면 다이어그램이 너무 복잡하게 된다. 또한 이런 복잡성을 줄이기 위하여 다이어그램을 상위 수준으로 추상화시키게 되면 이해하기가 어려워진다는 문제점이 있다. 그러나 위의 (그림 7)에서처럼 역할을 기반으로 한 다이어그램은 이해하기 쉬운 수준으로 표현되어진다.

또한 모델 확장의 경우(예를 들어, 비행기 티켓 구매), 일반적인 클래스 다이어그램은 더욱 복잡한 구조를 가지게 된다. 그러나 역할 클래스 다이어그램은 비행기 티켓 구매 처리에 관한 역할 모델을 생성하여 출장 경비 모델과 통합함으로서 확장시킬 수 있다.

컴포넌트 추출의 경우 하나의 역할 모델을 하나의 컴포넌트 단위로 추출할 수 있으므로 비행기 티켓 구매에 따른 역할 모델도 하나의 컴포넌트로 추출할 수 있다. 따라서 확장된 모델에 따른 컴포넌트는 출장경비에 대한 컴포넌트와 비행기 티켓 구매에 대한 컴포넌트를 조합함으로서 나타낼 수 있다.

6. 결 론

본 논문에서는 객체를 중심으로 한 객체 모델링 방법으로 컴포넌트를 추출하지 않고 객체의 역할을 중심으로 한 역할 모델링 방법을 이용하여 컴포넌트를 추출하는데 목적을 두고 있다.

물론, 현재 대부분의 컴포넌트 추출이 객체 모델링 방법을 사용하고 있고 많은 장점을 가지고 있지만 복잡한 대규모의 시스템을 모델링하기에는 적합하지 않으며 상호작용과 협력의 표현, 상속으로 인한 재사용 문제 등을 해결하는 데는 한계가 있다.

따라서 본 논문에서는 역할 모델링을 통하여 생성된 역할 모델을 UML에서 요구되는 다양한 관점의 다이어그램들로 표현하였다. 역할 모델링은 객체들의 패턴을 추상화하고 복잡한 대규모 시스템을 관계의 분리를 통하여 작은 모델들로 분할할 수 있기 때문에 시스템의 이해력을 높일 수 있으며 객체의 상호작용과 협력을 보다 잘 표현할 수 있고 재사용성을 향상시킬 수 있다. 또한 모델링 표준화 언어인 UML에서 역할 모델의 표현은 역할 개념을 보다 이해할 수 있도록 하고 다양한 관점을 제공할 수 있다. 하나의 역할 모델 단위로 하나의 컴포넌트를 추출할 수 있으므로 컴포넌트 추출이 용이하다.

향후 연구과제로는 UML에서 확장된 컴포넌트 명세 방법이 필요하며 완전한 컴포넌트를 개발하는 것이다.

참 고 문 헌

[1] Rumbaugh, J., Blaha, M., Premerlani, W., Eddy, F., Lorensen, W., "Object-Oriented Modeling and Design," Prentice-Hall, Englewood Cliffs, New Jersey, 1991.

[2] Liping Zhao, Elizabeth A. Kendall, "Role Modeling for Component Design," Proceedings of 33rd International Conference on Technology of Object-Oriented Languages, pp.312-323, 2000.

[3] Ralph Depke, Gregor Engels, Jochen Malte Küster, "On the Integration of Roles in the UML," Technical Report No. 214, University of Paderborn, August, 2000.

[4] D. Bäumer, D. Riehle, W. Siberski, M. Wulf, "The Role Object Pattern," In Proceedings of 4th Conference on Pattern Languages of Programs, 1997.

[5] T. Reenskaug, P. wold, O. A. Lehne, "Working with objects : The OOram Software Engineering Method," Manning/Prentice Hall, 1996.

[6] Bent B. Kristensen, "Object Oriented Modeling with Role," In Proceedings of the 2nd International Conference on Object-Oriented Information Systems(OOIS'95), Dublin, Ireland, London, pp.57-71, 1995.

[7] Dae-Kyoo Kim, France, R., Ghosh, S., Eunjee Song, "Using Role-Based Modeling Language(RBML) to characterize model families," Engineering of Complex Computer Systems, Proceedings. Eighth IEEE International Conference on, pp.107-116, Dec., 2002.

[8] Georg Gottlob, Michael Schrefl, Brigitte Rock, "Extending Object-Oriented Systems with Roles," ACM Transaction on Information Systems, 13(3), pp.268-296, 1996.

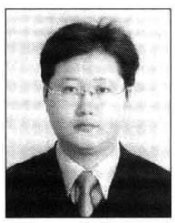
[9] E. P. Andersen, "Conceptual modeling of Object : A Role Modeling Approach," PH.D Thesis, University of Oslo, 1997.

[10] Dirk Riehle, Thomas Gross, "Role Model Based Framework Design and Integration," ACM SIGPLAN Notices, 33(10), pp.117-133, October, 1998.

[11] Grady Booch, James Rumbaugh, Ivar Jacobson, "The Unified Modeling Language User Guide," Addison-Wesley, 1999.

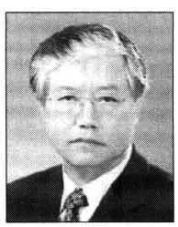
[12] The Object Management Group(OMG), "Unified Modeling Language," Version 1.3, OMG, June, 1999.

송 호 영



e-mail : humanism@zeus.kyungnam.ac.kr
1993년 경남대학교 전자계산학과(학사)
1995년 경남대학교 대학원 전자계산학과(공학석사)
1999년 경남대학교 대학원 컴퓨터공학과 박사과정 수료
2002년~현재 창원전문대학 컴퓨터정보처리과연구교수
관심분야 : 소프트웨어공학, 객체지향 방법론, 역할 모델링, 컴포넌트 등

김 정 종



e-mail : jjkim@zeus.kyungnam.ac.kr
1977년 중앙대학교 전자계산학과(학사)
1981년 중앙대학교 대학원 전자계산학과(이학석사)
1988년 중앙대학교 대학원 전자계산학과(이학박사)
1981년~현재 경남대학교 컴퓨터공학부 교수
관심분야 : 소프트웨어공학, 객체지향 방법론, 역할 모델링, 컴포넌트 등