한국산업정보학회논문지 제16권 제5호(2011. 12)

IDEF4 모델에서 UML 모델로의 변환

(Transformation from IDEF4 models to UML models)

유 문 성*
(Moon-Sung Yoo)

요 약 IDEF는 전통적인 구조적 소프트웨어 개발에 많이 사용된 방법론이다. 객체지향 소프트웨어가 널리 사용됨으로 이에 맞는 모델을 IDEF 개발 기관에서 제시한 것이 IDEF4다. UML은 객체지향 소프트웨어의 산업표준으로서 널리 이용되는 도구이다. IDEF는 CALS/EC 등에서 많이 사용하고 UML은 일반적인 객체지향 소프트웨어 개발할 때 많이 사용하기 때문에 소프트웨어 개발자들은 IDEF4모델의 표현법 보다는 UML의 표현법에 익숙하다. 또한 UML을 사용한 CASE 도구가 많이 개발되어 있어 IDEF4 모델을 UML 모델로 표현하면 이런 도구를 사용하여 소프트웨어 개발을 좀 더 효율적으로 할 수 있다. 본 논문은 IDEF4 모델을 UML 모델로의 변환하는 방법에 관하여 기술하였다. 변환 규칙을 제시하고 이를 사례에 적용하였다.

핵심주제어 : IDEF, UML, 모델 변환, 클래스 다이어그램, 상태 다이어그램. 정적모형, 동적모형

Abstract IDEF is a widely used methodology for traditional structured software development. As object-oriented softwares are widely used, an object-oriented version of IDEF, IDEF4, is developed. UML is de facto standard for object-oriented software development methods. Whereas IDEF is widely used for CALS/EC, UML is used for general object-oriented software development. Most software developers are not familiar with IDEF4 but familiar with UML. Moreover, UML has many CASE tools. So we can develop software efficiently if we convert IDEF4 model to UML model. In this paper, we transform IDEF4 models to UML models. We explain the rules and methods to convert IDEF4 model to UML model and applied the transform methods and rules to a case study.

Key Words : IDEF, UML, model transformation, class diagram, state diagram. static model, dynamic model

1. 서 론

IDEF모델은 조직과 시스템이 복잡한 연관을 가진 환경을 분석하기에 적절한 기법으로 전통적인 구조적 소프트웨어 개발 방법론을 사용하는데 주로 적용되어 왔다. IDEF모델은 기능 모델에 제한되지 않고 정보, 프로세스 등을 분석할 수 있는 모델을 제공한다[1]. IDEF모델 중에서 IDEF0, IDEF1X, IDEF3등이 구조적 개발 방법론을 사용할 경우 많이 사용되었다. 그런데 현재 소프트웨어 개발이 객체지향적 소프트웨어 개발 위주로 가는 추세이기 때문에 IDEF모델도 객체지향 개발 모델인 IDEF4를 개발하였다.

UML(Unified Modeling Language) 기법은 기존의 모델링 기법인 Booch 방법론, OMT, Jacob -son 방법론을 집대성하여 향상시킨 객체지향 분석/설계방법이다[2]. UML 기법은 객체지향 모델링이 가져야 할 필

* 상지대학교 컴퓨터정보공학부 교수

수 기능으로 일반화된 표기법을 지원하고 모델링 요소간의 관계를 명시해 주는 등의 기준을 잘 갖추고 있다. UML은 객체지향 소프트웨어 개발에서 산업표준이다.

본 논문은 IDEF4로부터 UML 다이어그램의 변환을 다루는데 크게 두 가지 이유로 이 변환이 필요하다.

첫째, IDEF는 CALS/EC등에서 많이 사용하고 UML은 일반적인 객체지향 소프트웨어 개발시 사용하는 산업표준인 모델이기 때문에 대부분의 객체지향 소프트웨어 개발자들은 IDEF4 모델의 표현법보다는 UML의 표현법에 익숙하다. IDEF4는 UML 표현과 유사한 점도 있지만 둘 사이에는 여러 가지 개념 및 표현의 차이가 있어 IDEF4로 표기한 것을 일반적인 소프트웨어 개발자가 쉽게 이해하지 못한다.

둘째, IDEF4는 객체지향에 대한 CASE 도구가 적은데 반해 UML은 CASE 도구가 많이 개발되어 있어 IDEF4로 표현한 것을 UML로 표현하면 UML용 도구를 사용하여 소프트웨어 개발을 좀 더 효율적으로 할 수 있다.

이 논문의 구성은 다음과 같다. 2장에서는 관련 연구에 대해 기술하며. 3장에서는 IDEF4로부터 UML다이어그램의 변환을 다루기 위하여 변환 규칙을 제시한다, 4장은 3장에서 제시한 규칙을 사용하여 사례연구에 적용하여 실제 변환할 수 있음을 보인다. 그리고 마지막으로 결론과 향후 연구에 대해 기술한다.

2. 관련연구

2.1 IDEF

2.1.1 IDEF 개요

IDEF(Integrated DEFinition) 방법론[3]은 1980년대 미 공군에서 ICAM(Integrated Com- puter Aided Manufacturing)프로젝트의 생산 시스템 분석 및 설계 목적으로 개발되었다. 생산 시스템을 세가지 관점 즉, 기능, 정보, 그리고 동적 관점으로 나누고 각각의 관점을 모형화 할 수 있는 방법론으로 개발하였다.

IDEF는 조직이나 시스템의 의사결정, 행동, 활동을 모델화 할 수 있는 툴로서 시스템을 분석하고 조직화하고 분석가와 고객사이의 효과적인 커뮤니케이션을 촉진시킨다.

데이터를 표준화하고, 일관성 있으며, 예측 가능한 방법으로 모델링하기 위해 사용되고 주된 목적은 데이터 자원을 완전히 이해하고 분석하기 위한 수단을 제공, 데이터의 복잡성을 표현하고 전달하기 위한 공통의 수단을 제공하여 데이터의 통합된 관점을 표현, 물리적인 데이터 베이스로 변환될 수 있는 독립적인 데이터의 관점을 정의하기 위한 수단을 제공, 기존의 데이터 자원으로부터 통합된 데이터 정의를 이끌어내기 위한 방법을 제공한다.

IDEF모델 중 IDEF0, IDEF1X, IDEF3 모델은 전통적인 소프트웨어 개발방법인 구조적 방법론을 사용할 때 많이 사용하였다.

IDEF0는 기능 모형을 위한 그래픽 언어로서, 시스템과 환경의 기능들 그리고 기능과 관련된 정보 또는 객체들을 그래픽 표현을 이용하여 구조적으로 표현한다.

IDEF1X는 데이터를 표준화하고 일관성 있으며, 예측 가능한 방법으로 모델링하기 위해 사용되는 정보 모델링 기법으로 Entity를 정의하고 그 속성을 기술하고 Entity간의 관계를 정의하여 모델을 구축한다

IDEF3은 시스템의 작동 프로세스를 모델링하는 방법론이다. 활동이나 행위를 시간적인 측면에서 순차적으로 표현한다.

2.1.2 IDEF4를 이용한 정보 모델링

IDEF4는 객체지향 소프트웨어 개발을 위하여만들어진 모델이다. IDEF4는 크게 3가지 모델 즉 정적 모델(SM), 행위모델(BM), 동적모델(DM)로 되어있다[4].

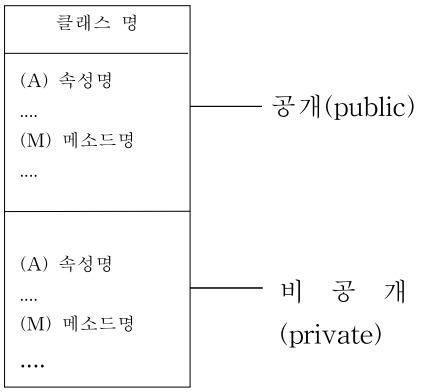
2.1.2.1 정적 모델(SM:STATIC MODEL)

정적 모델은 클래스의 정적인 구조와 관계를 표현하는데 사용되는데 구조 다이어그램(structure diagram), 상속 다이어그램 (class inheritance diagram), 관계 다이어그램(relation diagram), 연결 다이어그램(link diagram)이 포함된다.

구조 다이어그램

구조 다이어그램은 <그림 1>과 같이 나타나는데 크

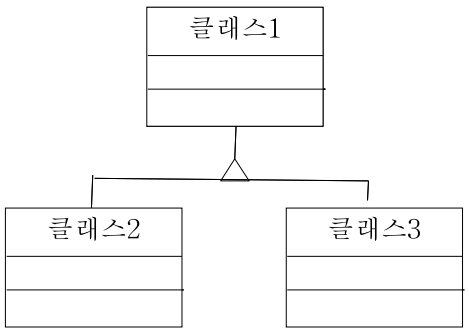
게 3가지 부분, 즉 클래스명, public부분과 private 부분으로 나눈다. 속성(A)과 메소드 (M)가 public과 private 부분에 나누어 기술되는데 public부분에 기술된 것은 외부 클래스에서도 사용할 수 있으며 private부분에 기술된 것은 그 클래스에서만 사용이 가능하다.



<그림 1> 구조도의 구성요소

상속 다이어그램

상속 다이어그램은 <그림 2>과 같이 표현하는데 <그림 2>는 클래스1에서 클래스2와 클래스3으로 상속이 됨을 나타낸다. 상속이 되는 것은 속성과 메소드다.



<그림 2> 상속 다이어그램

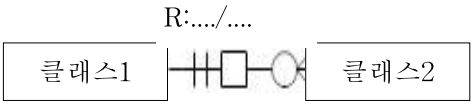
관계 및 링크 다이어그램

관계 다이어그램은 <그림 3>처럼 표현하는데 관계가 있는 클래스들을 선으로 연결하고 관계를 선위에 표시하는데 /를 경계로 앞부분은 왼쪽 클래스가 오른쪽 클래스에 대한 관계를 나타내며 뒷부분은 오른쪽 클래스가 왼쪽 클래스에 대한 관계를 나타낸다.

다중성은 클래스에 가까운 순으로 최대값과 최소값 순으로 표현하며 다중성은 1:1, 1:다, 다:다 세가지 경우가 있다. 원과 선은 0과 1을 표현하며 까치발(fork)은 최소값 이상을 표시한다.

다중성과 관련하여 왼쪽에 나타낸 것은 오른쪽 하나의 객체에 하나와 대응되는 값이며, 오른쪽에 그 반대다.

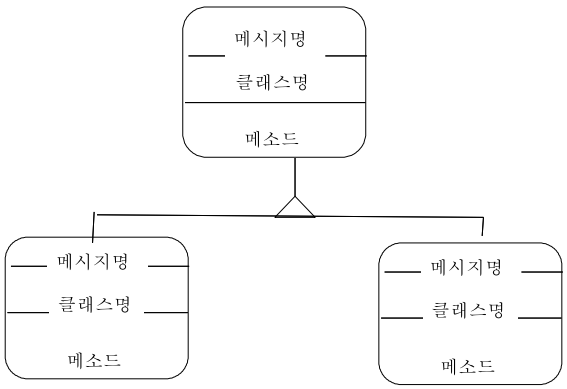
링크 다이어그램은 관계 다이어그램과 표현 방법이 같으며 다만 구현 단계에서의 클래스들의 관계를 다룬다.



<그림 3> 관계 다이어그램

2.1.2.2 행위 모델(Behavior Model)

행위 모델은 객체의 메소드들에 의한 메시지 구현을 사양화하며 행위 다이어그램으로 나타낸다. BM에서 사용하는 메소드는 <그림4>의 표현을 가진다. 행위 모형은 같은 메시지에 대하여 반응하는 여러 클래스들의 메소드의 다형성 (polymorphism)을 보여 주는데 사용한다. 구조 다이어그램의 상속 다이어그램의 일반화/특수화 방식으로 메소드들 간의 관계를 표시한다. 메시지들을 구현한 메소드간의 관계를 보여주며 외부 행위와 내부 구현의 관계를 사양화한다.



<그림 4> 행위 다이어그램

2.1.2.3 동적 모델(DM:Dynamic MODEL)

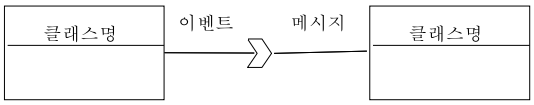
객체간의 통신과 상태간의 전이를 표현하는데 사용되는데 이 모델은 다음과 같은 사항을 그래픽으로 표현한다.

(1) 객체간에 전달되는 메시지

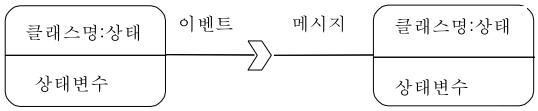
(2) 메시지를 실행시키는 원인이 되는 이벤트,

(3) 객체가 메시지 실행시 한 상태에서 다른 상태로 객체의 전이

동적 모델은 클라이언트/서버 다이어그램 (client /server diagrams)과 상태 다이어그램 (state diagrams)으로 표현되는데 클라이언트/서버 다이어그램은 상태의 변화를 클래스 단위에서 나타냈으며 상태 다이어그램은 클래스의 각 상태별 단위에서 나타내었다.



<그림 5> 클라이언트/서버 다이어그램



<그림 6> 상태 다이어그램

2.2 UML

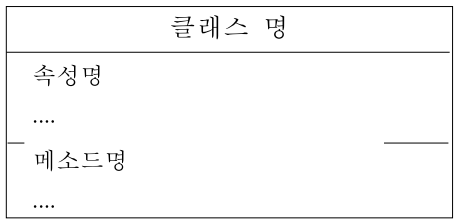
UML(Unified Modeling Language)[5][6]은 사실상 소프트웨어 산업 표준으로 시각화, 사양화, 구축, 시스템의 요소를 문서화하는 모델링 언어이다. UML은 문법과 의미가 잘 정의되어 있으며 요구 사항 수집에서부터 소프트웨어 구성 요소의 배포까지 객체 지향 소프트웨어 시스템을 개발하는데 하향식 정련기법을 사용하며 이를 위하여 많은 그래픽 도구들을 제공한다.

UML 다이어그램은 그림에 의해 표현되는 클래스, 객체, 노드 및 관계와 같은 모델 요소를 포함한다. UML 다이어그램에는 사용사례(Use Case) 다이어그램, 클래스(Class) 다이어그램, 상태(State) 다이어그램, 순차(Sequence) 다이어그램, 통신(Communication) 다이어그램, 컴포넌트(Component) 다이어그램, 배포(Deployment) 다이어그램 등이 있다.

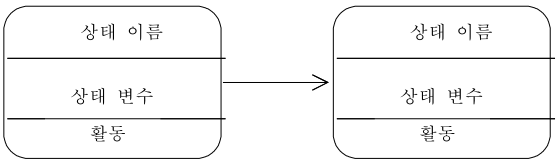
본 논문에서 IDEF4로 표현된 다이어그램을 UML 다이어그램을 변환할 때 클래스 다이어그램과 상태 다이어그램이 사용된다. 클래스 다이어그램은 클래스의 내부 구조와 클래스와 클래스 사이의 정적인 관계를 나타낸 다이어그램으로 클래스는 클래스 이름, 속성과 연산들로 구성되어 있으며 관계에는 연관(association), 다중성(multiplicity), 상속(inheritance), 의존(depen -dency)등이 있다.

상태(State) 다이어그램은 객체가 상태의 변화를 사건과 동작(action)에 대해 일으키게 되는 것을 그림으로 보여주며, 각 생태에 상태명, 상태변수, 활동을 기술한다.

2.3 IDEF에서 다른 모델로의 변환에 대한 기존 연구



<그림 7> UML 클래스 구성 요소



<그림 8> UML 상태 다이어그램

IDEF는 각 모델마다 특징이 있어 이들을 결합하여 새로운 모델로 만들거나 기존의 다른 모델로 변환하려는 연구가 있어왔다.

먼저 IDEF 모델에서 시뮬레이션 코드로 변환하는 연구가 있는데[7][8] 두 모델의 구조적인 차이로 효율적이지 못하였다.

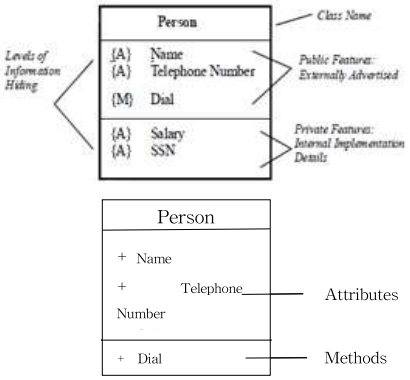
IDEF 모델에서 Petri Net로 변환한 연구[9][10][11]가 있는데 특히 김중인 등의 연구[11]에서는 IDEF0에서 IDEF3로, IDEF3에서 Petri Net으로의 변환을 연구하였다.

IDEF0, IDEF1X, IDEF3 모델을 이용하여 UML다이어그램을 도출한 연구[12]는 IDEF0, IDEF1X, IDEF3 모델이 객체지향적 모델 기반이 아닌 구조적 모델 기반이기 때문에 UML 다이어그램 도출하는데 근본적인 한계가 있다.

기존 연구들은 구조적 모델을 사용하여 다른 모델로 변환하는 방법에 관한 것인데 IDEF에서 제안한 객체지향 모델인 IDEF4 모델에서 UML 모델로의 변환은 기존 연구가 없어 본 연구에서는 이 변환에 대해 연구하였다.

3. IDEF4모델에서 UML모델 변환 방법

IDEF4는 객체지향 데이터 모델인데 다음의 규칙을 사용하여 UML의 클래스 다이어그램과 상태 다이어그램으로 변환 할 수 있다. 여기서 사용한 IDEF4의 예는 IDEF4 Method Report [4]에 있는 예제를 사용하였다.



<그림 9> 구조 다이어그램의 변환

3.1 클래스 다이어그램으로의 변환

규칙1: 구조 다이어그램에서 public과 private로 구분한 것을 속성과 메소드로 구분한다.
public과 private는 +와 -로 변환한다.

규칙2: 상속 다이어그램의 상속관계는 그대로 유지하되 상속을 표시하는 표기법만 바꾼다.

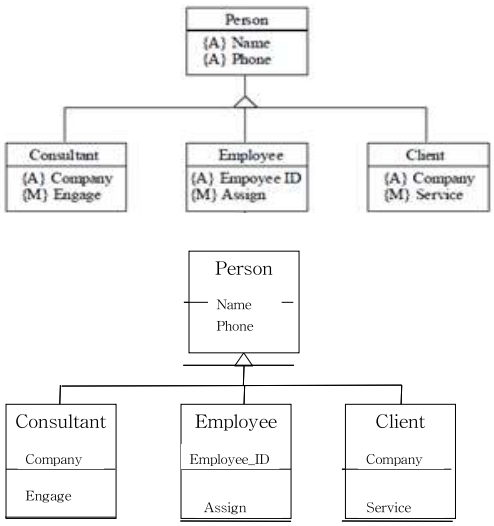
규칙3: 관계(relation) 다이어그램과 연결(link) 다이어그램에서 표현한 것을 UML의 클래스 다이어그램의 관계 표현으로 다음과 같이 변환한다.

① IDEF4의 관계(연관)는 R:.../...형식으로 표시되는데 그 관계 둘 중 하나를 선택하고
방향성을 나타내기 위하여 화살표(▶)를 사용한다

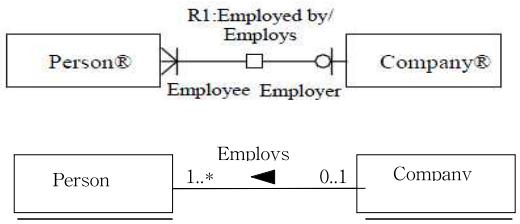
②. 선택한 관계의 다중성을 UML 표기법으로 다음과 같이 바꾼다. 즉 IDEF4의 원과 선은 0과 1로 변환하며 까치발(fork)은 *로 바꾼다.

③ 최소값과 최대값은 인접하여 표현한 것에서 하나의 기호나 두 기호를 두 점(..)을 사용하여 연결한 표현으로 바꾼다.

규칙4: BM의 행위 다이어그램에 나타난 다형성은 UML 클래스 다이어그램에서 종속관계인 클래스들이 같은 메소드 명을 가진 것이 있는지 확인한다.



<그림 10> 상속 다이어그램의 변환



<그림 11> 관계 다이어그램의 변환

3.2 상태 다이어그램으로의 변환

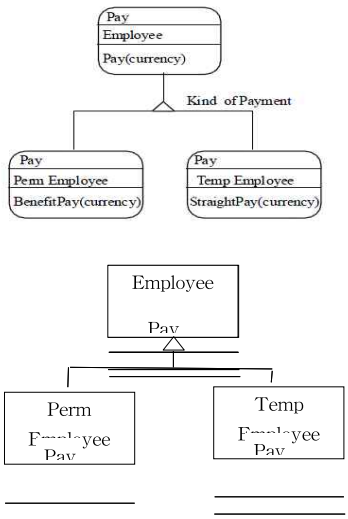
규칙5: Client/Server 다이어그램은 그 내용이 IDEF4의 상태 다이어그램에 반영되었기 때문에UML

의 상태 다이어그램 작성 시 사용하지 않는다.

규칙6: 상태 다이어그램은 UML의 상태 다이어그램으로 다음과 같이 바꾼다.

① IDEF4에서의 클래스와 상태 표기를 UML 상태 다이어그램에서는 상태만으로 바꾸고 클래스를 나타내고 싶을 때는 클래스를 따로 병기한다. 전이표시를 ─〉─ 에서 → 로 바꾼다.

② IDEF4에서 전이의 이벤트와 메시지는 E: 와 M: 을 사용하여 표현하는데 이를 UML에서 사용하는 표현인 /를 중심으로 앞에는 이벤트 뒤에는 메시지(액션) 표현으로 바꾼다.



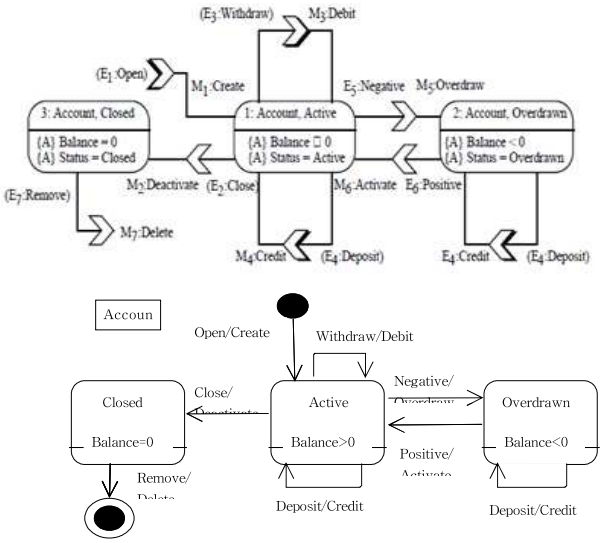
<그림 12> 행위 다이어그램의 변환 확인

③ UML에서는 시작 상태(●)와 종료상태(◉)를 추가한다.

④ IDEF4에서 전이의 이벤트(E)와 메시지(M)는 순서 번호가 있는데 UML에서는 시작 상태로부터 순서를 추정할 수 있으므로 순서 번호를 생략한다.

4. 사례연구

사례 연구로 비디오 대여점 업무를 선택하였다. 비디오 대여점에서는 고객이 비디오를 빌리고 반납하는 기능이 주 기능이다. 고객(custo mer)은 일정한 회비를 낸 member와 그렇지 않은 guest로 구별하였다. 비디오 대여점의 업무는 고객이 와서 비디오를 선택해서 점원에게 대여를 신청하면 대여업무 로그인 업무에서 고객 의 아이디와 비디오의 아이디가 포함한 바코드를 스캐너로 읽고 고객이 비디오를 연체한 것이 있는지를 점검한 다음 없으면 대여를 한다. 이후 고객이 마감일이 되어 비디오를 반납하면 .반납일이 지난지의 여부를 확인하고 비디오를 돌려받는다. 비디오가 이상이 없는지를 점검한 후 진열대에 꽂는다.



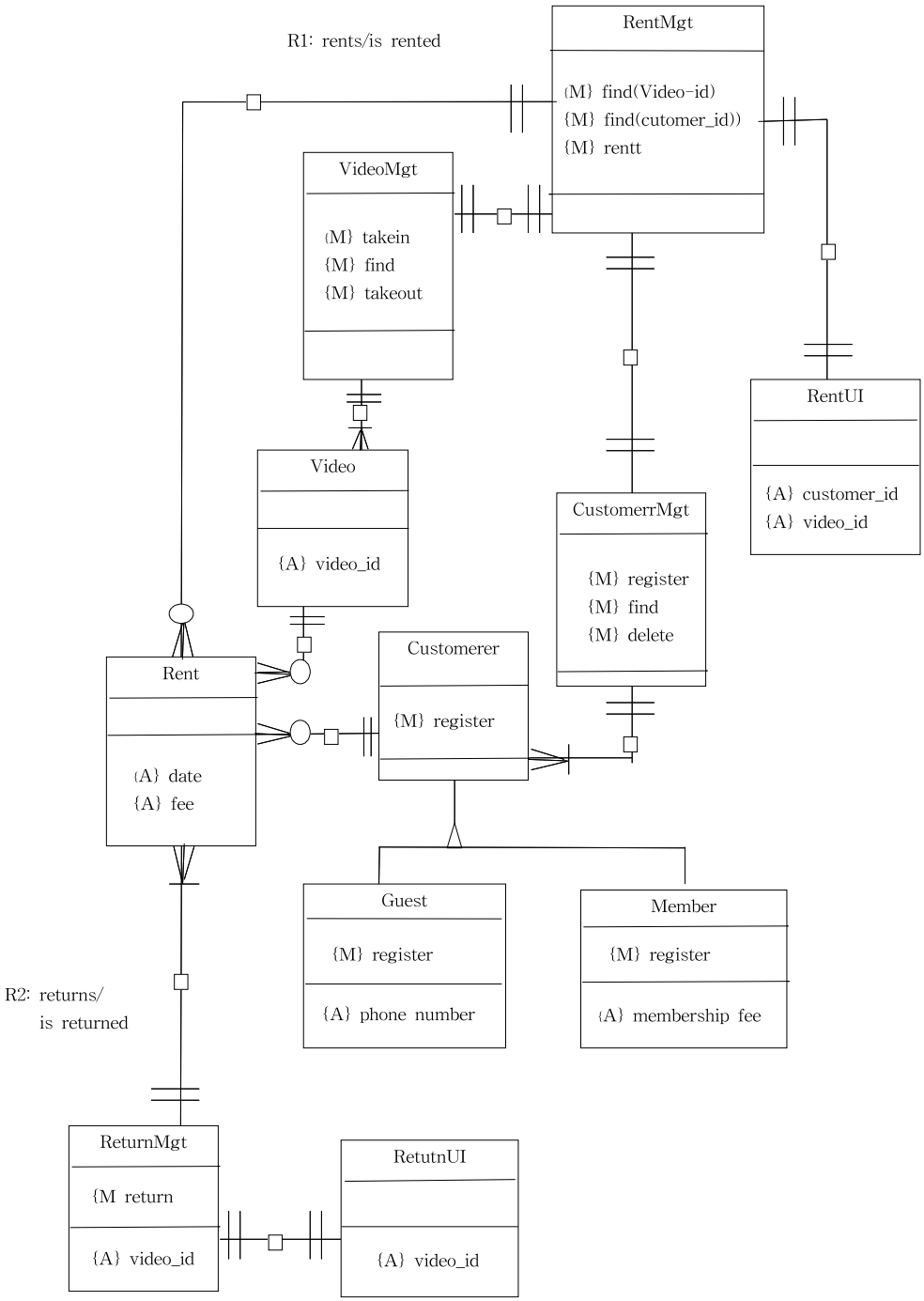
<그림 13> 상태 다이어그램의 변환

4.1 IDEF4를 이용한 모델링

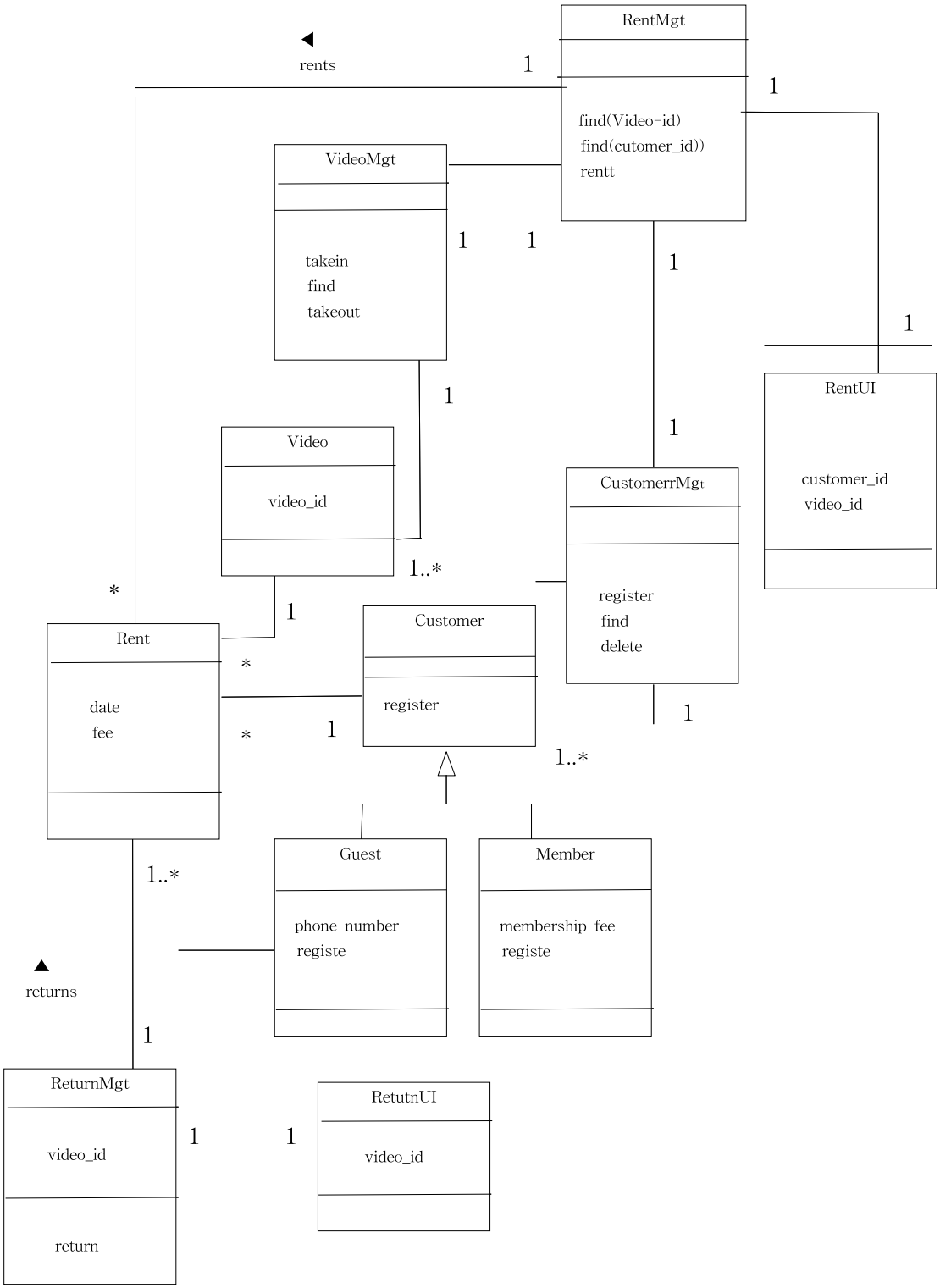
비디오 대여 업무를 IDEF4 모델을 사용하여 나타내었는데 <그림 14>는 SM(정적 모델)을, <그림 16>은 BM(행위 모델)을, <그림 17>은 DM(동적 모델)을 나타내었다

SM(정적 모델)은 구조 다이어그램(structure diagram), 상속 다이어그램 (class inheritance diagram), 관계 다이어그램(relation diagram), 을 모두 한 다이어그램에 그렸다. BM(행위 모델)은 고객(Customer) 등록시 두 종류의 고객 즉 일정한 액수의 금액을 미리 낸 멤버(Member) 고객과 그렇지 않은 Guest 고객이 등록하는 과정이 다른 것을 나타내었다

DM은 상태 다이어그램으로서 비디오의 상태의 변환을 나타낸 것이다. 비디오 상태를 reserved, checked_out, rented, returned의 네 개의 상태로 나타내었다.



<그림 14> 비디오 대여점의 IDEF4 SM

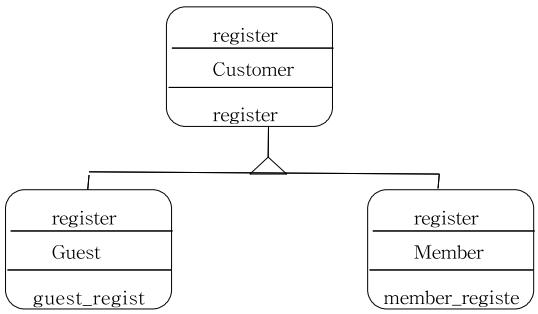


<그림 15> 비디오 대여점의 UML 클래스 다이어그램

4.2 UML을 이용한 모델링

4.2.1 클래스 다이어그램으로 변환

클래스 다이어그램은 SM(정적 모델)의 구조 다이어그램 (structure diagram), 상속 다이어그램 (class inheritance diagram), 관계 다이어그램(relation diagram) 각각에 대해 3장에서 기술한 규칙을 적용하여 변환하였다. <그림 16>의 BM에 나타난 다형성은 SM을 이용하여 변환한 해당 클래스 다이어그램이 그 메소드를 가지고 있는가를 점검하는데 본 사례에서는 그 메소드들이 존재함을 확인할 수 있다. 변환한 클래스 다이어그램이 <그림 15>에 도시되어 있다



<그림 16> 비디오 대여점의 IDEF4의 BM

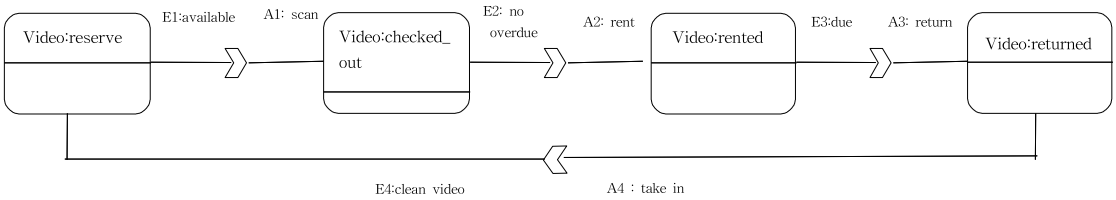
4.2.2 상태 다이어그램으로의 변환

상태 다이어그램은 DM(동적 모델)의 상태 다이어그램에 3장에서 기술한 규칙을 적용하여 변환하였다. 변환한 상태 다이어그램이<그림 18> 에 도시되어 있다

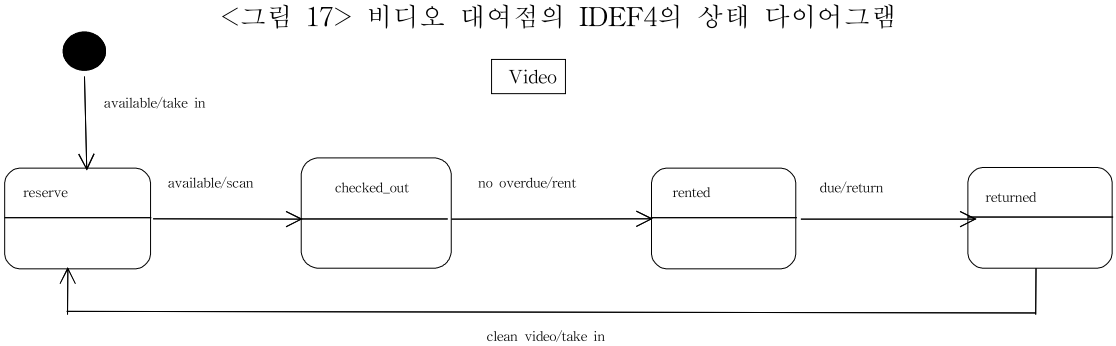
5. 결 론

본 논문에서는 객체지향 시스템을 나타낸IDEF4에서 UML 모델로의 변환 방법에 대해 그 규칙과 방법을 제시하였고 그 방법을 비디오대여점 사례 연구를 통하여 확인하였다. IDEF4 로 표현한 모델은 UML의 클래스 다이어그램과 상태 다이어그램으로 표현할 수 있었다. 이렇게 UML로 표현하여 보다 많은 소프트웨어 개발자들이 모델로 표현한 것을 쉽게 이해하게 되었으며 UML을 사용한 CASE 도구를 사용하여 소프트웨어 개발을 좀 더 효율적으로 할 수 있게 되었다.

이 변환을 하면서 IDEF4와 UML의 객체지향 표기법에 대한 차이를 발견할 수 있었다. 또한 UML은 클래스 다이어그램과 상태 다이어그램 이외에도 유스 케이스 다이어그램, 통신 다이어그램, 순차 다이어그램 등의 많은 표현이 있으나 IDEF4로는 이 다이어그램들을 표현할 수 없었다



<그림 17> 비디오 대여점의 IDEF4의 상태 다이어그램



<그림 18> 비디오 대여점의 UML의 상태 다이어그램

IDEF4 모델에서 UML모델 변환을 자동으로 하는 CASE 도구를 구현해야 효율적이다. 이것에 대한 향후 연구가 필요하다.

참 고 문 헌

[1] J. Sarkis and L. Lin, "An IDEF0 func- tional planning model for the strategic implementation of CIM systems," Inter- national Journal of Computer Integrated Manufacturing, Vol.7, No.2, pp.100-115, 1994.

[2] G. Booch, J. Rumbaugh, and I. Jacobson, Unified Modeling Language User Guide, Addison-Wesley Professional, 2005.

[3] http://www.idef.com/

[4] R. Mayer, A. Keen, D. Browne, S. Harrington, C. Marshall, M. Painter, F. Schafrik, J. Huang, M. Wells, and H. Hisesh, "IDEF4 object-oriented design method report," Wright-Patterson AFB, Ohio, 1995.

[5] OMG, "Formal Specifications: Unified Modeling Language," http://www.omg.org/ spec/UML/.

[6] M. Fowler, K. Scott, UML Distilled (3rd ed.), Addison-Wesley, Reading, MA, 2003.

[7] R. Gay, R. Lim, and L. Shing, "Using IDEF methodology for functional, infor- mation and dynamic modelling of a FMS," pp.97-100, 1991.

[8] E. Y. ROBERT and J. Vesterager, "An approach to CIM system development whereby manufacturing people can design and build their own CIM systems," International Journal of Computer Integ- rated Manufacturing, Vol.4, No.5, pp.288- 299, 1991.

[9] V. Pinci and R. Shapiro, "An integrated software development methodology based on hierarchical colored Petri Nets," Advances in Petri Nets 1991, pp.227-252, 1991.

[10] J. Zaytoon, E. Niel, and A. Mille, "A functional-to-behavioural transformation framework for specification and design of manufacturing systems," Computer Integ- rated Manufacturing Systems, Vol.7, No.4, pp.247-258, 1994.

[11] 김중인, "비즈니스 프로세스 모델링 연계 방법론 -IDEF0, IDEF3, Petri Net," 한국 CALS/EC 학회지, 제3권, 제.2호, pp.141-160, 1998.

[12] 신기태, 박찬권, 심억수, 김은갑, "UMM 적용을 위한 IDEF 매핑 방법에 대한 연구," 경영과학, 제21권, 제.2호, pp.61-77, 2004.

유 문 성 (Moon-Sung Yoo)

- 정회원
- 서울대학교 수학과 이학사
- Louisiana State University, Computer Science, Ph.D.
- 상지대학교 컴퓨터정보공학부 교수
- 관심분야 : 소프트웨어공학, 웹 소프트웨어 개발, 모바일 소프트웨어