韓國經營科學會誌
第21卷 第3號
1996年 12月


# 구조적 분석 산출물을 이용한 객체 모델 유도 방법론

이희석* · 배한욱** · 유천수***

# A Methodology for Deriving an Object Model by Using Structured Analysis Results

Heeseok Lee* · Hanwook Bae** · Cheonsoo Yoo***

Abstract

In conventional analysis methods, data and process are loosely coupled for building information systems. Several object oriented approaches have been proposed to integrate data and process. However, object oriented analysis requires a radical paradigm and thus system analysts find difficulties in generating object models directly from end users. To alleviate these difficulties, this paper proposes a methodology for deriving an object model by using structured analysis results. Objects are obtained primarily from entities in Entity-Relationship Diagram. Methods are obtained through the analysis of the relationship between processes and data stores in Data Flow Diagram. Methods are assigned to the objects by using object/process matrices. A real-life case is illustrated to demonstrate the usefulness of the methodology.

## 1. 서 론

인간이 다수의 대상에서 동일성을 파악하고 모순되거나 애매한 것을 배제하면서 진위를 판단하며 사고해 나가듯이 하나의 정보 시스템을 표현한 모델도 인간의 사고의 표현이므로 이러한 논리성이 요구된다. 이러한 모델은 구현하고자 하는 현실 문제를 표기법과 도식을 이용해 일관성 있는 규칙에 따라 가시화한 추상적

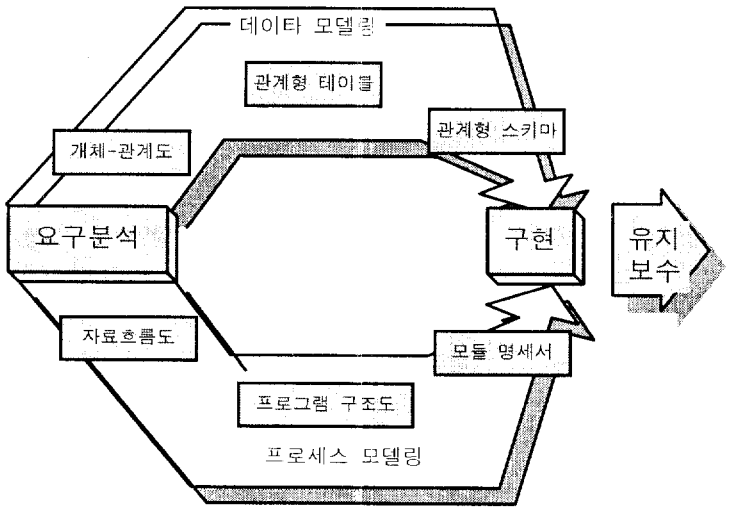
* 한국과학기술원 테크노경영대학원 부교수
** 엔더슨컨설팅(주)
*** 한국과학기술원 테크노경영대학원 경영정보전공

인 것이다.

구조적 분석 방법 (Structured Analysis Method)에서는 시스템의 내부 및 외부에서 상태변화를 발생시키는 사건을 파악하고 사건에 대한 시스템의 반응을 식별하여 시스템의 핵심이 되는 논리적인 기능 및 프로세스들을 추출한다[37]. 이러한 논리적인 프로세스는 시스템에서 데이타와 상호 작용을 하는 기능으로서 입력 자료 흐름 (Data Flow)의 변환 또는 다른 프로세스들에게 전송하는 역할을 수행한다. 또한 프로세스는 처리된 자료를 내부 자료 저장소에 저장하고 작업 시 이를 참조하기도 한다. 개별 프로세스는 데이타와는 별개로 해야 할 일의 순서에 따라 동작하여 프로세스에 주어진 임무에 필요한 데이타를 처리하게 된다. 여기서의 초점은 개별 프로세스가 어떠한 순서로 주어진 임무를 수행하는 가에 있으며, 중요한 관심 사항은 데이타를 처리하는 프로세스가 된다.

구조적 분석 방법에 의한 정보 시스템 구축 시 사용되는 모델은 주로 시스템 기능을 표현하는 프로세스 모델과 그에 필요한 데이타의 논리적 표현인 데이타 모델로 대별된다. 이 두 모델을 표현하기 위한 도구로써 현재 가장 많이 사용되고 있는 것이 자료 흐름도 (Data Flow Diagram, DFD)와 개체-관계도 (Entity-Relationship Diagram)이다. 결과적으로 하나의 시스템이 시스템의 동적인 면과 정적인 측면이 고려되어 각각 프로세스 모델과 데이타 모델로 형상화되며 [그림 1-1]은 이러한 관계를 도시한 것이다.

하나의 시스템에 대해 이러한 분리된 접근은 데이타와 프로세스를 별개로 고려해야 하므로 시스템 통합성의 저해 요소이다. 즉, 데이타와 프로세스로 분리하여 구현하기 때문에 개별 프로세스나 데이타에 영향을 미치는 업무의 작은 변화에도 전체 시스템의 변경이 요구된다. 예를 들면, 단위 데이타에 대한 구조 변화는 이 데이타를 활용하는 모든 프로세스에 영향을 미치며 반대로 한 단위 프로세스의 변화도 이 프로세스와 연계되는 모든 데이타 구조의 변화를 필요로 한다. 그러므로 상호 영향을 주고 받는 프로세스와 데이타의 변화를 상호 조화시키는 것이 중요한 과제이다.




[그림 1-1] 구조적 분석 방법의 시스템 구현 양 측면

이러한 프로세스와 데이타의 결합을 위한 체계적인 대안으로 객체 지향 (Object Oriented) 개념이 제시되었다고 할 수 있다. 1960년대 후반에 시뮬레이션을 위한 프로그래밍 언어로서 등장한 객체 지향 개념은 70년대와 80년대를 거치며 시스템 분석 및 설계 방법으로도 발전해 왔다. 객체 지향 프로그래밍 언어에 사용되었던 개념들이 시스템 개발을 위한 일반적인 원칙으로 적용되기에 이른 것이다[25]. 객체 지향 모델링에 있어서는 실제 업무의 실체, 사건 또는 개념들이 각각 구현될 시스템의 객체(Object)로 표현되어 실제 업무의 실체, 사건 또는 개념들과 관련된 정보 및 행위(Behavior)가 해당되는 시스템의 객체 속에 그대로 반영된다[32].

이러한 객체 지향 개념은 응용 프로그램 개발뿐 아니라 시스템의 분석 및 개발 도구로서도 다수의 장점을 가지므로 90년대에 들어서 정보 산업 전반에 걸쳐 광범위 하게 쓰이게 되었고 상업적으로 큰 성공을 거두고 있다[23]. 그러나 이러한 객체 지향 개념의 확산에도 불구하고 기존 개발자 및 개발 조직은 이러한 변화된 기술 환경을 받아들이기에 어려움을 겪고 있다. 그 주된 이유는 객체 지향 개념은 완전히 다른 사고 방식을 요구하므로 개념 전환이 어려울 뿐만 아니라 학습을 통해 익숙해져서 객체 지향 개념이 지니는 장점을 충분히 활용하게 되는 데에도 상당한 시간을 요구하기 때문이다. 또한 객체 지향 모형화를 위한 정형화된 요구 분석 방법의 미비에도 그 요인이 있다[22][20].

본 논문에서는 기존의 개체-관계 모델과 자료 흐름도를 기초로 객체 모델 (Object Model)의 풍부한 표현력을 충분히 반영하는 데이타와, 이 데이타와 밀접하게 영향을 주고받는 프로세스를 통합한 객체 모델을 효과적으로 유도하기 위한 방법론을 구현하고자 한다. 본 방법의 개략적 구도는 저자들에 의해 [3]에서 처음으로 제기되었다. 본 방법론에 의해 기존의 구조적 방법론에 익숙한 개발자나 이들 산출물을 객체 지향 기술에 적용하여 재구축하려는 설계자가 객체 모델을 보다 용이하게 도출할 수 있다 하겠다.

본 논문은 다음과 같이 구성되어 있다. 2장에서는 관련된 기존 연구에 대하여 문헌 연구가 이루어 졌다. 3장에서는 기존의 개체-관계 모델과 자료 흐름도에서 데이타와 프로세스를 통합한 객체 모델을 유도하기 위한 방법론이 제안되었고, 4장에서는 이 방법론을 실제 문제에 적용한 사례연구를 다루었다. 5장에서는 본 논문의 요약과 공헌, 앞으로의 연구과제에 대하여 서술했다.

## 2. 관련 연구

시스템 분석을 위한 데이타와 프로세스의 통합은 크게 두 방향으로 연구가 진행되어 왔다. 시스템의 두 가지 측면의 모델을 결합하려는 시도와 새로운 개념인 객체 지향 모델을 활용하는 방안이 그 두 방향이며 이들에 대해 살펴보면 아래에 기술한 바와 같다. 이하 본 논문에서는 객체 지향 모델을 구성하는 여러 가지 모델 중 객체 지향 데이타베이스의 설계를 위한 모델을 객체 모델이라고 칭한다.

### 2.1 데이타 모델과 프로세스 모델의 결합

Markowitz[24]는 자료 흐름도의 프로세스가

개체-관계 모델의 개체에 영향을 미치는 행위라는 것에 바탕을 두고 자료 흐름도의 프로세스를 개체-관계 모델로 표현하였다. 즉, 프로세스는 입력된 데이타를 변환하여 변환된 결과를 출력하는 것이라는 개념에 근거하여 프로세스를 하나의 관계 (Relationship)로 하여 개체-관계 모델을 표현하였다. 그러나 개체-관계 모델로 표현된 개념 스키마가 데이타와 프로세스를 통합하여 전체적인 관점을 제공할 수 있다는 면에서 의의가 있을 수 있으나, 실제 시스템으로의 전환 과정에 대한 연구가 미흡하여 실제적 시스템 분석 시 사용하기가 용이하지 않다.

Batini et al.[7]는 개체-관계 모델과 자료 흐름도를 이용하여 데이타의 개념적 설계와 프로세스의 설계를 동시에 진행해 나갈 것을 제안했다. 자료 흐름도에서 특정 데이타 저장소나 한 프로세스에 의해서 사용되는 데이타 구조의 일부를 외부 스키마로 설정하고 이것을 자료 흐름도와 결합하여 개체-관계 모델과 자료 흐름도를 정제해 나가는 것이다. 이처럼 서로 다른 두 가지 모델을 상호 보완해 나가는 것을 상호 완결 (Mutual Completeness)이라 하였다. 즉, 자료 흐름도에서 나타나는 모든 데이타는 반드시 개체-관계 모델에서 다루어져야 하며 또 역으로 개체-관계 모델의 데이타를 다루는데 필요한 모든 행위 (Behavior)는 자료 흐름도에 존재해야만 한다는 가정을 바탕으로 개체-관계 모델과 자료 흐름도를 계속 상호 보완해 가면서 완성한다.

### 2.2 객체 모델로의 변환

Choobineh et al. [13]는 개체-관계 모델에서 개별 개체들의 행위 또는 연산 (Operation)에 관한 면이 간과되고 있음을 직시하고 객체 지향 개체-관계 모델 (Object-Oriented Entity-Relationship Model)을 제안하였다. 이 객체 지향 개체-관계 모델은 개체들간의 동적인 면을 표현할 수 있고 객체 지향 개념을 도입하여 어떤 개체가 다른 개체의 속성이나 행위에 접근하기 위해서는 메세지 전달을 통해서만 가능하도록 하여 객체 지향 개념의 정보 은닉 (Information Hiding)을 구현하려고 하였다. 또한 개별 개체의 행위를 확장된 개체-관계 모델의 속성으로 두어 개체들간의 계승을 통하여 상위 개체에서 설정된 행위가 하위의 개체들에서도 사용될 수 있도록 하였으며 객체 지향 개체-관계 모델을 표현할 수 있는 다이어그램도 제시했다. 그러나 시스템이 수행하여야 할 기능들을 객체의 단위 프로세스에 소속시킴으로써 상위의 프로세스를 하위로 세분화하여 나갈 수가 없어 전체적인 관점의 조망이 어렵다는 단점이 있다.

Alabiso et al.[5]는 구조적 분석 결과에 새로운 요소를 추가하여 객체 지향 설계로의 변환을 시도하였다. 즉, 자료 사전에서 객체를 정의하고 자료 흐름도에서 객체의 메쏘드 (Method)에 관한 정보를 추출하여 객체 모델을 구현하였다. 그러나 자료 흐름도의 프로세스를 이용한 객체의 메쏘드 정의 시 메세지의 대상 즉, 메쏘드가 속한 객체를 프로세스의 입출력 데이타 중에서만 고려함으로써 하나의 프로세스가 다수의 객체의 메쏘드에 관여할 수 있는 경우를 간과하였다.

Bailin[6]은 요구 분석의 한 방법으로서 구조적 설계로부터 객체 지향 설계로 전이하기 위한 중간 단계를 제시하였다. Bailin은 각 외부 개체 (External Entity 또는 Terminator)가 객체나 프로세스가 되는 개체 자료 흐름도 (Entity Data Flow Diagram, EDFD)를 기반

으로 하여 이 EDFD로부터 각 프로세스를 하위 단계로 전개해 나가는 과정에서 구조적 분석 방법과 객체 지향 분석 방법을 비교 분석하였다. 구조적 방법에서는 프로세스들이 모여 상위의 프로세스를 이루어 하나의 기능을 수행할 경우 그 상위의 프로세스를 구성하는 하위의 프로세스들을 하나의 모임으로 하나, 객체 지향 분석 방법에서는 같은 데이타에 작용하는 프로세스들을 하나의 모임으로 하고 있으며 이러한 차이가 구조적 분석으로부터 객체 지향으로의 전환을 어렵게 만드는 주요 요인임을 지적했다.

Wieringa[35]는 구조적 분석 방법과 객체 지향 분석 방법이 근본적으로 서로 다른 개념임을 지적하고 기존의 두 가지 방법을 통합하려는 시도 자체를 비판하였다. Wieringa는 구조적 분석 과정을 거쳐 나온 모델을 객체 지향 개념으로 산출된 모델로 바꿔봄으로써 두 가지 방법을 비교하였다. 그는 기존의 구조적 분석 방법이 데이타의 변환에 초점을 두고 데이타의 알고리듬을 외부로부터 차단한 반면, 객체 지향 분석 방법은 어떠한 시점에서의 상태와 행위를 외부로부터 은닉함에 주목하였다. 또한 기존의 자료 흐름도에는 시스템의 기능 명세뿐만 아니라 실 세계에 대응하는 모델도 포함되어 있어 완전한 모델의 성립 전에 정의된 시스템의 기능이 모호할 수 있으나 객체 지향 분석 방법에서는 이를 분리하여 먼저 모델을 구축하고 난 다음 이에 기반을 두고 시스템의 기능을 명세화함을 지적하였다. 또한 구조적 분석 결과 중의 하나인 자료 흐름도로부터 객체 지향 모델로 변환하는 과정에서 객체가 될 수 있는 대상인 외부 개체와 데이타 저장소 중에서 객체로 되는 경우의 수를 줄일 수 있는 몇 가지 방법을 제시하였다.

Fong[17]은 확장된 개체-관계 모델 (EER, Extended Entity Relationship Model)로부터 객체 지향 모델 중 Rumbaugh et al. [28]의 표기법인 OMT (Object Modeling Technique)으로의 변환을 제시하였다. Fong은 기존의 데이타베이스를 객체 지향 데이타베이스로 변환하기 위한 한 과정으로 이러한 방법론을 제시하였고 이를 위해 OMT로 치환 (Mapping)하는 규칙들을 제시하였다. 즉, 개체는 OMT의 클래스로, 관계는 OMT의 제휴 (Association)로 바꾸었다. 또한 EER에서 표현되고 있는 개체에 관한 제약 조건들은 해당 클래스의 메쏘드로 표현하였다. 이러한 변환은 객체 지향 개념에 익숙치 않은 개발자들에게 기존의 개체-관계 모델에서 OMT로 변환하는 과정을 보여 준다는 면에서 의의가 있을 수 있으나, 개별 객체의 데이타 행위에 대한 고려가 미흡하다.

한편, 기존의 관계형 데이타베이스를 확장하여 객체 지향 데이타베이스로 변환하는 연구도 진행되고 있다 (예를 들면, [4]). 관계 테이블은 제반 단계를 거쳐서 객체 클래스로 변환되게 된다. 변환 과정에서 객체 지향 기법에 필요한 여러 관계성이 추가될 수 있다. 그러나, Fong과 마찬가지로 객체의 메쏘드에 관해서는 처리가 미비하다.

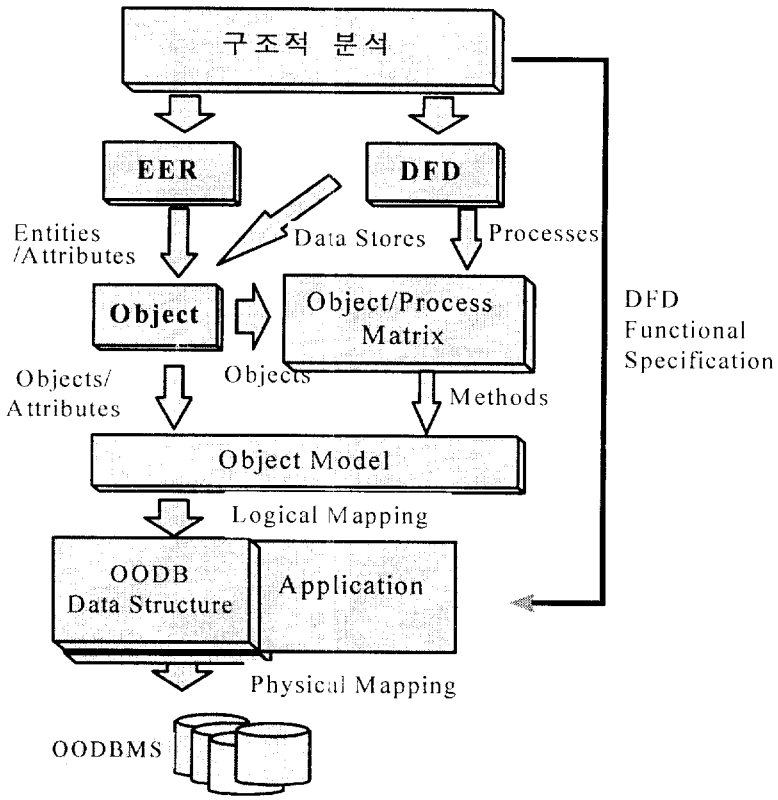
## 3. 객체 모델의 유도

### 3.1 객체 모델 유도 아키텍쳐

구조적 분석 방법의 산출물인 개체-관계 모델은 시스템이 유지하고 처리하여야 하는 데이

타에 대해 일관된 관점을 제공하여 데이타의 중복을 방지할 수 있는 좋은 수단이다. 자료 흐름도는 시스템이 수행하여야 하는 기능을 하위로 세분화해 나갈 수 있는 장점이 있다. 이상의 구조적 방법론의 두 주요 산출물에 근거한 객체 모델 유도 과정에 대한 기본 아키텍처가 [그림 3-1]에 도식화되었다.

즉, 본 방법론에서는 구조적 분석의 산출물




[그림 3-1] 구조적 분석에 의한 객체 모델 유도 아키텍처

인 개체-관계 모델과 자료 흐름도를 입력으로 한다. 개체-관계 모델을 객체의 구조 부분(Structural Part)으로 전환하는 방법에 대한 많은 연구 결과 중 본 아키텍처에서는 Fong [17]의 규칙을 기초로 하였다. 여기에 객체의 행위적 요소 (Behavioral Part)인 메쏘드를 찾기 위해 DFD를 입력으로 객체와 프로세스간 상관관계를 분석하는 기법을 새로이 접목하여 하나의 아키텍처로 제시하였다. 개체-관계 모델의 개체와 개체의 속성을 객체와 그 속성으로 변환하고, 객체와 데이타 저장소에 대한 프로세스의 행렬을 작성하여 프로세스와 객체의 상관관계를 파악한다. 이를 통해 프로세스가 어떠한 객체의 메쏘드로 구현될 것인 지를 규명할 수 있다. 이렇게 유도된 객체 즉, 데이타로서의 개체와 개체에 대한 연산이 되는 메쏘드가 결합하여 객체 모델이 된다. 구현 단계에서 객체 모델은 객체 지향 데이타베이스 (Object-Oriented DBMS, OODBMS)로 구축될 수 있으며 자료 흐름도에 의거한 해당 응용업무의 구조에 따라

객체 지향 응용 시스템이 개발될 수 있다.

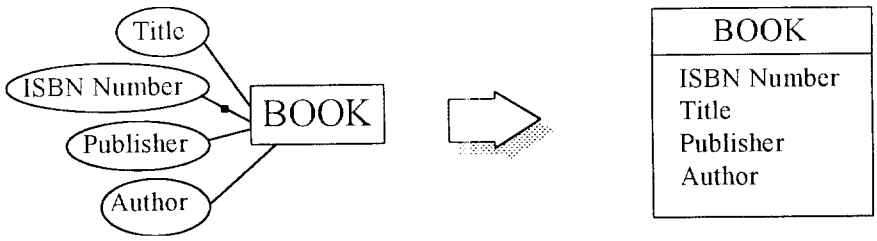
본 설계 아키텍쳐에는 어떠한 객체 모델도 이용 가능하나 본 논문에서는 Rumbaugh의 OMT의 객체 모델 (Object Model) [28] 표기법을 사용하여 개별 객체의 속성과 메쏘드를 표현하였다. 여기서 사용된 개체-관계 모델과 자료 흐름도는 Batini et al.[7]에서의 정의대로 완성된 것을 전제로 하였다.

### 3.2 객체 모델 변환 방법론

#### 3.2.1 개체의 객체로의 변환

일반적으로 객체 중심 방식은 객체의 상태를 나타내는 데이타를 중심으로 그 데이타의 동작까지 하나의 객체로 담는 것이다. 따라서 데이타와 객체는 상호 긴밀한 연관 관계에 있다 [5]. 개체-관계 모델은 데이타의 구조를 정형화하므로 그것에 표현된 데이타에 관련된 행위의 토대로서의 역할도 기대된다. 이러한 접근 방식은 기존의 객체 지향 방법론에서 있을 수 있는 데이타의 중복성 문제를 방지할 수 있다. 아래에서는 개체를 객체로 변환하는 과정에서 적용할 수 있는 다섯 가지의 규칙과 그 내용에 대해 논하기로 한다. 본 논문에서는 설명의 용이상 기존의 여러 가지 개체-관계 모델 중 EER (Extended ER) 모델[34]을 사용했다.

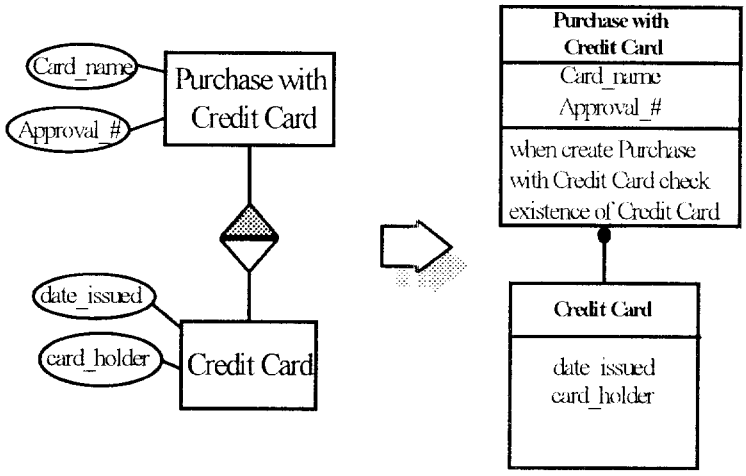
변환 규칙 1: 개체-관계 모델의 개체는 객체 모델의 클래스로 변환된다.




[그림 3-2] 개체에서 객체로의 변환

개체-관계 모델은 개체와 그것의 특성을 결정하는 속성으로 이루어져 있고 객체 모델은 객체와 그 객체를 다른 객체와 구분짓는 속성과 그 객체의 행위를 서술하는 메쏘드로 이루어져 있으므로 객체 모델의 클래스 (Class)는 개체-관계 모델에서의 개체의 개념을 포함한다. 따라서, 개체-관계 모델의 개체는 클래스로 변환된다.

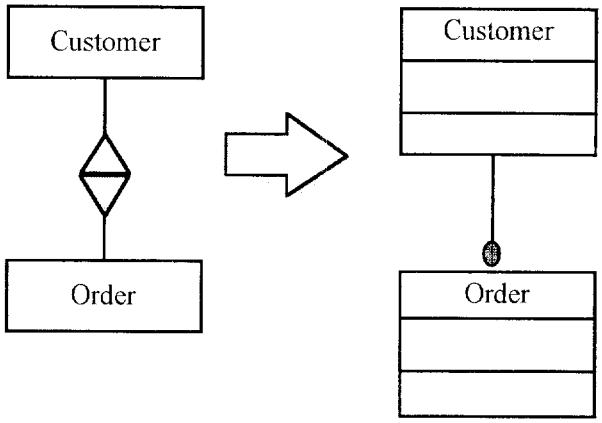
변환 규칙 2: 개체-관계 모델에서 개체의 속성은 객체 모델에서 개별 객체의 속성이 된다.




[그림 3-3] 개체-관계 모델의 속성의 변환

대부분의 객체 지향 프로그래밍 언어에서는 묵시적으로 객체의 유일성을 제공하므로 일반적으로 객체 지향 모델에서는 명시적인 식별자는 필요하지 않으나 개체의 식별자가 개체-관계 모델의 개체에서 의미를 가지고 쓰일 수 있으므로 객체의 속성으로 포함시킨다.

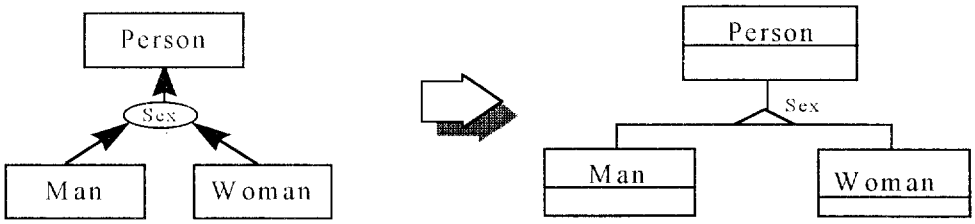
변환 규칙 3: 개체-관계 모델의 관계 (Relationship)는 객체 모델에서의 제휴가 된다.




[그림 3-4] 개체-관계 모델의 관계의 변환

객체간의 제휴는 양방향 (Bi-directional)이며 개념적인 연결이다. 클래스가 개별 객체의 일반화이듯이 하나의 제휴는 가능한 객체간의 연결을 대표한다[28]. 이러한 객체의 제휴는 전혀 새로운 개념이 아니며 개체-관계 모델의 관계로 볼 수 있다. [그림 3-4]에서와 같이 개체 Customer와 Order의 일대다 관계는 객체 Customer와 Order의 일대다 관계로 변환될 수 있다. 일대일 및 다대다 관계도 객체 모델에서의 일대일 및 다대다 관계로 변환이 가능하다.

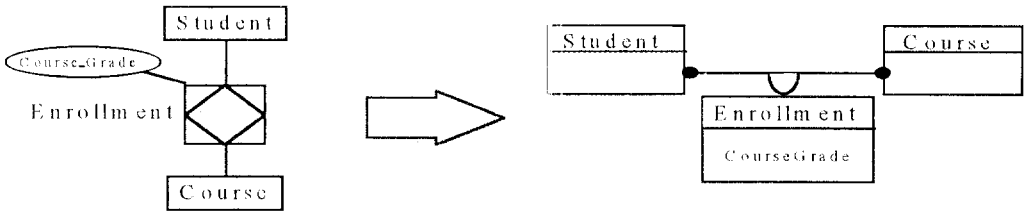
변환 규칙 4: 개체-관계 모델의 일반화 (Generalization)나 계층구조 (Subset Hierarchy)는 객체 모델의 일반화로 표현된다.




[그림 3-5] 일반화

개체-관계 모델의 일반화와 하위의 부분 집합 (Subset)은 각각 상위의 개체에 공통된 속성을 계승하며 하위의 개체는 고유의 속성만을 유지하여 객체 모델의 일반화로 전환할 수 있다. [그림 3-5]에서 개체-관계 모델의 개체 분류자 (Discriminator)인 Sex는 오른 쪽의 객체 모델에서도 구현될 수 있다.

변환 규칙 5: 개체-관계 모델의 다대다 관계 중 개체화 된 관계는 Link attribute로 표현된다.

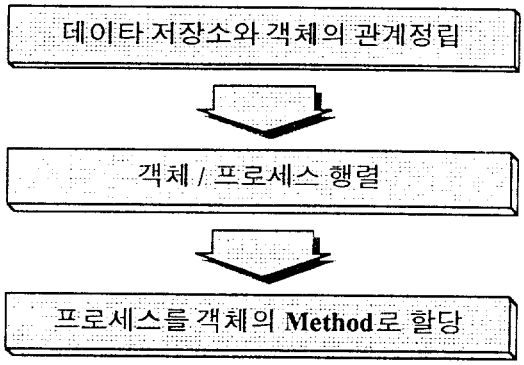



[그림 3-6] 다대다 관계의 변환

개체-관계 모델의 다대다 관계는 양 개체에서 나온 두 인스턴스간의 관계로서 자체가 고유의 속성을 가질 수 있다. 이러한 속성은 제휴에서 링크 (Link)를 표현하는 링크 속성으로 나타낼 수 있다. [그림 3-6]에서 개체화된 관계 (Entitized Relationship)인 Enrollment와 그 속성인 Course_Grade는 객체 모델에서 링크 속성인 CourseGrade로 변환된다.

### 3.2.2 객체의 메쏘드 추출 및 변환

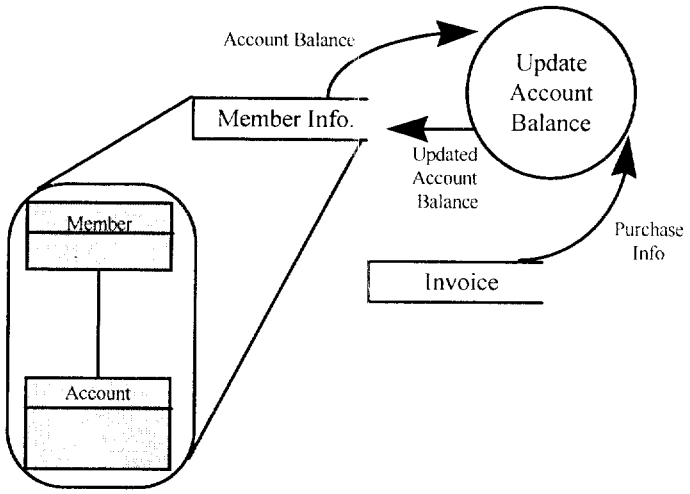
본 절에서는 앞에서 개체-관계 모델에서 변환 규칙에 의해 나온 객체들의 메쏘드를 구현하는 구체적인 방법을 제시하고자 한다. 이 과정은 [그림 3-7]과 같이 크게 3단계로 이루어져 있다. 우선 자료 흐름도의 데이타 저장소와 객체들의 관계를 명확히 한 후 이 데이타 저장소와 데이타를 주고 받는 프로세스가 어떠한 객체의 메쏘드가 될 것인 지를 규명하기 위하여 객체와 프로세스의 행렬을 만든다. 이후 마지막 단계에서 행렬을 작성하여 개별 객체의 메쏘드로 소속시킨다.




[그림 3-7] 메쏘드를 구현하는 3단계

단계1: 데이타 저장소와 객체의 관계를 정립한다.

구조적 분석 방법에서는 상위의 프로세스를 하위의 프로세스들로 기능을 분할하여 세부 명세를 정의한다. 이처럼 프로세스 중심의 하향식 분할 방식으로 인해 프로세스들과 관련된 데이타들이 자료 흐름도 안에서 산개하여 존재하게 된다. 프로세스들이 데이타를 중심으로 존재하는 것이 아니라 그 프로세스들이 구성하는 상위의 프로세스와 연관되어 묶이게 되는 것이다. 따라서 자료 흐름도 내의 데이타 저장소는 하나의 데이타 구조 단위 즉, 객체가 아니라 객체의 일부이거나 또는 여러 객체들과 그 객체들의 관계로 이루어진 경우가 대부분이므로 자료 흐름도의 데이타 저장소에 해당 하는 객체 모델의 해당 부분이 실제 객체들과 어떤 관계인 지를 파악하여야 한다. 이러한 대응 관계는 자료 흐름도에서 자료 저장소와 프로세스 사이의 데이타의 흐름을 살펴 보면 용이하게 파악할 수 있다. 예를 들면, [그림 3-8]에서 프로세스와 자료 저장소 Member Info. 사이의 Account Balance에 관련된 데이타 흐름에서 자료 저장소 Member Info. 가 단순히 객체 Member가 아니라 객체 Account도 포함하고 있음을 알 수 있다.



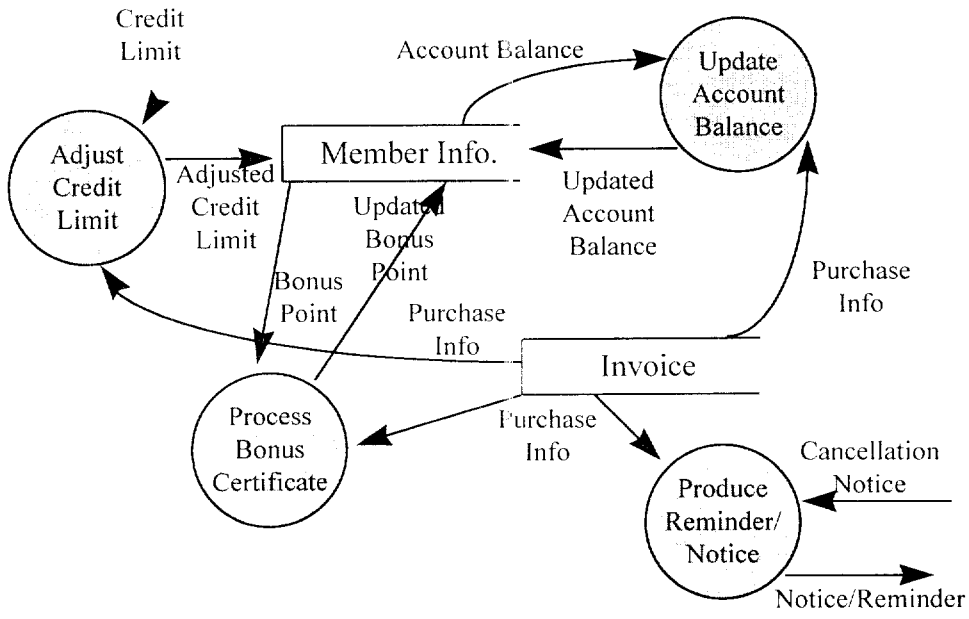

[그림 3-8] 자료 저장소와 객체의 관계

결과적으로 자료 흐름도 내의 데이타 저장소는 하나의 데이타 구조 단위가 아니라 객체의 일부이거나 또는 여러 객체들과의 관계로 이루어지므로 하나의 프로세스는 여러 객체의 메쏘드로 구현되기도 한다. 따라서 프로세스의 적절한 대상 객체를 찾는 일이 중요하다. 이를 위하여 객체와 자료 저장소 사이의 행렬이 작성된다.

단계2 : 단계1에서 정립된 자료 저장소의 객체를 중심으로 프로세스와 객체의 행렬을 작성한다.

자료 흐름도에서 프로세스의 대상이 되는 객체를 찾는 방법으로는, [5]에서와 같이 직관적으로 프로세스의 입력이나 프로세스에 의해 변환되어 나오는 출력을 이용할 수 있다. 그러나 구조적 분석에서는 프로세스 기능 분할의 특성상 프로세스가 여러 개의 메쏘드로 구현되거나 다수의 프로세스가 하나의 메쏘드의 일부로 될 경우가 많으므로 이와 같은 방식은 적절하지 않다.

본 논문에서는 개별 프로세스와 객체 사이의 관계를 식별하기 위해 객체/프로세스 행렬을 사용했다. 객체/프로세스 행렬에서 각 프로세스와 객체가 만나는 교차 칸은 그 프로세스에 의해 주어지는 개별 객체의 역할을 표시하게 된다. 즉, 프로세스가 수행하여야 할 역할이 개별 객체의 메쏘드로 산개하여 소속되고 프로세스는 이러한 객체의 메쏘드들이 연쇄적으로 작용하여 이루어지게 된다. 만약, 특정 프로세스가 어떤 객체의 내용을 필요로 한다면 해당 객체의 일부의 값 (속성)을 그 프로세스가 메쏘드로 속하는 객체에서 필요로 하는 것이므로 그 객체는 해당 프로세스에 필요한 데이타를 객체 외부에서 접근할 수 있도록 해주는 메쏘드를 가져야 한다. 하나의 프로세스가 복수의 객체와 관련을 맺게 되면 그 복수의 객체에 속하는 메쏘드들이 연쇄적으로 작용하여 프로세스를 수행하게 된다. 예를 들면 [그림 3-9]의 프로세스 Adjust Credit Limit의 경우 객체 Invoice의 내용을 참조하여 객체 Member의 속성인 Purchase-Limit을 갱신한다. [그림 3-10]은 [그림 3-9]를 기초로 작성된 객체/프로세스 행렬이다. 행렬에서 Read는 프로세스가 객체를 참조한다는 의미이며 Update는 프로세스에 의해 객체의 값이 변경된다는 것을 나타낸다. 이외에도 본 예에는 없으나 삭제 (Delete)와 삽입 (Insert)이 중요한 행렬 내부 값이다.




[그림 3-9] 자료 저장소와 프로세스

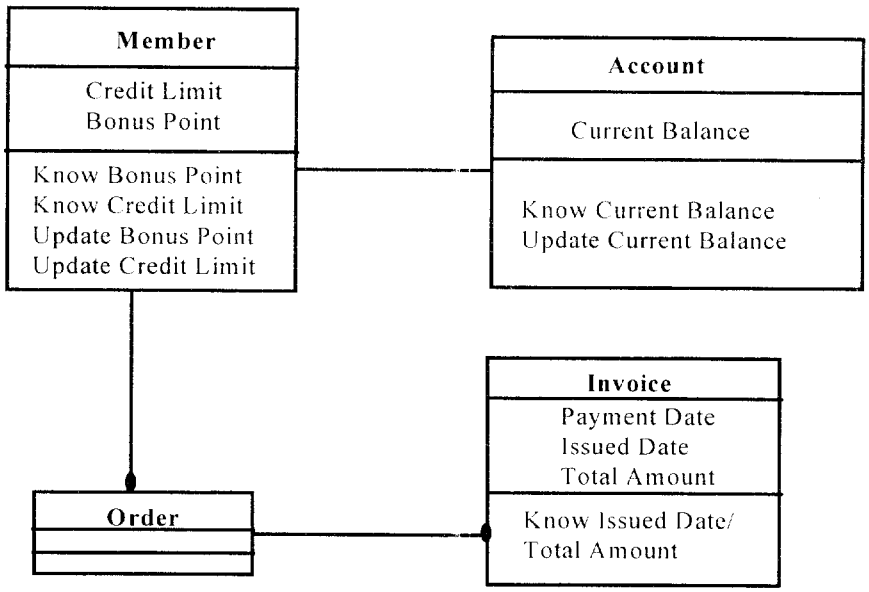
| 객체 / 프로세스 | Member | Account | Invoice |
|---|---|---|---|
| Adjust Credit Limit | Read<br>Update | | Read |
| Update Account Balance | | Read<br>Update | Read |
| Process Bonus Certificate | Read<br>Update | | Read |
| Produce Reminder /Notice | | | Read |

[그림 3-10] 객체/ 프로세스 행렬

단계3 : 프로세스들을 단계2에서 규명된 객체의 메쏘드로 소속시킨다.

단계2에서 프로세스의 형식은 동사와 데이타의 융합된 형태이다. 예를 들면 Adjust Credit Limit는 Adjust라는 동사와 Credit Limit라는 명사로 조합되어 있다. 이러한 프로세스와 이 프로세스의 데이타 입출력에 관한 사항이 자료사전에 정의되어 있으므로 개별 프로세스의 대상이 되는 각각의 객체와 객체의 속성을 발견할 수 있게 된다. 단계3에서 이러한 프로세스를 해당 객체의 메쏘드로 소속시킨다.

[그림 3-10]의 객체/프로세스 행렬에서 프로세스에 의한 객체의 Read는 해당 객체가 가지고 있는 데이타의 속성 값을 읽는다는 의미로서 외부 또는 다른 객체에게 자신의 내부 상태를 알려주는 역할을 한다. 따라서 Know라는 이름을 갖는 메쏘드로 전환되게 된다. 또한, 객체를 Update하는 프로세스는 객체 내부의 속




[그림 3-11] 객체의 메쏘드로 소속된 프로세스

성 값을 변경하는 행위로서 Update 메쏘드로 전환된다. 예에서 보는 바와 같이 본 방법에서는 객체의 동적인 특성을 반영하는 메쏘드 보다는 내부 정보관리를 위주로 하는 메쏘드가 중심임을 알 수 있다. 이는 자료 흐름도 상의 정적인 요소인 자료 저장소를 중심으로 객체를 추출한데 기인한다. 이와는 달리 동적인 역할을 수행하는 객체에 중점을 두기 위해서는 일차적으로 유도된 객체 모델을 기초로 객체의 책임 (Responsibilities) [32]에 관한 요소를 가미하는 방향으로 보완이 필요할 것이다.

### 3.3 방법론의 유효성

데이타와 프로세스를 통합하는 방법 중 객체 모델로의 변환법들을 비교 분석하였다. 2.2절에서 설명된 기존의 변환 방법들과 본 고에서 제안된 방법을 비교 평가하여 본 방법론의 유효성을 검증하였다.

각 방법론을 비교 분석한 결과를 〈표 3-1〉과 같이 정리할 수 있으며 세부적으로 설명하면 다음과 같다. 객체의 정의면에서 대부분의 방법론들이 개체-관계 모델을 이용하며 부분적으로 자료 사전과 자료 저장소를 활용한다. 본 논문의 방법론에서도 객체 정의를 위해 개체-관계 모델을 활용한다. 객체의 속성은 개체-관계 모델의 개체의 속성 및 자료 사전에 정의된 명세가 객체의 속성으로 정의된다. 그러나 Bailin과 Wieringa[35]의 방법에서는 객체 속성에 대한 정의가 곤란하다.

객체의 일반화 관계도 개체-관계 모델을 기초로 한 방법의 경우 객체 모델로의 전환에 문제가 없다. 그러나 자료 저장소를 이용하는 Wieringa[35]의 방법에는 적용되지 않는다. 또한, Alabiso[5]는 OSC (Object Structure Chart)를 통해 일반화를 표현한다. 기능 분할

〈표 3-1〉 방법론 비교 분석

| 방법론 / 비교기준 | Alabiso (1988) | Bailin (1989) | Choobineh (1991) | Wieringa (1991) | Nachouki et al.(1991) | Elmasri et al. (1994) | Fong (1995) | 이희석/배한욱/유천수(1996) |
|---|---|---|---|---|---|---|---|---|
| 객체의 정의 | 자료사전에서 OSC 작성 | 개체-관계 모델 이용 | 개체-관계모델에서OOERD작성 | 자료흐름도의 자료저장소 이용 | 개체-관계 모델 이용 | 개체-관계 모델 이용 | 개체-관계 모델 이용 | 개체-관계 모델 이용 |
| 개체의 속성 | 자료사전 이용 | N/A | 개체-관계 모델 이용 | N/A | 개체-관계 모델 이용 | 개체-관계 모델 이용 | 개체-관계 모델 이용 | 개체-관계 모델 이용 |
| 일반화 | OSC | 개체 관계 모델 이용 | 개체 관계 모델 이용 | N/A | 개체 관계 모델 이용 | 개체 관계 모델 이용 | 개체 관계 모델 이용 | 개체 관계 모델 이용 |
| 기능의 분할 | 자료흐름도에서 FDC 작성 | N/A | N/A | 자료흐름도 | N/A | N/A | N/A | 자료 흐름도 |
| 메쏘드의 구현 | 자료 흐름도의 프로세스를 획일적으로 적용 | N/A | N/A | 체계적인 방법 부족 | Logical Access Path고려 | N/A | N/A | 객체/프로세스 행렬 |
| 프로세스와 메쏘드의 관계 정립 | N/A | N/A | N/A | N/A | N/A | N/A | N/A | 객체/프로세스 행렬 |
| 최종 산출물 | OSC, FDC | EDFD | OOERD | Module Diagram | O2 OODBMS용 OO Schema | 가상적(Virtual) OO모델 이용 | OMT 메쏘드 없음 | OMT |

OSC: Object Structure Chart　　FDC: Funotional Design Chart
EDFD: Entity DFD　　N/A: Not Applicable

에 있어서는 구조적 방법론의 원리에 의거 자료 흐름도를 이용한다. 그러나 개체-관계 모델을 방법론 전체의 기초로 활용하는 Choobineh 및 Fong[17]의 방법에서는 가능하지 않다.

프로세스와 메쏘드의 관계를 정립함에 있어 Nachouki et al.[27]는 논리적인 접근 경로(Logical Access Path)를 분석하여 이를 스키마에 함께 표현하는 방식을 사용하고 있는 외에 타 방법론에서는 별도의 방법이 존재하지 않는다. 이에 비해 본 논문에서 제안한 방법론에서는 객체/프로세스 행렬을 활용하는 방안을 제시했으며 유효하게 활용될 수 있음을 입증하였다. 메쏘드를 찾아서 객체와 연결하는 문제에 있어서도 자료 흐름도의 프로세스를 획일적으로 적용하는 Alabiso[5] 방법 외에는 적용할 수 있는 방법을 제공하고 있지 못하다. 본 논문에서는 객체/프로세스 행렬을 객체와 메쏘드를 연결하는 방법으로 제안했다. 최종 산출물은 각 방법론에서 선정된 다양한 객체 모델이 이용되고 있다. 특히, Elmasri et al.[16]는 가상적 객체 모델 (Virtual OO Model)을 최종 산출물로 하여 특정한 객체 모델에 종속되지 않도록 하며, Nachouki et al.[27]는 O2 DBMS의 객체 모델을 대상으로 하여 두 방법 모두 객체 스키마 수준까지 작성하는 특징을 갖는다.

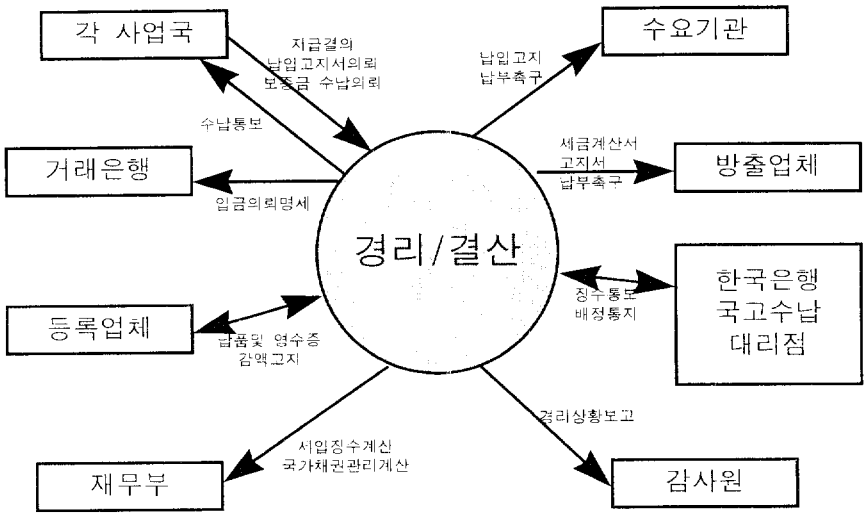
결국, 본 방법론은 데이타베이스의 개념적 설계도인 개체-관계도를 기초로 하고 객체 모델에서 요구되는 메쏘드를 자료 흐름도로부터 파악하여 객체 지향 데이타베이스 설계도로 활용될 객체 모델을 도출한 면에서 초유의 방법이다. 즉, 데이타베이스의 저장 자료와 같이 지속성 (Persistence)을 요구하는 객체에 대한 모델을 작성하는 방법을 제안했다는 점이다. 특히, 구조적 분석의 주요 산출물인 자료 흐름도를 이용하여 프로세스를 객체와 연결하고 개별 프로세스가 어떠한 객체에서 어떠한 역할을 수행하여야 하는 지를 분석하여 객체의 메쏘드를 규명한 점은 타 방법론으로의 연계 가능성도 크다 하겠다.

# 4. 사례 분석

## 4.1 사례 개요 및 범위

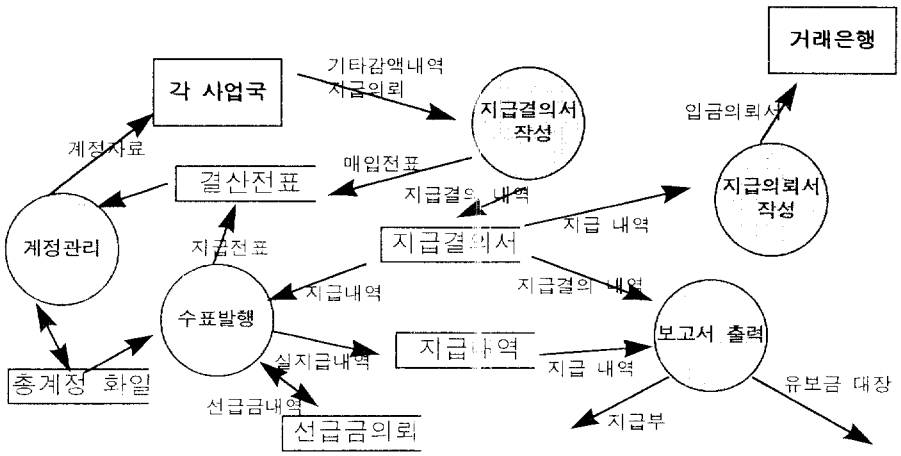
정부 조달업무 담당 청에서는 Unisys의 DBMS인 MAPPER를 사용하여 응용 업무가 개발되었는 바 단순 통계 및 집계용이 주종이었다. 본청과 지방청을 연결하는 네트워크도 열악하여 지청과 본청간 원활한 업무의 교류가 힘든 형편이었다. 이에 따라 본청에서는 1993년 종합개발을 위해 외부 용역을 통해 종합개발 계획서를 93년 11월에 작성하고 1994년 용역 사업을 추진하여 94년 8월 S업체와 계약, 95년 3월에 조달 행정 통합 정보관리 시스템 구축을 완료하였다. 본 논문에서는 이 구축 프로젝트의 구조적 산출물인 개체-관계도와 자료 흐름도를 이용하여 제시된 방법론을 이용한 객체 지향 모델로의 전환이 가능함을 보이고자 한다. 본 조달업무는 여러 가지 하위의 세부 기능으로 나뉘어지며 본 논문에서는 예시 목적상 이들 중 경리/결산 부문을 중심으로 적용한 내용을 소개하고자 한다.

경리/결산 부분의 자료 흐름도상의 배경도 (Context Diagram)가 [그림 4-1]에 도식화 되었다. 경리/결산의 서브 프로세스는 다른 시스템에서 발생되는 고지서 및 수납 사항에 대해 자동 전표 처리를 하여 결산 파일을 생성한다.

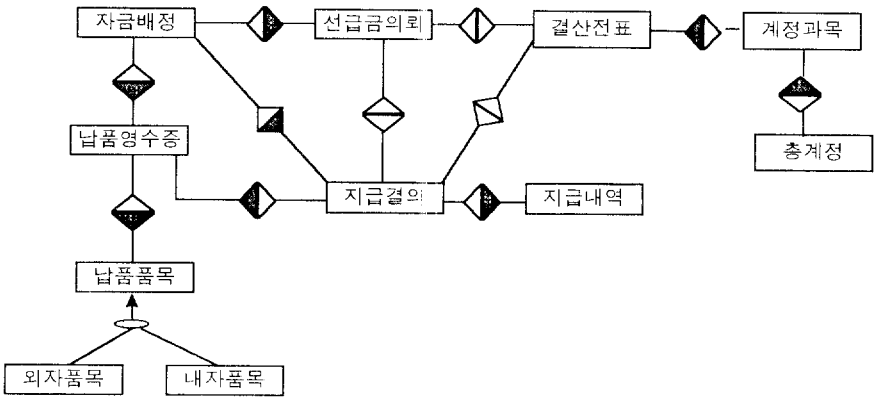



[그림 4-1] 경리/ 결산의 배경도

즉, 모든 시스템에서 발생되는 고지서 내역 및 수납 현황이 실시간(Real-time)이나 하루 단위의 배치 작업으로 처리됨으로써 결산 시스템과 연계 처리되어 결산 작업이 단축되는 효과를 목표로 한다. 본 사례 분석에서는 경리/결산의 8개의 하위 프로세스 중 지급 프로세스에 해당하는 자료 흐름도 및 개체-관계 모델을 대상으로 하였다 ([그림 4-2], [그림 4-3]). 단, 개체-관계 모델에서 속성은 자료 사전으로 별도로 표기되어 본 예에서는 명시되어 있지 않다.



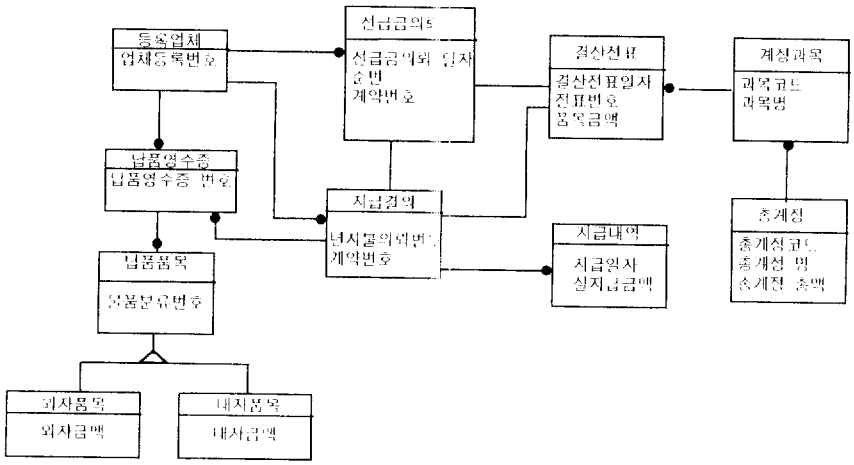

[그림 4-2] 지급 프로세스의 자료 흐름도




[그림 4-3] 지급 프로세스의 개체-관계도

### 4.2 객체 모델로의 변환 과정

우선, [그림 4-3]의 개체-관계 모델을 메쏘드를 제외한 객체로 변환한 내용이 [그림 4-4]에 도식화되었다. 총 9개의 객체가 유도되며 각 속성은 자료 사전에서 정의한 개체-관계 모델의 속성으로부터 유도되었다.




[그림 4-4] 지급 부문의 객체 모델

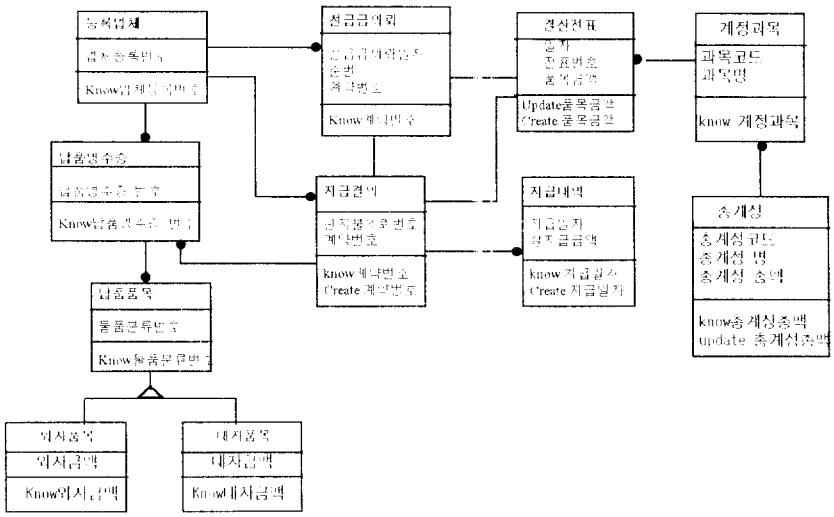
다음으로 [그림 4-2]의 프로세스와 [그림 4-4]의 객체를 사용하여 [그림 4-5]와 같은 객체/프로세스 행렬을 작성할 수 있다. 예를 들어, 지급결의 작성 프로세스는 지급결의 객체를 생성 (Create)하게 된다. 이와 같은 방식으로 수표발행 프로세스는 결산전표 객체를 생성하고 객체의 속성 값을 갱신 (Update)하기도 한다. 이외 대부분의 프로세스는 객체를 단순히 참조 (Read)함이 보여진다.

| 객체 / 프로세스 | 지급내역 | 결산전표 | 지급결의 | 선금금의뢰 | 납품품목 | 납품영수증 | 등록업체 | 외자품목 | 내자품목 | 계정과목 | 총계정 |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 지급결의서 작성 | | | C | R | R | | R | R | R | | |
| 입금의뢰서 작성 | | | R | | | | | | | | |
| 수표 발행 | C | C,U | R | R | | | | | | | R |
| 보고서 출력 | R | | R | | | R | | | | | |
| 계정 관리 | | | | | | | | | | R | U |

C:Create, R:Read, U:Update, D;Delete

[그림 4-5] 결산 프로세스 부분의 객체/ 프로세스 행렬

이러한 객체/프로세스 행렬과 기존 시스템의 구현 과정에서 나온 다른 산출물들인 업무 기능 분해도, 입출력 자료 조사서, 사용자 면담 기록, 자료 사전 등을 이용하여 [그림 4-6]과 같이 개별 객체에 메쏘드를 할당할 수 있었다.




[그림 4-6] 메쏘드가 추가된 지급 부문의 객체 모델

## 4.3 사례 적용 결과 요약

본 논문에서 제시된 구조적 분석의 산출물을 이용하여 객체 모델로 전환하는 방법론을 실제 사례에 적용하여 보았다. 실제 예에 적용하여 본 결과, 프로세스들을 개별 객체의 메쏘드로

적절히 분산 배치하는 것이 가능했다. 또한 자료 흐름도와 개체-관계 모델뿐만 아니라 업무 기능 분해도, 입출력 자료 조사서, 사용자 면담 기록, 자료 사전 등의 다른 산출물들도 유용하게 사용될 수 있음을 확인할 수 있었다. 개체-관계 모델에서 개체를 객체로 바꾸는 것 보다 객체/프로세스 행렬을 구성하여 특정 프로세스가 어떠한 객체와 객체의 속성에 영향을 미치는 지를 분석하기 위해서는 비교적 다양한 자료를 관찰하고 조사하여야 하며 개발자의 주의가 보다 많이 요구되는 부분이라는 사실을 알 수 있었다. 이러한 사실은 일반적으로 데이타가 조직 변화에 민감하지 않고 정적인 반면, 프로세스가 주 변화 대상이 되는 점을 감안할 때 기대되던 결과이다.

본 사례 적용을 통하여 기존의 구조적 분석 산출물들을 기초로 객체 지향 분석의 주요 산출물인 객체 모델을 유도할 수 있다는 사실이 확인되었다. 이로써 새로운 시스템의 도입 시 기존 시스템을 구현하는 데에 투입되었던 노력과 자원이 충분히 재활용될 수 있을 것으로 기대된다.

## 5. 결 론

본 연구에서는 구조적 분석 방법을 이용하여 작성된 산출물을 기초로 객체 모델을 유도하는 단계별 방법론을 제시하였다. 구조적 분석 방법에서 출발하여 객체 모델을 작성하는 기존의 방법론들은 데이타의 행위를 다루는 프로세스 측면의 고찰이 부족하며 시스템이 구현해야 할 기능들을 하위의 기능으로 나누어 구체적인 각 하위의 기능들을 세부적으로 명세화하는 면이 미약했다. 기존의 구조적 분석에서는 프로세스들을, 그 프로세스들이 협력하여 상위의 기능을 수행하도록 모아서 분석함으로써 데이타를 중심으로 그 데이타의 행위를 서술하는 객체 모델로의 전환이 용이하지 않았다. 본 논문에서는 이러한 구조적 분석의 프로세스를 객체와 연결하고 개별 프로세스가 어떠한 객체에서 어떠한 역할을 수행하여야 하는 지를 분석하여 자료 흐름도에서 묵시적, 명시적으로 표현하고 있는 객체의 메쏘드를 규명하였다. 이러한 분석 작업을 위하여 자료 흐름도와 개체-관계 모델뿐만 아니라 구조적 분석의 다른 산출물인 자료 사전 및 업무 기능 분해도 등도 적절히 사용될 수 있음을 실제 사례의 적용을 통하여 예시하였다.

객체 모델이 많은 장점에도 불구하고 완전히 다른 개발 사고 방식 (Paradigm Shift)을 요구함에 기인하여 기존의 개발자들이 겪을 수 있는 어려움이 본 연구에서 제안한 방법론을 적용할 경우 완화될 것이 기대된다. 또한 객체 지향 기술을 적용하여 구현되는 새로운 시스템의 도입 시에도 기존의 자료를 충분히 활용할 수 있어 기존 시스템을 구축하는 데에 투입되었던 자원을 유용하게 재활용할 수 있을 것으로도 판단된다.

본 논문에서 제시된 방법론에 근거하여 향후 몇 가지 주요 연구과제가 주목된다. 우선, 객체/프로세스 행렬과 함께 객체의 메쏘드를 규명하는 효과적인 방안에 대한 보완 연구가 필요하다. 이 단계에서는 보다 구체적인 규칙들이 제시되어야 할 것이다. 또한 기존의 구조적 분석 결과를 단순히 객체 모델로 바꾸는 것뿐만 아니라 기능 개선 등 확장의 동기가 되는 제요소를 적절히 반영한 객체 모델을 만들 수 있는 방법론에 대한 연구도 필요하다. 한편 본

연구에서는 데이타 중심의 객체 모델링이 중심이 되었으나 프로세스 중심 객체 모델링의 경우와의 비교 분석 및 이의 접목 등이 향후 연구과제의 하나이다. 이를 위하여 시나리오(Scenario) 분석을 통한 책임 요소의 접목도 관심 대상이라 하겠다[36].

## 참 고 문 헌


[1] 민대환, 권순호, "EER 모델을 이용한 객체 지향 데이타베이스 설계", 「데이타베이스 심포지움 및 학술대회 논문집」, 1995, pp. 281-297.

[2] 삼성데이타시스템, 「기능 사양 보고서(경리/결산관리)」, 조달 행정 통합 전산 관리 시스템, 1994.

[3] 이희석, 배한욱, 유천수, "구조적 분석 스키마에 의한 객체 모델 구현", 「한국 경영정보학회 춘계 학술대회 논문집」, 1996, pp. 118-137.

[4] 주경수, 심현숙, "객체 지향 기법을 위한 관계 데이타베이스 변환 방법", 「데이타베이스 심포지움 및 학술대회 논문집」, 1995, pp. 223-235.

[5] Alabiso, B., "Transformation of Data Flow Analysis Models to Object-Oriented Design", *Proceeding of the Conference on Object-Oriented Programming: System, Language, and Applications-OOPSLA*, 1988, pp.335-353.

[6] Bailin, S. C., "An Object-Oriented Requirements Specification Method", *Communications of the ACM*, Vol.32, No.5(1989), pp.608-623.

[7] Batini, C., S. Ceri, and S. B. Navathe, Conceptual Database Design, *An Entity-Relationship Approach*, The Benjamin/Cummings Publishing Company, Inc., 1992.

[8] Blaha, M. R., W. J. Premerlani, and J. E. Rumbaugh, "Relational Database Design using an Object-Oriented Methodology", *Communications of the ACM*, Vol.31, No.4(April 1988), pp. 414-427.

[9] Bock, B. and T. Ryan, "Accuracy in Modeling with Extended Entity Relationship and Object-Oriented Data Models", *Journal of Database Management*, Vol.4, No.7(1993), pp.30-39.

[10] Booch, G., *Object-Oriented Analysis and Design with Application*, The Benjamin/Cummings Publishing Company, Inc., 1994.

[11] Burleson, D., "Adding Behaviors to Relational Database", *DBMS*, September 1995, pp. 68-76.

[12] Campbell, J. A. and J. J. Campbell, "The Object-Oriented Design and Implementation of a Relational Database Management System", *Journal of Object-Oriented Programming*, July-August 1995, pp. 43-47.

[13] Choobineh, J. and K. Gorman, "The Object-Oriented Entity-Relationship Model(OOERM)", *Journal of Management Information System*, Vol.7, No.3

(Winter 1990-1991), pp.41-65.

[14] Coad, P. and E. Yourdon, *Object-Oriented Design*, Prentice-Hall Inc., 1991.

[15] Date, C. J., *An Introduction to Database System*, Addison-Wesley Publishing Company, Inc., 1995.

[16] Elmasri, R. and S. B. Navathe, *Fundamentals of Database Systems*, Benjamin/Cummings, Redwood City, 1994.

[17] Fong, J., "Mapping Extended Entity Relationship Model to Object Modeling Technique", *SIGMOD Record*, Vol.24, No.3(September 1995), pp.18-22.

[18] Frank, M., "Object-Relational Hybrid", *DBMS*, July 1995, pp. 46-56.

[19] Gala, S. and W. Kim, "Database Design Methodology: Converting a Relational Schema into an Object Relational Schema", *Database Journal*, Vol.1, No.1(1992).

[20] Jackson, R. B., D. W. Emblet, and S. N. Woodfield, "Developing Formal Object-Oriented Requirements Specification: A Model, Tool and Technique", *Information System*, Vol.20, No.4 (1995), pp.273-289.

[21] Li, W., S. Henry, D. Kafura, and R. Schulman, "Measuring Object-Oriented Design", *Journal of Object-Oriented Programming*, July-August 1995, pp. 48-55.

[22] Ling, T-W, P. K. Teo, and L. L. Yan, "Generating Object-Oriented views from an ER-Based Conceptual Schema", *Proceeding of Very Large Database-VLDB*, 1993.

[23] Linthicum, D. S., "The Object Revolution", *DBMS*, October 1995, pp. 46-52.

[24] Markowitz, V. M., "Representing Processes in the Extended Entity-Relationship Model", Proc. of the 6th Conference on Data Engineering, IEEE Computer Society Press, 1990.

[25] Martin, J. and J. J. Odell, *Object-Oriented Method: A Foundation*, Prentice-Hall Inc., 1995.

[26] Martin, J., *Information Engineering, Book II: Planning And Analysis*, Prentice-Hall Inc., 1990.

[27] Nachouki, J., M. P. Chastang, and H. Briand, "From Entity-Relationship diagram to an object-oriented database", *Proc. 10th Int. Conf. On the Entity-relationship Approach*, San Mateo, 1991, pp. 459-482.

[28] Rumbaugh, J. E., M. Blaha, W. Premerlani, F. Eddy, and W. Lorensen, *Object-Oriented Modeling and Design*, Prentice-Hall Inc., 1991.

[29] Rumbaugh, J. E., "OMT: The Development Process", *Journal of Object-Oriented Programming*, May 1995, pp. 8-16.

[30] Rumbaugh, J. E., "OMT: The Function Model", *Journal of Object-Oriented Programming*, March-April 1995, pp. 10-14.

[31] Shlaer, S. and S. J. Mellor, *Object Lifecycles Modeling the World in States*, Yourdon Press, 1992.

[32] Taylor, D. A., *Business Engineering with Object Technology*, John Wiley & Sons, Inc., 1995.

[33] Taylor, D. A., *Object-Oriented Technology: A Managers Guide*, Addison-Wesley Publishing Company, Inc., 1990.

[34] Teorey, T. J., D. Yang, and J. P. Fry, "A Logical Design Methodology for Relational Databases Using the Extended Entity-Relationship Model", *ACM Computing Surveys*, Vol.18, 2(June 1986), pp.197-222.

[35] Wieringa, R. J., "Object-Oriented Analysis, Structured Analysis, and Jackson System Development", *Proceedings of the IFIP Working Conference on the Object-Oriented Approach in Information System*, 1991.

[36] Wisnsberg, P., "A Comparison of Data and Responsibilities Driven Methods: Essential Difference", *Database Programming and Design*, Vol.8, No.12 (1995), pp.23-24.

[37] Yourdon, E., *Modern Structured Analysis*, Prentice-Hall, 1989.

[38] Zand, M., V. Collins, and V. Caviness, "A Survey of Current Object-Oriented Database", *Data Base Advances*, Vol.26, No.1(February 1995), 14-29.