다측면 객체 모델의 정형화에 관한 연구

조동영* · 황종선* · 백두권** · 손진곤**

* 고려대학교 전산과학과

** 한국방송통신대학 전자계산학과

< 요 약 >

다측면 객체모델은 개념적 데이타베이스 모델인 EA 모델을 확장한 객체 중심 데이타 모델로서, 여러 측면에서 관측되고 표현될 수 있는 실세계의 동일객체들을 직접 데이타베이스의 동일객체로 모델링할 수 있도록 하여 데이타베이스의 개념적 스키마 설계를 용이하게 한다.

본 논문에서는 이러한 다측면 객체모델의 기본개념들을 설명하고, 수학적 접근에 의한 형식론을 구성하였으며, 그 모델의 구성을 위한 단계적 접근 방법을 소개하였다.

I. 서 론

실세계의 데이타 객체들은 모델구성자들의 관점에 따라 여러 측면에서 관측되고 표현될 수 있다. 그러나 지금까지의 대부분의 개념적 데이타 모델들은 실세계의 객체들에 대해 고정된 한 측면의 표현만을 허용하기 때문에 다측면으로 모델링될 수 있는 실세계 객체의 표현에 어려움을 갖는다[5,11,12]. 특히, 객체 표현에 대한 단일측면만의 허용은 데이타베이스 설계과정에 있어서의 뷰 통합(view integration)이나, 상향식 접근에 의한 분산데이타베이스 구성에 있어서의 분산스키마 통합(distributed data schema integration)을 더욱 어렵게 한다[2,6,8,11].

일반적으로 대부분의 개념적 데이타 모델들은 데이타베이스 스키마의 설계를 용이하게 하지만 직접 모델을 지원하는 DBMS 의 부재로 실제 데이타베이스의 구성시에는 직접 DBMS 의 지원을 받는 구현모델의 스키마로 변환되어야 한다[4,6]. 수학의 집합(set)과 관계(relation) 이론에 의한 이론적 토대를 갖춘 관계형 모델(relational model)은 가장 널리 사용되는 데이타

모델이다. 데이타 모델에 대한 수학적 이론에 의한 형식론의 구성은 모델 구조와 모델링 과정에 있어서 의미표현의 모호성을 피할 수 있으며, 특히 모델을 직접 지원할 수 있는 소프트웨어의 개발에 유용하다[7].

따라서 데이타베이스 시스템의 구축을 보다 용이하게 하기 위해서는 궁극적으로 보다 정확한 데이타 서맨틱 구문들을 제공하고 또 직접 DBMS 의 지원을 받는 모델이 개발이 필요하며, 이를 위해서는 먼저 모델에 대한 형식론의 구성이 요구된다.

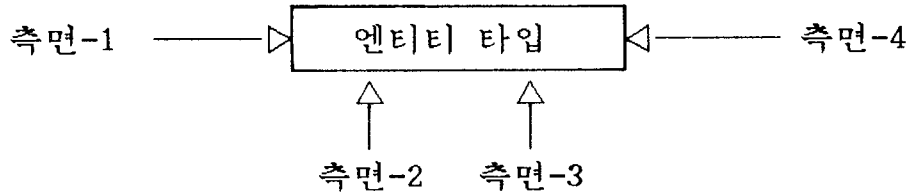
본 논문에서는 백두권, 김창화 등에 의해 처음으로 제안되었던 EA 모델링의 기본 개념을 토대로 보다 정확한 데이타 서맨틱 개념들을 제공하는 다측면 객체모델(Multi-Aspect Object Model:MAOM))의 개념을 설명하고 수학적 접근을 토대로 MAOM 형식론을 구성하였으며, 이 모델의 구성을 위한 단계적 접근방법을 소개하였다.

II. 다측면 객체 모델링의 개념

2.1 EA 모델의 한계

EA(Entity-Aspect) 모델은 모델링 객체의 엔티티와 측면 구조를 분해, 특수화, 다중성의 세가지 추상화 개념을 이용하여 나타내는 계층적 트리 구조의 정적 모델로서 실세계의 데이타 객체들을 여러 측면의 세부적인 계층구조로 표현할 수 있는 장점을 갖는다. 그러나 EA 모델은 모델링 과정에 있어서 구성타입간의 모순, 시스템 구성의 복잡성, 일관성 유지의 어려움 등의 문제점을 가지며, 특히 데이타 객체들간의 관계성의 일관된 표현에 많은 문제점을 나타내고 있다. 따라서 EA 모델은 인공지능의 지식 표현 모델로는 적합하지만 일반적인 데이타베이스의 개념적 스키마 설계를 위한 개념적 모델로서는 부적절하다.

본 논문에서 제안하는 다측면 객체모델은 '실세계의 동일객체가 모델링의 관점에 따라 여러 측면으로 표현될 수 있다.'는 EA 모델의 기본 개념과 인식을 같이하면서도 개념적 데이타 스키마의 효율적 설계를 지원한다.



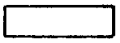
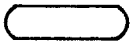
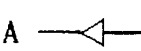
<그림 1> EA 모델과 MAOM 의 공통 개념

2.2 다측면 객체 모델링의 개념

실세계는 다측면의 관측이 가능한 많은 데이타 객체들로 구성되어 있다. MAOM 은 '동일 객체가 여러 측면으로 관측되어 각기 다르게 이해 되더라도

결국은 동일 객체이기 때문에 공통된 표현을 갖는다.' 라는 인식을 토대로 하는 객체중심 데이타 모델이다.

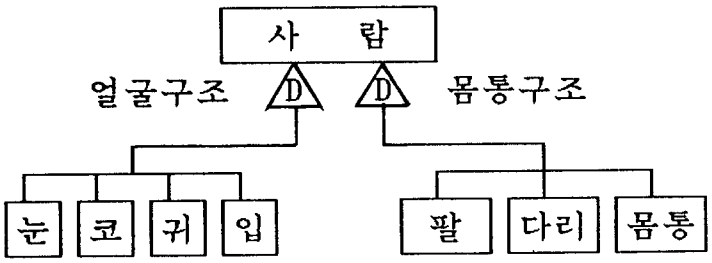
MAOM 에서는 실세계의 데이타 객체들을 객체타입의 개념과 각 객체타입에 대한 측면의 개념을 이용하여 다측면 객체들을 표현한다. 객체타입은 모델 구성자에 의해 독립적으로 인식될 수 있는 모델링 객체인 엔티티 타입(entity type)과 여러 엔티티 타입들간의 관계성에 의해 추상적으로 인식되는 연관성 타입(association type)으로 구성된다. 측면(aspect)은 엔티티 타입에 대한 한 뷰(view)를 나타내는 것으로 엔티티 타입에 대한 한 측면의 결과로 또다른 여러 엔티티 타입의 인식이 가능하다. 즉, MAOM 의 엔티티 타입들은 측면의 개념을 이용하여 상호간의 암시적 혹은 명시적 관계성을 갖는다. MAOM 의 엔티티 타입과 그 측면들은 EEA 도표 (Extended Entity -Aspect Diagram)를 이용하여 엔티티 타입들의 다측면 구조를 그림으로 나타낼 수 있다. EEA 도표의 주요 구성기호는 그림-2 와 같다.

구성 기호	의 미
	엔티티 객체 타입
	연관성 객체 타입
A  :	객체타입 A 의 측면 링크
 :	분해타입(D-type) 측면
 :	부분집합타입(S-type) 측면
 :	분할타입(P-type) 측면
 :	범주타입(C-type) 측면
 :	관계성타입(R-type) 측면

<그림-2> EEA 도표의 주요 구성기호

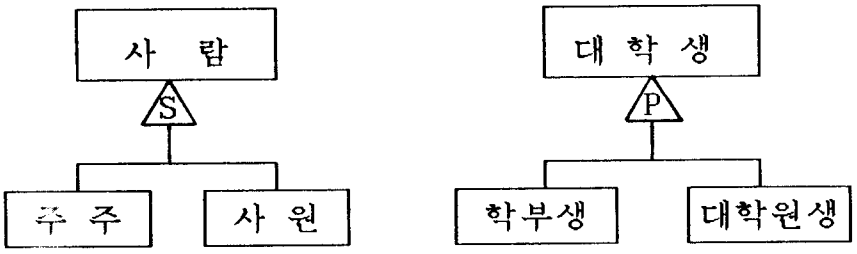
MAOM 에서 측면은 객체 타입을 바라보는 뷰의 특성에 따라 분해(decomposition),부분집합(subsetting),분할(partition), 범주(category), 관계성(relationship)의 5가지 타입으로 세분된다.

분해타입(D-type) 측면은 주어진 엔티티 타입이 서로 다른 여러 성분 엔티티 타입들의 합성으로 이해될 수 있음을 나타낸다. 한 엔티티 타입은 다수의 분해타입 측면을 가질수 있으며, 각 분해타입 측면은 분해측면 이름에 의해 구분된다.



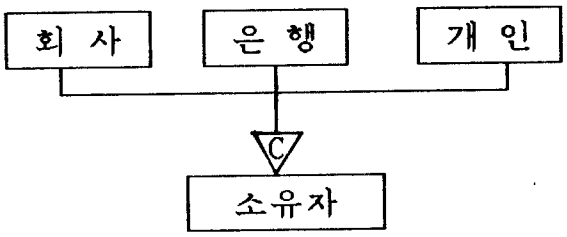
<그림 3> 분해타입(D-type) 측면

부분집합 타입(S-type) 측면은 주어진 엔티티 타입의 객체들이 여러 중복 가능한 엔티티 타입들로 관측될 수 있음을 나타내며, 분할타입 (P-type) 측면은 주어진 엔티티 타입이 다수의 상호 배제적인 엔티티 타입들로 분할되어 이해될 수 있음을 나타낸다. 분할타입 측면은 하위 타입들이 서로 상호 배제적이고 하위타입들은 상위 타입에 대해 완전해야 한다는 점에서 부분집합 타입과는 구별된다.



<그림 4> 부분집합 타입과 분할 타입 측면

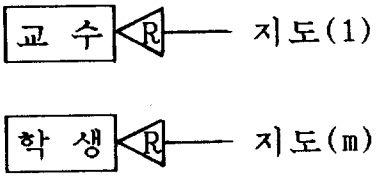
범주타입(C-type) 측면은 주어진 엔티티타입이 서로 다른 엔티티타입들의 합집합으로 표현됨을 나타내는 것으로 범주타입 측면을 갖는 엔티티 타입은 서로 배타적이고 독립적인 상위 엔티티 타입들에 의해 구성된다. 즉, 엔티티 타입은 범주타입 측면을 통해 하나 이상의 상위 타입들을 가질 수 있다. 예를 들면, 그림-5 에서 '소유자' 타입의 객체들은 반드시 '회사', '은행', '개인' 중 어느 한 타입의 객체임을 의미한다. 그러나 '회사'(혹은 '은행', '개인')타입의 모든 객체들이 '소유자'타입의 객체일 필요는 없다.



<그림 5> 범주타입(C-type) 측면

관계성 타입(R-type) 측면은 주어진 엔티티 타입이 다른 독립적인 엔티티 타입들과의 연관성을 갖는다는 것을 의미하는 것으로 주어진 엔티티 타입이 참여하는 연관성 타입을 암시한다. 그림-6은 ER(Entity-Relationship) 모델의 관계성 표현이 MAOM 에서는 관계성 타입 측면을 이용하여 표현될 수 있음을 보여준다.

< MAOM 모델 >



< ER 모델 >

<그림 6> MAOM 과 ER 모델의 비교 (관계성 표현)

III. MAOM 형식론의 구성

2 장에서 설명한 다측면 객체모델에 대한 형식론은 수학의 집합론과 함수론을 바탕으로 한다. MAOM의 형식론을 위한 기본 공리는 다음과 같다.

[식별공리] 모든 데이타 객체들은 모델 환경 전체에서 유일하게 식별될 수 있는 시간불변의 수단(UID: Universal IDentifier)을 갖는다.

[클래스공리] S(e)를 데이타 객체 e 에 대한 서맨틱(semantic)이라고 할 때, 임의의 시간 t 에서 S(e)를 만족하는 모든 데이타 객체들로 구성되는 클래스(class)가 존재하며, 한 클래스내의 모든 데이타 객체들은 그 클래스의 서맨틱하에서 유일하게 식별될 수 있는 수단(CID: Class IDentifier)을 갖는다.

[서맨틱변화공리] 데이타 객체들은 시간이 지남에 따라 서맨틱의 획득(생성), 변화, 상실(소멸)이 가능하다.

[null공리] null 은 항상 모든 클래스의 인스턴스인 객체이다.

클래스들을 보다 정확하게 언급하기 위해서는 모델이 존재하는 시간구간 (time interval)의 개념이 필요하게 된다. 이러한 시간구간 개념을 모델의

'시간베이스(timebase)' 라고 하고 $\mathbb{T}$ 로 나타낸다. MAOM 의 형식론을 위한 주요 정의들은 다음과 같으며, 이 이외의 수학 개념들은 [3]의 정의에 따른다.

정의-1: (클래스함수) C_1, C_2 를 모델 클래스라고 하자. 그러면, 대응 $F:C_1 \rightarrow C_2$ 가 set-valued 함수로서 다음을 만족할때, F 를 '클래스함수 (class function)'이라고 한다.

1) 임의의 시간 $t \in \mathbb{T}$ 에서, $F=F_t:\langle C_1,t\rangle \rightarrow \langle C_2,t\rangle$ 은 $F(null)=\{NULL\}$ 이고 null이 아닌 임의의 $e \in \langle C_1,t\rangle$ 에 대해 $F(e)=\{null\}$ 이거나 $null \notin F(e)$ 이다.
2) 임의의 시간 $t \in \mathbb{T}$ 에서 C_2 의 인스턴스 e 가 C_2 에서 소멸되면, e 를 포함하는 $F_t(x)$ ($x \in \langle C_1,t\rangle$) $\#(F_t(x))=1$ 이면 {null}, $\#(F_t(x))\neq 1$ 이면 $F_{t'}(x)-\{e\}$ 로 된다. (여기서 t' 은 e 가 소멸되기 직전의 모델시간 이다.)

정의-2: (ID 존재결정 함수) C_1, C_2 가 모델 클래스이고, 클래스 함수 $F:C_1 \rightarrow C_2$ 가 injective 일때, F 를 'C_1 에 의한 C_2 의 ID 존재결정함수 (ID existence determinant function)' 라고 한다.

정의-3: (클래스타입) C 를 모델 클래스라고 하자. 어떤 모델클래스들의 집합 $\{C_1,\ldots,C_n\}$ 이 존재해서 surjective ID 존재결정함수

$F: \coprod_{i=1}^{n} C_i \rightarrow C$ 이 정의된다면, C 를 $\{C_1,\ldots,C_n\}$ 에 의한 연관성 (association) 타입 클래스'라고 한다. 그리고 이러한 클래스들이 존재하지 않을 때, C를 '엔티티(entity) 타입 클래스' 라고 한다.

위의 정의-3 에서 C 가 $C_1,\ldots,C_n$ 에 의한 연관성 타입 클래스일 때, $C_1,\ldots,C_n$ 을 'C 의 참가자(participant)' 라고 한다.

정의-4: (ID 대응함수) C_1, C_2 가 모델클래스이고, $F:C_1 \rightarrow C_2$ 가 다음을 만족할 때, F 를 'C_1 에서 C_2 로의 ID 대응함수 (ID correspondence function)'라고 한다.

1) F 는 injective single-valued ID 존재결정함수
2) $ID(F(e)) = ID(e)$, for $\forall e \in C_1$

정의-5: (합집합객체/원소객체) C_1, C_2 가 모델 클래스이고, 클래스 함수 $F:C_1 \rightarrow C_2$ 가 다음을 만족한다고 하자.

1) F 는 surjective ID 존재결정함수이고,
2) $\forall e \in C_1$ 에 대해, $F(e)=\{e_1,\ldots,e_n\}\neq\{null\}$

그러면, e 를 $\{e_1,..,e_n\}$ 의 F 하의 '합집합객체(union object)'라고 하고,

$ID(e) \equiv ID(\{e_1,\ldots,e_n\})$으로 정의한다. 그리고 e_i (i=1,..., n) 를 e 의 '원소객체(element object)' 라고한다.

위의 정의들을 토대로 MAOM의 엔티티 객체타입의 측면과 그 측면의 타입은 다음과 같이 정의된다.

정의-6: (측면함수) C, $C_1,\ldots,C_n$ 이 모델 클래스이고, 특별히 C 는 엔티티 클래스라고 하자. $A:C \rightarrow \coprod_{i=1}^{n} C_i$ 가 다음 두 조건중 어느 하나를 만족하는 클래스함수라고 할 때, A 를 'C 의 측면함수(aspect function)' 라고 한다.

1) A 가 surjective ID 존재결정함수이다.
2) A 가 ID 대응함수이다.

정의-7: (측면함수의 타입 구분) $A: C \rightarrow \coprod_{i=1}^{n} C_i$ 가 측면함수라고 하자. 그러면,

1) 특별히, $\bigcup_{i=1}^{n} C_i = R$ (R 은 C 를 참가자로 갖는 연관성 클래스) 이고, A 가 surjective non-ID 대응함수일 때, A 를 '관계성타입(R-type) 측면함수'라고 한다.
2) $C_1,..,C_n$ 이 엔티티클래스이고 $ID(C) \supseteq \bigcup_{i=1}^{n} ID(C_i)$이며 A 가 surjective ID 대응함수 일때, A 를 '부분집합타입(S-type) 측면함수'라고 한다.
3) $C_1,\ldots,C_n$ 이 상호 disjoint 엔티티 클래스들이고 $ID(C) = \bigcup_{i=1}^{n} ID(C_i)$ 이며 A 가 surjective ID 대응함수일 때, A 를 '분할타입(P-type) 측면함수'라고 한다.
4) $C_1,\ldots,C_n$ 이 상호 disjoint 엔티티 클래스들이고 $ID(C) \subseteq \bigcup_{i=1}^{n} ID(C_i)$ 이며 A 가 ID 대응함수일 때, A 를 '범주타입 (C-type) 측면함수'라고 한다.
5) $C_1,\ldots,C_n$ 이 상호 disjoint 엔티티 클래스들이고, A 가 surjective non-ID 대응함수일 때, A 를 '분해타입(D-type) 측면함수' 라고 한다.

위의 정의-6에서 측면은 엔티티클래스 위에서만 정의됨에 주의해야 한다. 따라서, MAOM 객체 클래스의 스키마 구조는 다음과 같이 정의된다.

정의-8: C 를 MAOM 의 한 클래스라고 하자. 그러면 C 의 구조는

$$C = \langle CLASS\_NAME, CLASS\_SCHEMA, INSTANCES, \mathbb{T} \rangle$$

로 정의된다. 여기서, CLASS_NAME은 클래스 이름이고, CLASS_SCHEMA 은 클래스의 서맨틱이며, INSTANCES은 인스턴스들의 집합이다. C가 엔티티 타입일때와 연관성 타입일때 각각의 CLASS_SCHEMA는 다음과 같은 구조를 갖는다.

엔티티 타입: CLASS-SCHEMA=< Atts(C), Aspects(C) >
연관성 타입: CLASS-SCHEMA=< Atts(C), Pats(C) >

여기서 Atts(C) 는 C 의 속성들의 집합이고 Aspects(C))는 C 위에서 정의되는 측면들의 리스트이다. 그리고 Pats(C)는 C 의 참가자 리스트이다.

위의 정의-8에서 C=<CLASS_NAME,CLASS_SCHEMA> 를 '클래스 타입(class type)' 혹은 '클래스 내연(class intension)' 이라고 하고 INSTANCES(C) 를 '클래스 외연(extension)'이라고 한다. 그리고 클래스 타입들의 모임을 '모델 스키마(model schema)'라고 한다.

이상의 개념과 정의들을 토대로 해서 MAOM 의 구조는 다음과 같이 형식화된다.

정의-9: (MAMO 의 형식론) 다측면 객체 모델의 구조는 다음과 같다.

$$MAOM = \langle\langle \mathbb{E}, \mathbb{R}\rangle, \mathbb{A}, \mathbb{T}\rangle$$

여기서 $\mathbb{E} = \{ E \mid E=\langle Atts(E), Aspects(E)\rangle \}$

$\mathbb{R} = \{ R \mid R=\langle Atts(R), Pats(R)\rangle \}$

$\mathbb{A} = \{ A \mid A: C \rightarrow \bigcup_{i=1}^{n} C_i$:측면함수, $E \in \mathbb{E}$, $C \in \mathbb{E} \cup \mathbb{R} \}$

$\mathbb{T}$ = 모델 시간베이스

IV. MAOM 의 설계방법론

MAOM 은 객체타입 노드들로 구성되며, 객체타입 노드들은 측면을 나타내는 링크를 통해 다른 객체 타입 노드들과 관련된다. 객체 타입은 독립적으로 인식이 가능한 실세계의 대상체를 의미하고, 측면은 객체 타입 노드에 대한 한 뷰를 나타낸다. 객체 타입 노드에 대한 한 측면의 적용 결과는 새로운 여러 객체 타입 노드들을 생성한다. MAOM은 다음과 같이 단계적 접근 방법으로 설계한다.

단계-1 : 시스템 노드(system node)를 생성한다. 시스템 노드는 단지 분해타입 측면만을 가질 수 있다.

단계-2 : 모델링의 목적에 부합되는 주요 엔티티타입 노드들을 선정한다. 이러한 노드들은 시스템 노드의 구성측면으로 나타난다.

단계-3 : 단계-2 에서 선정된 각 엔티티 타입 노드에 대해, 필요하다면, 분해타입 및 특수화 타입 측면들을 고려한다.

단계-4 : 단계-3 까지의 선정된 엔티티 타입 노드들로부터 범주 타입의 측면을 갖는 새로운 엔티티 타입 노드를 생성한다.

단계-5 : 단계-5까지에서 선정된 엔티티타입 노드들간의 관계성을 토대로 연관성 타입을 선정한다. 엔티티타입들 사이의 포함 혹은 존재의존성이 있는 경우, 즉 한 엔티티 타입의 측면으로 다른 엔티티타입이 생성되는 경우 이들이 함께 참가하는 연관성 타입은 정의될 수 없다.

단계-6 : 단계-6 에서 선정된 연관성 타입들을 토대로, 각 엔티티 타입에 대해 관계성 측면을 추가한다.

단계-7 : 모델의 재구조화 과정을 통하여 모델의 구조를 단순화한다. 단계-6까지의 과정에서 엔티티타입에는 많은 측면이 부여될수 있는데, 이것은 측면의 상속성, 통합성 등의 성질을 이용하여 모델 서맨틱의 변경없이 모델 구조의 단순화가 가능하다.

V. 결 론

본 논문에서는 개념적 데이타 모델인 MAOM 을 제안, 설명하고, 수학적 접근을 토대로 형식론을 구성하였으며, 이러한 모델의 설계를 위한 단계적 접근 방법론을 제시하였다.

MAOM 은 다측면의 실세계 객체를 직접 모델의 다측면 데이타 객체로 표현할 수 있도록 함으로써 개념적 데이타베이스 스키마의 설계를 보다 용이하게 한다. 그리고 MAOM 에 대한 형식론의 구성은 MAOM 에 대한 이론 연구의 기초로 이용될 수 있다.

MAOM 은 개념적 데이타 스키마의 설계는 물론 특히 구성요소 스키마들의 통합에 강력한 도구로 이용될 수 있을것으로 기대된다. 추후, 스키마통합을 위한 MAOM 의 측면 관리에 대한 연구가 필요하다.

< 참고문헌 >

[1] Bernard P. Zeigler, *Multifacetted Modeling and Discrete Event Simulation*, Academic Press, 1984.

[2] C. Batini and Maurizio Lenzerini, "A Methodology for Data Schema Integration in the Entity Rerlationship Model", IEEE Transaction on Software Engineering , Vol. SE-10, No. 6, Nov. 1984, pp 650-644.

[3] Charles C. Pinter, *Set Theory*, Addison-Wesleey Series in Mathematics, 1971.

[4] Dionysios C. Tsichritzis and Frederick H. Lochovsky, *Data Models*, Prentice-Hall, Inc., 1982.

[5] James RumBaugh, Michael Blaha, William Premerlani, Frederick Eddy,

and William Lorensen, *Object-Oriented Modeling and Design*, Prentice-hall, 1991.
[6] Ramez Elmasri and Shamkant B. Navathe, *Fundamentals of Database Systems*, the Benjamin/cummings Publishing Company, Inc., 1989.
[7] S. Abiteboul, P. C. Fischer, and H.-J. Schek(Eds.), *Nested Relations and Complex Objects in Databases*, Lecture Notes in Computer Science 361, Springer-Verlag, 1989.
[8] S. B. Navathe and S. G. Gadgil, "A Methodology for View Integration in Logical Database Design", Proceedings of the 8th International Conference on Very Large Data Bases(Mexico city), Sept. 1982, pp 142-152.
[9] Shamkant Navathe, Ramez Elmasri, and James Larson, "Integrating User Views in Database Design", IEEE Computer, Vol.19, No. 1, Jan. 1986, pp 50-62.
[10] Won Kim, Introduction to Object- Oriented Databases, The MIT Press, 1990.
[11] 김성훈, "E-A 모델을 이용한 데이타베이스 스키마 통합 방법론에 관한 연구", 고려대학교, 석사학위 논문, 1988.
[12] 김창화, "EA 모델에 의한 지식표현 모델링", 고려대학교, 박사학위 논문, 1989.
[13] 이석호, 황수찬, "객체중심 데이타베이스에서 추상화 개념을 기본으로 한 관계성 및 제약조건의 모델링", 한국정보과학회 데이타베이스 연구회지, 6권, 1호, 1990. 4.