# 이질의 분산 데이타베이스에서 객체 통합을 위한 검증 모델

김 용 원†

**요 약**

분산된 지역 데이타베이스의 스키마 통합시 중심 개념은 엔티티 통합을 의미하며 이때 엔티티의 시멘틱은 여러 요소에 영향을 받는다. 따라서 분산된 데이타베이스 환경에서의 스키마 통합은 통합되어야 하는 각 지역 스키마의 엔티티 타입간의 도메인 관계의 정의로 부터 시작된다할 수 있다. 이렇게 설계자에 의해 정의되는 각 엔티티간의 도메인 관계는 스키마 통합 시스템의 신뢰도의 확보와 검증을 위한 작업이 반드시 수행되어야 한다. 그러나 이러한 작업이 기존의 통합 시스템에서는 제시되지 못하였다. 본 논문은 분산 시스템 환경에서 지역 데이타베이스의 스키마 통합을 위한 객체 중심의 통합을 정의하고, 통합의 타당성 검증을 위한 모델을 제안하고 검증 시스템을 구현한다.

# Verification Model for Object Integration in Heterogeneous Distributed Database

Yoeng Won Kim†

**ABSTRACT**

When we integrate schema of distributed local databases, we mean entity integration as central concept of schema integration, and semantic of entity affects several factors. Thus the schema integration in distributed database environment starts from definition of domain relation among entity types of local schemas. Moreover, the domain relation defined by designer needs some works for confidence and validation of schema integration system. But this work was not presented in previous integration system. In this paper, we define object oriented integration for schema integration of local databases in distributed system environment, present models to verify validation on its integration, and implement the validation system.

## 1. 서 론

분산 데이타베이스는 데이타베이스의 분산과 통합의 개념으로 정의할 수 있다. 분산된 데이타들은 상호 관계되고 시스템들 사이에서의 상호 접근 허용하며 사용자에게는 시스템의 투명성을 제공하는 논리적인 데이타베이스로 존재한다[1, 11]. 독립적으로 구성된 지역의 데이타베이스는 논리적으로 통합되고 각 데이타베이스는 사용자에게 통합된 뷰로서 제공되어야 한다[2, 14, 15]. 즉 사용자는 지역 데이타베이스에 분산되어 있는 관심있는 데이타들이 명세되는 통합된 뷰를 요구한다.

기존에 존재하는 데이타베이스들에 대한 분산 데이타베이스의 성공적인 구축은 자연스럽게 시멘틱의 표현 능력을 지원할 수 있는 공통 모델의 연구와 개발에 의존하고 있다[3, 4]. 이러한 모델의 연구와 개발은 계속되어 왔으며 최근에는 개념 모델을 측면의 개념을 이용하여 단일 측면의 모델과 다중 측면의 모델로 분류하여 정의하고 있다[5, 6]. 여기서 측면(aspect)이라하면 실세계에 존재하는 하나의 객체에 대하여 이 객체를 바라보는 관점이 다르므로 해서 발생하는 객

† 정 회 원 : 고려대학교 전산과학과 박사과정
논문접수 : 1994년 10월 5일, 심사완료 : 1995년 1월 3일

체의 다양한 의미 표현을 제공하고자하는 개념이다[5, 6, 16]. 이러한 모델을 이용해서 정의되는 통합 스키마는 설계자와의 상호 대화를 통해 이루어지는 통합 도구에 의해 설계되며, 이때 통합되는 엔티티 타입은 그 통합 타당성이 검증되어야 한다. 일반적으로 엔티티 타입의 통합은 이들의 도메인 관계에 의해 정의되고 이를 기반으로 통합된다. 따라서 본 논문에서는 기존 모델의 다중 측면에 의한 엔티티 타입 통합[7, 16]에 기반하여 엔티티 타입의 도메인별 관계에 따른 통합을 검증하는 알고리즘에 관하여 논의하고 앞으로의 연구 방향에 대하여 제안하고자 한다.

## 2. 다중 측면 기반 모델

분산 환경에서 의미적, 구조적으로 지역 스키마의 다양성을 지원하기 위한 중요한 과제 중의 하나가 데이타 모델의 개발과 연구이다. 기존의 모델[3, 7, 8]은 분산 환경의 실세계와 시멘틱의 차이를 내포하고 있으며 시멘틱의 손실과 표현의 중복을 초래한다. 개념 모델은 정적인 구조를 이용하여 실세계의 다양한 시멘틱을 제공하고, 개념적 특성인 측면에 따라 단일 측면과 다중 측면으로 분류하고 있다[6]. EA 모델[5]과 MAO 모델[6]은 다중 측면 모델이며 본 논문은 이러한 모델을 기반으로 도메인 관계에 따른 객체 통합을 정의한다. EA(Entity−Aspect) 모델은 측면 개념을 이용하여 실세계의 다양한 관점을 표현하려는 지식표현 모델이다. 이 모델은 유형, 무형의 객체와 여러 관점 그리고 관계를 각각 객체 타입 노드와 측면 타입 노드, 그리고 링크로 표현한다. 그리고 속성 타입으로 구성된다. MAO (Multiple Aspect−based Object)모델은 동일한 객체라할지라도 모델을 설계하는 설계자의 관점에 따라 실세계의 객체를 다양하게 표현할 수 있다는 모델이다. 이 모델은 EA모델을 확장하여 쉽고 정확하게 시멘틱을 표현하고자하는 모델이다.

다중 측면 모델에서 측면은 실세계(분산되어 있는 기존의 데이타베이스 환경)의 객체에 대한 다양한 시멘틱을 제공하고자 하는 개념으로 정의한 바 있다[6, 16]. 그러므로 통합된 데이타베이스 모델에서는 지역의 스키마에서 다른 관점으로 표현된 동일한 객체의 시멘틱을 측면 타입을 이용하여 쉽게 모델링한다. 또한 기존의 수직 관계 구조를 측면 관계를 통하여 세부적이고 명시적으로 지원하고 있다.

본 논문에서 기반하고 있는 다중 측면 모델은 기존의 측면 모델 [5, 6, 16]에 근거하여 ER모델을 측면의 수직적 구조로 세분하여 확장하는 모델로 정의한다. 그 정의는 본 절에서 다음과 같이 기술한다. 실세계의 모든 유형과 무형의 객체 집합을 T라 하면 이 집합의 원소들은 각각의 이름을 가지며 독립된 형태의 시멘틱을 정의한다. 이때 데이타베이스 모델은 엔티티 타입, 측면 타입, 그리고 관계성 타입을 포함하는 T의 부분 집합으로 정의된다.

[정의 2.1] 데이타베이스 모델

T : 실세계의 모든 형태의 객체 집합,
M : 데이타베이스 모델이라 하자.
$t \in T \Leftrightarrow t = \langle t\_type, t\ name, t\_schema \rangle$
, t_type: 엔티티 타입, 측면 타입, 또는 관계성 타입
t_name: 객체 타입의 이름
t_schema: 객체 타입 (t type)이 갖는 시멘틱
$M \subset T,\ M = \bigcup_{i}^{n} t_i (t_i \in T)$

[정의 2.2] 측면 구조 및 타입 정의

ASPECT을 측면구조들의 집합이라 하자.
$ASPECT \subset T,$
$\alpha \in ASPECT \Leftrightarrow \alpha$ schema = ⟨ Aspect name, OWNER, MEM, Aspect type⟩,
Aspect name : 측면 $\alpha$의 이름
OWNER: $\alpha$를 측면으로하는 엔티티 타입들의 집합
MEM: $\alpha$에 의해 OWNER와 관련되는 엔티티

타입들의 집합

Aspect_type: OWNER와 MEM 사이의 측면 타입(D, S, P, C)

$E_1, E_2 \in MEM(\alpha)$, $E \in OWNER(\alpha)$, Dom(E): 엔티티 타입 E의 도메인이라 하자.

1) Aspect_type = 'D'

$Dom(E) = Dom(E_1) \circ Dom(E_2)$ & $Dom(E_1) \cap Dom(E_2) = \emptyset$

즉, D 타입의 측면을 갖는 OWNER 엔티티는 $E_1$, $E_2$라는 상호 배제적인 엔 티티 타입으로 합성되어 생성됨으로써 정의된다.

2) Aspect_type = 'S'

$Dom(E) \supset Dom(E_1) \cup Dom(E_2)$ & $Dom(E_1) \cap Dom(E_2) \neq \emptyset$

즉, S 타입의 측면을 갖는 OWNER 엔티티 타입은 $E_1$, $E_2$의 합을 포함하고 OWNER($\alpha$)에 속하는 엔티티 타입의 인스턴스는 하나 이상의 하위 엔티티 타입에 속할 수 있다.

3) Aspect_type = 'P'

$Dom(E) \supset Dom(E_1) \cup Dom(E_2)$ & $Dom(E_1) \cap Dom(E_2) = \emptyset$

즉, P 타입의 측면을 갖는 OWNER 엔티티 타입은 상호 배제적인 엔티티 타입으로 분할되어 인식되며, OWNER($\alpha$)의 인스턴스는 반드시 하나의 하위 엔티티 타입에 속해야 한다.

4) Aspect_type = 'C'

$\alpha, \beta, \gamma \in ASPECT$, $E_i, E_j \in MEM(\alpha)$ 또는 $MEM(\beta)$, $1 \le i, j \le m$

이때, $Dom(E_i) \cap \ldots \cap Dom(E_j) = \emptyset$,

$\exists E^k \in OWNER(\gamma)$

s. t. $\gamma \in ASPECT$, $Dom(E_i) \cap \ldots \cap Dom(E_j) \neq \emptyset$, $1 \le k \le n$.

즉, MEM에 속하는 엔티티 타입들은 'C' 타입의 측면으로 범주화될 수 있으며, MEM에 속 하는 엔티티 타입은 'C' 타입의 여러 OWNER 엔티티 타입 ($E^k$)을 갖을 수 있다. 역으로 'C' 타입 측면을 갖는 엔티티 타입은 상호 배제적인 엔티티 타입으로 분할될 수 있음을 의미하며, 'C'타입의 하위 엔티티 타입이 OWNER 엔티티 타입에 반드시 포함될 필요는 없다.

[예제 2.1] 실세계의 학생 객체는 측면에 따라 시멘틱이 부여될 수 있으며, 자격에 따라 학부생과 대학원생이라는 'P' 타입의 측면 구조로 정의된다. 이를 형식화하면,

Aspect_type = 'P', $\alpha \in ASPECT$라 하면, $\alpha$_schema = 〈자격, '학생', {'학부생', '대학원생'},'P'〉로 정의된다.

측면 타입으로 정의되는 엔티티 타입의 시멘틱은 측면의 이름과 그 타입으로 나타나며, 이는 엔티티 타입의 도메인을 정의하여 도메인들 사이의 관계를 정의하는데 중요한 요소가 된다.

[정의 2.3] 엔티티 타입 정의

엔티티 타입 $E \in T$의 시멘틱 E_schema는 다음과 같다.

E_schema = 〈Attr(E), Aspect(E)〉

, Attr(E): 엔티티 타입 E의 속성 집합

$Aspect(E) = \{\alpha \mid \alpha \in ASPECT \& E \in OWNER(\alpha)\}$

Entity_type(E): 엔티티 타입의 집합이라 하면, $Entity_type(E) \subset T$이다.

한편, 엔티티 타입들 사이의 시멘틱을 유지하는 관계성 타입은 정의 2.4에서 형식화하여 정의한다.

[정의 2.4] 관계성 타입 정의

R: 엔티티 타입들 간의 관계 집합이다 ($R \subset T$).

$r \in R \Leftrightarrow$ r_schema = 〈Attr(r), r_name, $E_1$_name(n[,$E_1$_role]), $E_2$_name(m[,$E_2$_role])〉

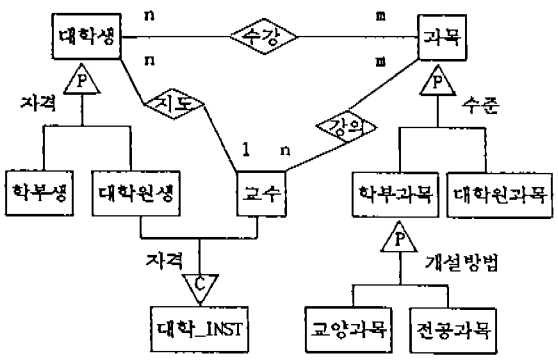
, Attr(r): 관계성 타입의 속성,

r name: 관계성 타입의 이름
$E_i$ name: 엔티티 타입 $E_i$의 이름, $i \in \{1,2\}$
$E_i$ role: 엔티티 타입 $E_i$의 역할을 말하며 생략될 수도 있다.
n, m: 관계성 타입의 카디널리티

이때, r은 엔티티 타입간의 관계를 정의한다. 통합모델에서는 지역 엔티티 타입간의 새로운 관계를 정의할 수 있으며, 지역 엔티티 타입은 통합된 모델의 엔티티 타입으로서 상호 의존적 시멘틱(mutually dependent semantic)을 갖는 수평적 관계로 모델링될 수 있다.

[정의 2.5] 관계 상속

$\alpha \in$ ASPECT & t $\in$ {'P','S'},
$E_1$-schema=〈 Attr($E_1$), Aspect($E_1$) 〉,
$E_2$-schema=〈 Attr($E_2$), Aspect($E_2$) 〉,
Aspect($E_1$)=〈 Aspect_name, $E_1$, {$E_{1a}$,...,$E_{1b}$}, Aspect_type 〉, $1 \le a \le b \le n$
Aspect(E2)=〈 Aspect_name, $E_2$, {$E_{2a}$,...,$E_{2b}$}, t 〉라 하자.
이때, 엔티티 타입 $E_1$, $E_2$ 사이의 관계성 타입 r 을
〈 Attr(r), r_name, $E_1$-name(m), $E_2$-name(n) 〉이라 하면,
관계성 타입 r은 〈 Attr(r), r_name, $E_1$-name(m), $E_2$-name(n) 〉
(단, $E_i \in \{E_{2a}, \ldots, E_{2b}\}$) 으로




(그림 1) 다중 측면 모델에 의한 모델링
(Fig. 1) Modelling by multiple aspects model

재정의된다. 즉, r이 하위 엔티티 타입과의 관계로 상속된다.

[예제 2.2] '대학생' 엔티티 타입은 'P' 타입 측면을 가지며 하위의 엔티티 타입으로 '학부생'과 '대학원생'을 갖는다. 한편, '교수' 엔티티 타입은 '과목' 엔티티 타입과의 사이에 '강의'라고 하는 관계성이 정의되어 있다. 이때 '대학생' 엔티티 타입과 '교수' 엔티티 타입 사이에는 '지도' 관계성 타입이 정의된다. (그림 1)은 이 관계를 앞서 정의한 다중 측면 모델로 모델링하고 있다.

## 3. 스키마의 엔티티 타입 도메인 관계

엔티티 타입 통합의 개념은 스키마 통합의 중심 개념이며, 통합 모델의 구축을 위하여 관계성 타입과 연관되어 통합된다. 이 엔티티 타입 통합은 다음과 같이 정의하는 엔티티 타입들의 도메인에 근거 한다. 다음의 도메인 관계 정의는 2장에서 형식화하여 정의한 다중 측면 모델에 근거하여 형식화하였으며, 이에따라 도메인 관계에 따른 통합 방법을 정리화하여 이를 증명하였다.

첫째, 동일한 도메인 사이의 관계는 두 엔티티 타입이 갖는 속성과 측면 타입의 관계에 의해 다음과 같이 형식화된다.

[정의 3.1] 동등 관계 도메인(I)

$\alpha, \beta \in$ ASPECT, $E_1$, $E_2 \in$ Entity_type(E)에 대하여,
Attr($E_1$)={$a_1$, ... ,$a_n$}, Attr($E_2$)={$b_1$, ... ,$b_m$}를 $E_1$, $E_2$ 각각의 속성 집합이라 하자.
Aspect($E_1$)=〈Aspect_name, $E_1$, MEM, Aspect_type〉,
Aspect($E_2$)=〈Aspect_name, $E_2$, MEM, Aspect_type〉라 하자.
I($E_1$,$E_2$) 는 동등 관계
s.t. {$a_1$ ... ,$a_n$} $\subseteq$ {$b_1$, ... ,$b_m$} 그리고
Aspect_name($\alpha$) = Aspect_name($\beta$)
& MEM($E_1$) $\subseteq$ MEM($E_2$)

둘째, 엔티티 타입의 도메인 관계가 포함관계를 갖는 경우를 정의할 수 있다. 하나의 엔티티 타입 도메인이 다른 엔티티 타입의 도메인에 포함됨으로서 엔티티 타입이 상/하위 클래스 관계를 갖는다. 이때의 속성 집합은 포함관계를 이루고, 'S' 또는 'P' 타입의 측면관계로 통합된다.

[정의 3.2] 포함 관계 도메인(C)

$\beta \in$ ASPECT, $E_1$, $E_2 \in$ Entity_type(E)라 하자.

$Attr(E_1)=\{a_1, \ldots, a_n\}$, $Attr(E_2)=\{b_1, \ldots, b_m\}$를 $E_1$, $E_2$ 각각의 속성 집합이라 하자.

$\beta=\langle$Aspect_name, $E_1$, $E_{1b}$, S or P$\rangle$라 하자.

$C(E_1, E_2)$는 포함 관계

s.t. $\{a_1, \ldots, a_n\} \subseteq \{b_1, \ldots, b_m\}$ 또는

$E_2 \in \{ E_{1b} \mid E_{1b} \in MEM(\beta) \}$

그때 $E_1$은 $E_2$의 상위클래스이고 일반화 관계이다.

[정리 3.1] $E_1$, $E_2 \in$ Entity_type(E)에 대하여, $(E_1,E_2)$ C라 정의하자.

$\alpha \in$ ASPECT에 대하여, 만일 $\alpha=\langle$Aspect_name, $E_1$,$E_2$, Aspect_type$\rangle$이면,

그때 Aspect_type($\alpha$)='S' 또는 'P'.

〈증명〉

가정에 의해, $(E_1,E_2)$ C, 그때 $Attr(E_1) \subset Attr(E_2)$.

만일 Aspect_type($\alpha$) 'S'이고 Aspect_type($\alpha$) 'P'이라 하면,

Aspect_type($\alpha$)='C' 또는 'D'이다.

① Aspect_type($\alpha$)='C' 이면,
측면 타입,'C' 의 정의에 의해, $E_1 \subset E_2$ 이다.

② Aspect_type($\alpha$)='D' 이면,
측면 타입,'D'의 정의에 의해, $E_1$이 $MEM(E_1)$의 합성으로 정의되므로, 그때, $E_1$ $E_2$ 를 만족하지 않는다.

이에따라, 가정, $Attr(E_1) \subset Attr(E_2)$, 에 모순 Aspect_type($\alpha$)='C' 또는 'D' 은 거짓이다.

그러므로 Aspect_type($\alpha$)='S' 또는 'P'.

셋째, 두 엔티티 타입의 도메인 관계가 공통부분의 도메인을 갖는 경우이다. 엔티티 타입들은 각각이 공통된 성질을 갖는 속성이 있으며, 이는 범주화라는 추상화를 통하여 정의된다. 즉, 'C' 타입의 측면 구조를 가지며 통합된다.

[정의 3.3] 중복 관계 도메인(O)

$\alpha \in$ ASPECT, $E_1$, $E_2 \in$ Entity_type(E)에 대하여,

$Attr(E_1)=\{a_1, \ldots, a_n\}$, $Attr(E_2)=\{b_1, \ldots, b_m\}$을 속성 집합이라 하자.

$O(E_1,E_2)$는 다음을 만족하고, E는 $E_1$, $E_2$의 상위클래스, 범주화.

① $\exists a_i \in Attr(E_1)$ & $b_j$ $Attr(E_2)$
s.t. $\{a_1, \ldots, a_n\} \cap \{b_1, \ldots, b_m\} \neq \emptyset$
& $a_i$ $Attr(E_2)$, $b_j$ $Attr(E_1)$.

② $\exists E \in OWNER(\alpha)$
s.t. $\alpha=\langle$Aspect_name, E, $\{E_1,E_2\}$, 'C' $\rangle$.

[정리 3.2] 'C' 타입에 의한 엔티티 타입 통합

$E_i \in$ Entity_type(E), $i \in \{1,\ldots,n\}$라 하자.

$O(E_1,\ldots,E_2)$, $\alpha \in$ ASPECT에 대하여,

만이 $\alpha=\langle$ Aspect_name, $E_1$, $E_2$, Aspect_type $\rangle$이면,

그때, $\forall E_i$ $\{E_1,\ldots,E_n\}$에 대하여, $\exists E_i$ s.t. $E_i \not\subset E$.

〈증명〉

$O(E_1,\ldots,E_n)$, Aspect_type($\alpha$)='C'이므로,

측면 타입'C'의 정의에 따라, 'C' 타입의 MEM에 포함되는 $E_i$은 OWNER($\alpha$)의 부분집합일 필요 없다.

이로써, $Ins(E_i)$, INS(E)를 각각 $Mem(E_i)$, Owner(E)의 인스턴스라 하면, $Ins(E_i) \in Mem(E_i)$일때, $Ins(E_i) \notin INS(E)$이다.

그러므로 $\exists E_i \in MEM$ s.t. $E_i \not\subset E$.

넷째, 지역에 존재하는 데이타베이스의 각 엔티티 타입들이 공통된 속성을 갖지 않고 상위의

새로운 도메인으로 정의되어 표현되는 경우이다. 각 엔티티 타입 도메인의 합(성)집합으로 상위 도메인이 정의되며 'D' 또는 'P' 타입의 측면으로 통합된다.

[정의 3.4] 이종 관계 도메인(D)

$\alpha \in$ ASPECT, $E_1$, $E_2 \in$ Entity_type(E)에 대하여, Attr($E_1$) = $\{a_1, \ldots, a_n\}$, Attr($E_2$)= $\{b_1, \ldots, b_m\}$를 속성 집합이라 하자.

Owner(E)를 E의 OWNER라 하자.

그때 D($E_1$,$E_2$) 은 이종 관계이고 다음을 만족한다.

$\exists E \in$ OWNER s.t. $E_1$, $E_2 \in$ MEM & $\{a_1, \ldots, a_n\} \cap \{b_1, \ldots, b_m\} = \emptyset$.

[정리 3.3] $\alpha \in$ ASPECT에 대하여,

만일 $E_1$, $E_2 \in$ Entity_type(E), D($E_1$,$E_2$), 그리고 $\alpha$=〈Aspect_name, E, {$E_1$,$E_2$}, Aspect_type〉이면 그때 Aspect_type($\alpha$)='P' 또는 Aspect_type($\alpha$)='D'이다.

〈증명〉

Ins($E_i$)를 Mem($E_i$)의 인스턴스, INS(E)를 Owner(E)의 인스턴스, 그리고 Dom(E)를 엔티티 타입, $E_i$의 도메인이라 하자.

1) 만일 Aspect_type($\alpha$)='S'이면, 그때 $E_1$, $E_2 \in$ MEM($\alpha$)은 다음 관계를 갖는다.

① Dom($E_1$) ∪ Dom($E_2$)⊂Dom(E), Dom($E_1$) ∪ Dom($E2_2$)=Dom(E), not necessarily.

② Ins(E) ∈ INS($E_i$), $1 \le i \le 2$.

① 과 ② 는 가정, ($E_1$,$E_2$) D, 에 모순.

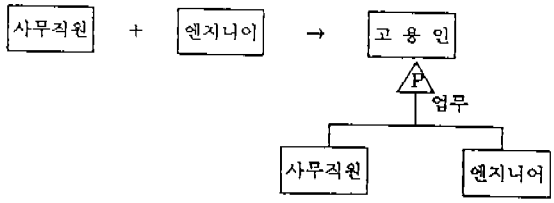
2) 만일 Aspect_type($\alpha$)='C'이면, 엔티티 타입은 정의3.3에 의해 중복 관계를 가져야 하므로, 이는 모순이다.

그러므로, 1), 2)에 의해,

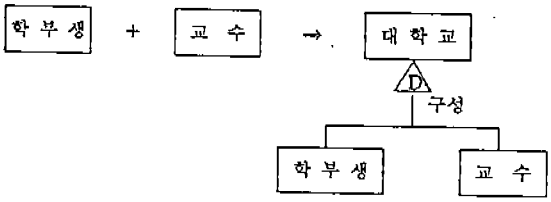
Aspect_type($\alpha$)='P' 또는 Aspect_type($\alpha$) ='D'.

[예제 3.1] 측면에 따른 정리 3.3의 예

① 사무직원, 엔지니어 ∈ Entity_type(E)에 대하여 $\alpha$=〈 업무, 고용인, {사무직원, 엔지니어}, 'P' 〉 이다.




② 학부생, 교수 ∈ Entity_type(E)에 대하여 $\alpha$=〈 구성, 대학교, {학부생,교수}, 'D' 〉 이다.




## 4. 통합을 위한 검증 알고리즘

3절에서는 다중 측면 모델을 이용하여 표현된 엔티티 타입들 사이의 도메인 관계를 정의하였다. 스키마 통합 과정에서 이들 사이의 관계를 정의하는 작업은 통합 스키마의 신뢰성 확보를 위하여 매우 중요한 작업이며 이는 통합 시스템에 의해 검증되어야 한다. 따라서 이 절에서는 이행적 규칙을 이용하여 엔티티 타입의 도메인간에 정의되는 관계를 검증하기 위한 모델을 예와 함께 알고리즘을 제안한다.

통합되어야 하는 엔티티 타입들 사이에는 3절

〈표 1〉 이행적 규칙 테이블(TR)
〈Table 1〉 Transitive Rule Table(TR)

| | I | CI | C | D | O |
|---|---|---|---|---|---|
| I | I | CI | C | D | O |
| CI | CI | CI | I,CI,C,D,O | D | CI,D,O |
| C | C | I,CI,C,O | C | C,D,O | C,O |
| D | D | CI,D,O | D | I,CI,C,D,O | CI,D,O |
| O | O | CI,O | C,D,O | C,D,O | I,CI,C,D,O |

I : Identical, C : Contains, CI : Contained In, D : Disjoint, O : Overlap

에서 정의한 관계가 정의되고 이에 따른 통합이 이루어 진다. 이 통합 과정에서 엔티티 타입의 도메인들 사이의 관계에 따라 도메인 관계 행렬(DR matrix : Domain Relation matrix, 성분값 : I, C, O, D)이 정의된다. 이때 포함관계 도메인을 정의하는 'C' 타입의 관계는 포함하고(contains), 포함되는(contained in) 관계인 $E_1 \subset E_2$, $E_2 \supset E_2$로 각각 분류하여 정의한다. 스키마 통합 시스템의 통합 과정에서 정의된 도메인 관계 행렬은 이행적 관계 테이블(TR table : Transitive Relation table)에 의해 조사됨으로써 엔티티 타입의 도메인 사이에 관계가 검증되는 것이다. 본 연구에서는 〈표 1〉에서 이행적 관계 테이블을 정의하고 있으며 〈표 2〉가 초기 도메인 관계 행렬(IDR)의 예를 보여주고 있다. 한편 [예제 4.1]은 추론 알고리즘을 예를 이용하여 보여준다.

IDR은 통합 단계에서 관계된 엔티티 타입으로 정의된다. 정의된 관계(I,C,CI,O, or D)는 null값과 함께 IDR의 성분값이 된다. 이때 null값은 정의된 IDR의 초기 값을 사용하여 이행적 규칙 테이블에 의해 구체적인 값으로 변환된다. Null 값은 5가지의 관계(I,C,CI,O,D)가 가능함을 의미한다. 한번의 반복 처리가 끝난 IDR은 다음의 반복을 위한 초기 IDR로 다시 정의되며 모든 성분의 값이 정의될 때까지 반복되어 새로운 도메인 관계 행렬(NDR : new DR matrix)을 생성한다. 새로이 생성된 도메인 관계 행렬은 통합 시스템에서 정의된 도메인 관계 행렬과 비교되어 불일치되는 성분의 값을 찾는다. NDR이 구성되는 과정은 다음과 같다.

〈표 2〉 초기 도메인 관계 행렬(IDR)
〈Table 2〉 Initial Domain Relation matrix(IDR)

| | E1 | E2 | E3 | E4 | E5 |
|---|---|---|---|---|---|
| E1 | I | D | n | n | CI |
| E2 | D | I | O | n | n |
| E3 | n | O | I | D | n |
| E4 | n | n | D | I | C |
| E5 | C | n | n | CI | I |

Ei : Entity_type, n : null(I,O,C,CI, or D)

[예제 4.1] IDR을 〈표 2〉에서 정의하고 이행적 규칙 테이블(TR table)을 이용하는 추론 과정을 설명한다. 다음과 같이 추론 관계를 정의하고 표기한다.

(E1 r1 E2) ∧ (E2 r2 E3) ---> (E1 r3 E3), Ei : 엔티티 타입, rj : 엔티티 타입의 도메인간 관계

① 만일 추론에 의해 r3(in E1 r3 E3)를 정의한다면, 3가지 추론 관계를 갖는다.

(E1 D E2) ∧ (E2 O E3) ---> (E1 r3 E3) r3 = [CI,D,O]

(E1 n E4) ∧ (E4 D E3) ---> (E1 r3 E3) r3 = [n]

(E1 CI E5) ∧ (E5 n E3) ---> (E1 r3 E3) r3 = [n]

이때, 3가지의 추론 관계를 모두 만족시키는 r3의 값은 r3의 값들([n],[n],[CI,D,O]) 중 하나 ,[CI,D,O], 이다. 즉, E1, E3의 관계는 1차 반복에 의해 3가지 관계인 CI, D, O가 가능하다는 것이다. 이 값은 계속 반복되는 처리를 위해 따로 저장되며, 반복을 통해 가능한 값의 범위를 축소하거나 구체적인 값으로 나타나게 된다. 첫 번째의 구체적인 값으로 정의되는 r3의 값은 IDR의 성분 값이되어 다음 반복을 준비한다.

② 만일 추론에 의해 r3(in E1 r3 E4)를 정의한다면, 3가지 추론 관계를 갖는다.

(E1 D E2) ∧ (E2 n E4) ---> (E1 r3 E4) r3 = [n]

(E1 n E3) ∧ (E3 D E4) ---> (E1 r3 E4) r3 = [n]

(E1 CI E5) ∧ (E5 CI E4) ---> (E1 r3 E4) r3 = [CI]

이때, 3가지의 추론 관계를 모두 만족시키는 r3의 값은 r3의 값들([n],[n],[CI]) 중 하나 , [CI]이다. 이 경우에 [CI]는 IDR($E_1$,$E_4$)의 구체적인 값으로 정의되어 IDR에 저장된다.

③ 만일 추론에 의해 r3(in E2 r3 E4)를

정의한다면, 3가지 추론 관계를 갖는다.

(E2 D E1) ∧ (E1 CI E4) ---> (E2 r3 E4) r3 = [CI,D,O]

(E2 O E3) ∧ (E3 D E4) ---> (E2 r3 E4) r3 = [C,D,O]

(E2 n E5) ∧ (E5 CI E4) ---> (E2 r3 E4) r3 = [n]

이때, 3가지의 추론 관계를 모두 만족시키는 r3의 값은 r3의 값들([CI,D,O],[C,D,O],[n]) 중 정의 가능한 관계는 [D,O]이다. 이 경우에 [D,O]는 IDR($E_2$,$E_4$)의 구체적인 값으로 정의되기 위하여 다시 반복되어 값의 범위가 축소되거나 하나의 값으로 확정된다.

④ 만일 추론에 의해 r3(in E2 r3 E5)를 정의한다면, 3가지 추론 관계를 갖는다.

(E2 D E1) ∧ (E1 CI E5) ---> (E2 r3 E5) r3 = [CI,D,O]

(E2 O E3) ∧ (E3 n E5) ---> (E2 r3 E5) r3 = [n]

(E2 n E4) ∧ (E4 C E5) ---> (E2 r3 E5) r3 = [n]

이때, 3가지의 추론 관계를 모두 만족시키는 r3의 값은 r3의 값들([CI,D,O],[n],[n]) 중 하나 ,[CI,D,O]이다. 이 경우에 [CI,D,O]는 다음 반복시 정의되는 값과 비교되기 위해 임시 테이블에 저장된다.

⑤ 만일 추론에 의해 r3(in E3 r3 E5)를 정의한다면, 3가지 추론 관계를 갖는다.

(E3 n E1) ∧ (E1 CI E5) ---> (E3 r3 E5) r3 = [n]

(E3 O E2) ∧ (E2 n E5) ---> (E3 r3 E5) r3 = [n]

(E3 D E4) ∧ (E4 C E5) ---> (E3 r3 E5) r3 = [D]

이때, 3가지의 추론 관계를 모두 만족시키는 r3의 값은 r3의 값들([n],[n],[D]) 중 하나 , [D]이다. 이 경우에 [D]는 IDR($E_1$,$E_4$)의 구체적인 값으로 정의되고 IDR에 저장된다.

이와 같은 방법에 의해 다시 정의되는 IDR 행렬은 〈표 3〉과 같으며 2차의 반복을 위하여 준비된다. 2차 반복에서는 재정의된 IDR을 이용하여 다시 알려지지 않은 관계를 추론하고 구체적인 값을 얻는다. 2차 반복에서도 확정되지 않는 값은 1차 반복에서 임시 정의된 값들과 비교 정의된다. 이때 값의 범위가 축소되거나 정의된다.

〈표 3〉 2차 정의된 초기 도메인 관계 행렬(IDR)
〈Table 3〉 Redefined Initial Domain Relation matrix(IDR)

| | E1 | E2 | E3 | E4 | E5 |
|---|---|---|---|---|---|
| E1 | I | D | n | CI | CI |
| E2 | D | I | O | n | n |
| E3 | n | O | I | D | D |
| E4 | C | n | D | I | C |
| E5 | C | n | D | CI | I |

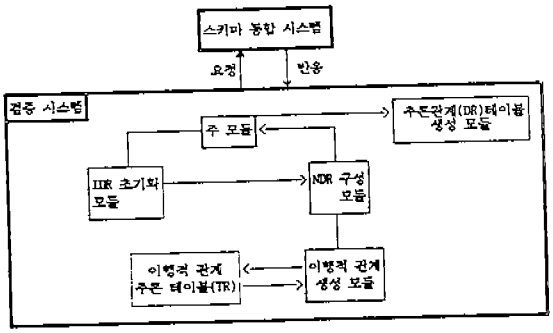
Ei : Entity_type, n : null(I,O,C,CI, or D)

[알고리즘 4.1] 엔티티 통합 관계 검증 알고리즘

```
/* 엔티티 통합 도메인 관계 검증 알고리즘 */
/* i,j : 통합될 엔티티 중 임의의 엔티티를
   정의한다. */
1. Input DR matrix /* user interactive
   value in integration step */
2. Construct TR table /* TR : Transitive
   Relation table */
3. Construct IDR matrix /* IDR : Initial
   Domain Relation */
4. REREAT
   US ::=Universal set of relations among
   domains;
   KR ::=Set of known relations;
   UR ::=Set of unknown relations with
   null;
   T ::=all transitive relation;
    WHILE (UR≠∅) Do
    { r ::=one of edge of UR;
      Ti ::=all transitive path for r ⊂ T;
        R ::={};
        WHILE (Ti≠∅) Do
```

{ ri: =value of TR;
R: =R∩ri;
}
ENDWHILE;
IF(n(R)=1) /* n(R) : The number of values that R hold */
THEN { IDR($E_i$,$E_j$): = R;
NDR($E_i$,$E_j$): =R; /* NDR : new DR matrix */
KR: =producted value of relation;
UR: =US − KR; }
ELSE IF(R=∅)
THEN inconsistency relation of r, IDR($E_i$,$E_j$);
ELSE NDR($E_i$,$E_j$): =R;
}
ENDWHILE;
UNTIL(IDR=NDR);
5. IF(NDR($E_i$,$E_j$)=DR($E_i$,$E_j$)) /* for all 1≤ i,j ≤ n */
≤THEN "validation of entity type integration";
ELSE "check inconsistency of DR's element";
6. STOP

알고리즘으로 제시하고 있는 검증 시스템은 스키마 통합 시스템에 의한 검증 요청에 따라 통합되는 엔티티들의 도메인 관계를 검증하고 이에 대한 타당성 관계를 제공한다. 본 논문에서 구현한 이 검증 시스템은 주 모듈을 포함하여 5가지 구성 모듈로 이루어지며 (그림 2)와 같다. IDR 초기화 모듈은 통합 시스템으로 부터 초기 엔티티 도메인 관계를 받아 도메인 관계 테이블을 초기화 한다. 이행적 관계 생성 모듈은 요청받은 관계에 대하여 이행적 규칙에 따른 타당한 관계 결과를 NDR 구성 모듈에 제공한다. 이때 주 모듈은 새로이 구성된 IDR 테이블을 이용하여 반복을 계속하며 모든 엔티티의 도메인 관계를 추론한다. 추론 관계(DR) 테이블 생성 모듈은 통합 시스템에 의해 이미 구성된 도메인 관계 테이블과의 분석 결과를 통합 시스템에 제공한다.




(그림 2) 검증 시스템 구조도
(Fig. 2) Diagram of verification system

## 5. 결 론

본 논문에서는 스키마 통합을 위한 기반 모델로 다중 측면 모델을 사용하여 각 지역 스키마의 시멘틱을 유지하고자 했다. 또한 효과적으로 다양한 시멘틱의 표현을 지원하고자 하였다. 이 모델을 기반으로 스키마의 통합을 위하여 엔티티 타입의 도메인 간의 관계를 정의하였으며 이를 기반으로 엔티티 타입의 통합 타당성을 검증하기 위한 알고리즘을 제안하고 이를 구현하였다. 제안된 알고리즘은 처리 과정에서 발생하는 요소들 사이의 관계와 이들의 관계식에 의해 다음과 같이 분석된다. n을 통합되는 엔티티 타입의 수, 릴레이션(relation)을 도메인들 사이의 관계(즉, I, C, CI, O, D)라 하자. 알고리즘의 수행에 의한 DR과 NDR의 비교는 DR성분의 수와 NDR과 일치하는 성분의 비로 관계에 대한 일치도를 정의할 수 있다. 일치도의 관계식은 다음과 같다.

$$\text{일치도(Consistency Degree : CD)} = \frac{\text{타당한 릴레이션의 수}}{\text{전체 릴레이션의 수}}$$

$$,0 \leq CD \leq 1.$$

$$, \text{전체 리레이션의 수} = \frac{n(n-1)}{2}$$

전체 알려지지 않은 릴레이션의 수

$$= \frac{n(n-1)}{2}$$

− 알려진 릴레이션의 수.

(null값을 갖지 않는 IDR의 성분 수)

하나의 릴레이션당 이행적 릴레이션의 수

$$= n-2.$$

한 릴레이션에 대한 알려진 이행적 릴레이션의 수와 그 릴레이션에 대한 전체 이행적 릴레이션 수의 비에 대한 관계식은 다음과 같다.

즉, 정확도(Accuracy of one Relation : AR)

$$= \frac{\text{알려진 이행적 릴레이션의 수(k)}}{\text{하나의 릴레이션당 이행적 릴레이션 수}}$$

$$= \frac{k}{n-2}$$

$$,0 \leq k \leq n-2$$

그러므로, AR의 값은 검증 알고리즘의 반복 횟수에 영향을 미치며 클수록 작은 반복 횟수에 의해 도메인별 관계 값을 찾는다. 이는 AR의 값이 알려진 이행적 relation의 수와 관계함에 따라 정확도에 영향을 주며 k값의 증가는 DR의 성분값에 빠르고 정확하게 접근하게 한다.

엔티티 타입의 통합은 관계(relationship) 통합과 함께 스키마 통합에 있어 중심 요소이며, 따라서 엔티티 타입의 통합을 위한 이들의 관계 검증은 통합 모델의 속성인 정확성의 확보를 위하여 중요한 과제이다. 또한 검증 모델의 이론적 배경을 위한 형식화된 정의를 위한 연구가 진행되고 있다. 이러한 알고리즘의 연구와 개발은 스키마 통합의 전과정의 검증을 위하여, 또한 통합된 스키마의 검증을 위하여 계속 연구되어야 하며 좀더 빠르고 정확한 검증을 위한 평가가 이루어져야 한다.

## 참 고 문 헌


[1] David Bell, Jane Grimson, Distributed Database Systems, Wokingham : Addison-Wesley, 1992.

[2] A.P. Sheth and J.A. Larson, "Federated database systems for managing distributed, heterogeneous, and autonomous database systems," ACM Computing Survey 22(3), pp. 183-236, Sep. 1990.

[3] R. Elmasri and D. Wiederhold, "Data Model Integration Using the Structural Model," ACM-SIGMOD, 1979.

[4] S. Navathe, T. Sashidhar, and R. Elmasri, "Relationship Merging in Schema Integration," In pro. of the 10th VLDB, pp. 78-90, Aug. 1984.

[5] Chang−Wha Kim, "Knowledge Representation Modelling based on the EA Model", Ph.D. dissertation, Korea University, 1989.

[6] Dong-Young Cho, "The Object-Oriented Database Modelling with Multiple-Aspects", Ph.D. dissertation, Korea University, 1991.

[7] Sung-Hun Kim, "A study on Database Schems Integration Methodology using the EA model", M.Ph. dissertation, Korea University, 1988.

[8] C. Batini and M. Lenzerini, "A Methodology for Data Schema Integration in the Entity Relationship Model," IEEE trans. on software engineering SE-10 (6), pp. 650−664, Nov. 1984.

[9] Won Kim and Jungyun Seo, "Classifying Schematic and Data Heterogeneity Multidatabase Systems," IEEE Computer 24(12), pp. 12-18, Dec. 1991.

[10] C. Batini, M. Lenzerini, and S. B. Navathe, "Comparative Analysis of Methodologies for Database Schema Integration," ACM Computing Survey 18 (4), pp. 323-364, Dec. 1986.

[11] G. Thomas and G.R. Thompson et al,

"Heterogeneous Distributed Database Systems for Production Use," ACM Computing Surveys 22(3), pp. 237-266, Sep. 1990.

[12] N. Cercone, M. Morgestern, A. Sheth, and W. Litwin, "Resolving semantic heterogeneity," International Conference on Data Engineering, Vol. 6, 1990.

[13] Chin-Wan Chung, "DATAPLEX : An Access to Heterogeneous Distributed Databases," Communications of the ACM 33(1), pp. 70-80, Jan. 1990.

[14] A.K. Elmagarmid, Database Transaction Models for Advanced Applications, Morgan Kaufmann Publishers Inc., San Mateo, California, 1990.

[15] W. Litwin, L. Mark, and N. Roussopoulos, " Interoperability of multiple autonomous databases," ACM Computing Survey 22(3), pp. 267-293, Sep. 1990.

[16] Yoeng-won Kim, Chong-sun Hwang, " Relationship Integration based on a Multiple Aspects Model," Pro. of first ICCTA, pp. 63-68, Aug. 1994.

김 용 원

1986년 고려대학교 수학과 졸업(학사)
1988년 고려대학교 대학원 전산학 전공(석사)
1993년 고려대학교 대학원 전산과학과 박사 과정 수료
관심분야 : 분산데이타베이스, 분산 시스템, 알고리즘.