論 文

객체-관계 대응 함수를 이용한 EDI-MS용 MSIB의 설계 방법

正會員 白 勝 惠* 正會員 金 泰 潤*

A Design Methodology of the MSIB(Message Store Information Base) for EDI-MS(EDI-Message Store) Using Object-Relation Mapping Function

Seung Hye Paik*, Tai Yun Kim* *Regular Members*

요 약

EDI 통신 전용 프로토콜로 1990년도에 CCITT와 ISO/IEC가 공동으로 발표한 X.435는 1988년도 X.413 권고안에서 제안한 MHS용 MS의 기능을 EDI로 확장하기 위한 EDI-MS의 메시지 형태와 추상적 서비스를 기술하고 있다.

본 논문에서는 MHS용 MS를 EDI-MS로 확장하고자 할 경우 그 저장 구조가 되는 MSIB(Message Store Information Base)를 구현할 때 필요한 설계 방법을 연구하였다.

본 논문에서 제시한 설계 방법은 객체-관계 대응 함수를 이용한 설계 방법이다. 이 방법은 MSIB에 저장되는 EDI 메시지 형태와 특성을 고려하였다.

제안한 설계 방법은 별도의 정규화 과정없이 제 3 정규형을 만족하는 MSIB 구성 테이블을 설계할 수 있는 장점이 있다. 제안한 설계 방법에 의해 구축된 MSIB는 다중 값의 저장 및 검색시 효율적이다.

ABSTRACT

X.435 is the EDI communication protocol published by CCITT and ISO/IEC in 1990. The message types and abstract services of EDI-MS are specified in X.435 recommendation.

In this paper, the design methodology on implementation of MSIB in the case that MHS-MS is required to be extended into EDI-MS has been studied.

The design methodology presented in this paper is the one using the mapping function of the object-relation. This methodology considers the EDI message types and chracterictics.

The proposed design methodology has an advantage that it is able to design the component tables of MSIB satisfying the third normal form and it dose not require the normalization processing. The MSIB constructed by the proposed method can store and retrieve multi-values efficiently.

*高麗大學校 電算科學科

論文番號 : 94 15

Ⅰ. 서 론

나날이 발전하는 첨단 정보 통신 기술은 고도의 정보화 사회로 가는 초석이 되고 있다. 첨단 정보 통신 기술로 기업간 문서 교환을 전자적으로 수행하여 사무자동화 처리의 효율을 극대화하는 EDI(Electronic Data Interchange : 전자식 자료 교환)에 대한 필요성이 요구되고 있다. EDI란 기업간 문서 교환을 표준화된 거래서식에 맞춰 컴퓨터와 컴퓨터 통신망을 이용하여 전자적으로 수행하는 시스템이다.[16]

이러한 EDI 시스템 개발이 국외는 물론 국내에서도 활발히 전개되고 이에 대한 많은 연구가 진행중이다.[16] 특히 EDI 통신 시스템에 관한 연구의 필요성이 증대되고 있다. 이에따라 1990년도에 CCITT와 ISO/IEC가 공동으로 EDI용 통신 표준으로 CCITT X.435/F.435 권고안을 발표하였다.

1988년도 CCITT X.413 권고안에서는 원격지 사용자를 대신하여 메시지를 수신하고 사용자의 요구에 따라 지연 송신이 가능한 MS(Message Store)를 제안하였다.[3] EDIMS(EDI Messaging System)의 범위가 확대됨에 따라 EDI 전용 서비스를 지원하고 X.435 권고안에 입각한 EDI용 MS(EDI-MS : EDI-Message Store)의 개발이 필요하다.

EDI-MS는 EDI 메시지를 저장 후 전송(store-and-forward)하는 기능과 사용자가 수신된 메시지를 검색, 관리할 수 있는 기능을 제공한다. EDI-UA가 EDI-MS에 P7 통신 프로토콜을 이용하여 요구하는 서비스를 얼마나 효율적으로 지원하는지의 여부는 EDI-MS의 수신 메시지를 저장하는 하부 저장 모듈인 MSIB 구조에 따라 다르다.

X.435 권고안에 따르는 EDI-MS의 설계와 구현은 다양한 방법으로 가능하다. EDI-MS를 구축하기 위해서는 기존에 구축되어 있는 X.413의 MHS-MS를 확장하는 것이 EDI 시스템에서 중시되는 경제적 측면과 연동성을 고려할 때 바람직하다.

본 논문에서는 MHS-MS가 관계형 데이타베이스 관리 시스템(RDBMS)을 도입, 저장 하부 모듈로 사용하는 추세[5, 15, 17]를 고려하였다. EDI-MS용 MSIB 또한 RDBMS를 수용하여 구축함을 전제로 하였다.

EDI-MS의 저장 하부 모듈로 RDBMS를 도입하여 구현하는 방법들이 본 논문과 동일한 관점에서 제시되고 있다.[17] 제시된 방법은 중첩된 구조로 회송 통신을 지원하게끔 구성된 계층 구조의 EDIM(EDI Message : EDI 메시지)을 ECR(Entity-Category-Relationship : 개체-카테고리-관계) 모델로 모델링하고 이를 관계 모델로 변환 하였다.

이 설계 방법은 EDI-MS용 MSIB에 저장되는 EDIM의 특성을 고려하지 않고 일반적인 설계 방법을 이용함으로 EDIM의 특성상 발생하는 복합 애트리뷰트의 처리, 다중 값 처리, 별도의 정규화 과정을 필요로 한다.

본 논문은 RDB 설계시 OMT(Object-Oriented Modeling Technique) 모델을 이용한 기법[10]을 EDIM의 특성에 적합하도록 확장한다. 확장하여 제안한 설계 방법을 EDI-MS용 MSIB 설계에 도입함으로써 기존 설계시 필요한 일련의 과정 없이도 MSIB를 이용한 서비스 오퍼레이션에 적합하도록 구조화된 MSIB를 설계할 수 있다.

제안한 객체-관계 대응 함수를 이용한 MSIB 설계 방법은 개념적 설계 단계의 OMT 객체 클래스들에 객체-관리 대응 함수를 적용하여 관계성이 필드로 추가된 논리 테이블을 생성할 수 있다. ASN.1의 구조형으로 표현되는 복합 애트리뷰트 처리를 위해 추가된 관계성 필드를 이용함으로써 기존에 필요로 하였던 driven table 없이도 복합 애트리뷰트 처리가 가능하다.

MSIB 구성 테이블에 관계성 필드를 보유함으로 P7 프로토콜을 이용하여 검색 연산을 실행할 경우 필요한 MSIB 구성 테이블간의 조인 연산(join operation)을 줄일 수 있다. 논리 설계시 필요한 별도의 정규화 과정을 생략하고도 MSIB 구성 테이블이 제 3 정규형을 만족함을 증명하였다.

이에 대한 내용으로 2장에서 EDI-MS의 정의, 기능 및 역할, MSIB와의 관계를 명시하였다. 그리고 MSIB의 저장 정보 객체인 EDIM과 EDIN의 역할, 구조상의 특징을 제시하였다. 3장에서는 MSIB 설계를 위한 새로운 방법인 객체-관계 대응 함수를 이용한 설계 방법을 제안하였다. 4장에서는 제안한 설계 방법을 분석하고 효과를 증명하였다. 5장에서는 본 논문의 결론과 추후 연구 과제를 제시하였다.

Ⅱ. EDI-MS 및 MSIB

2.1 EDI-MS(EDI Message Store)의 MSIB

최근에 EDI를 일반 업무에서 활발히 사용하게 됨으로써 CCITT X.400 프로토콜을 이용한 EDI 통신 처리에 대한 한계점이 발생하였다. 이러한 한계를 극복하기 위하여 CCITT와 ISO/IEC가 공동으로 X.400 MHS를 기반으로 하고 개방형 시스템을 지향하는 EDI 통신 규약인 X.435/F.435 권고안을 발표하였다.[4]

CCITT X.435 권고안에서는 EDI 정보 교환에 적합하

도록 작성된 메시지를 처리하는 형태인 EDIMS(EDI Messaging System)를 정의하였다. EDIMS는 UA (User Agent : 가입자 시스템), MS(Message Store : 메시지 저장 시스템), MTS(Message Transfer System : 메시지 전달 시스템)로 구성된다.[4]

1990년도에 발표한 X.435의 MS는 1988년도 X.413 권고안과 비교해 통지(notification), 회송(responsibility)에 관한 항목이 추가되고 메시지 보안(security)에 관한 항목이 강화되었다.[21] MS가 선택적으로 EDI 메시지 통신 특유의 애트리뷰트 유형과 자동 동작을 추가적으로 제공할 수 있을 때 EDI 메시지 통신을 위한 특별 MS로 'EDI-MS'라 한다.[4] 이는 X.413을 기초로 하는 MS의 애트리뷰트와 X.435 권고안을 기초로 하는 애트리뷰트 모두를 지원할 수 있어야만 독립된 다른 EDI 메시징 시스템과의 연동에 문제가 없다.[21]

EDI-MS를 사용함으로써 EDI-UA의 저용량성, 처리 속도, 처리 능력의 한계성을 극복할 수 있고 EDI-UA가 동작 불능시에도 수신 기능을 수행할 수 있다. EDI-MS는 EDI-UA를 대신하여 EDIM 회송시 필요한 EDI 특유의 책임 문제를 처리할 수 있다.

그림 1은 EDI-MS의 개념도다. EDI-MS는 X.413에서 제안한 MS와 동일하게 MTS로부터 P3 프로토콜을 이용하여 메시지 배달을 요청받는다. EDI-UA로부터 P7프로토콜에 따라 검색과 관리 서비스를 요청받게 된다. 제출에 대한 요청일 경우 EDI-MS는 MTS로 전달하는 역할을 담당하고 다른 요청일 경우는 직접 처리한다.[4, 19]

검색 서비스는 MTS로부터 배달된 EDIM, EDIN을 MSIB에 구성 애트리뷰트별로 저장하고 있기 때문에 EDI-UA에게 제공할 수 있다. MSIB에는 이러한 EDIM, EDIN과 같은 전달 메시지들, 자동 동작(auto-action)에 관한 매개 변수, MTS 연결/해제(Bind/Unbind) 및 EDI-MS 관리 전달 애트리뷰트가 있다.[19, 20]

MSIB는 기본적인 물리적 접근 방식의 제공, 다중 사용자의 동시 접근및 트랜잭션에 대한 무결성 보장, 각종 장애로부터의 회복및 복구 기능, 관리기능 등을 요구한다. 이러한 관점과 실용적인 관점을 고려하여 MSIB 개발에 관계형 데이타베이스 관리 시스템을 도입한다.[21]

2.2 MSIB 저장 정보 객체

MSIB에 저장되는 주된 정보는 EDIMS 환경에서 교환되는 정보 객체(Information Object)인 EDIM과 EDIN이다. EDIM은 사용자가 EDI 인터체인지를 전송하기에 적합하도록 설계된 X.400 메시지다. 이는 메시지를 회송하고 응답 신호를 보내는데 적합하도록 특별히 설계된 메시지다.

EDIM은 머리(heading) 부분과 몸체(body) 부분으로 구성된다. 머리 부분은 메시지에 관한 정보와 EDI 사용자가 요청한 서비스를 제공하기 위해서 필요한 요소들의

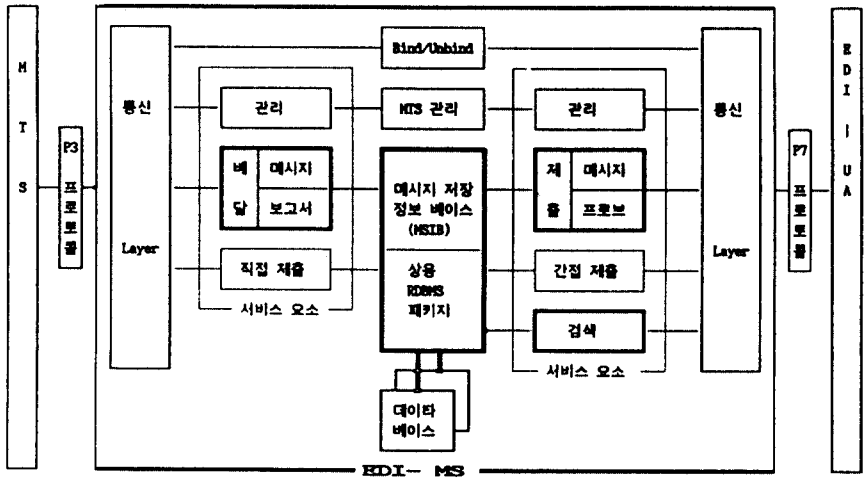

그림 1. EDI-MS 개념도
Fig 1. The conceptual description of EDI-MS

집합이다. 메시지 특징을 기술한 필드, 서비스 제공을 위한 필드, 전달에 관한 필드 등으로 이루어져 있다.

몸체 부분은 사용자에게 전달하는 데이타인 EDI 인터체인지를 포함하는 경우와 다른 곳으로 전송될 메시지인 회송 EDIM을 포함하는 경우로 나뉘어진다.[13] 회송 EDIM은 이를 포함한 EDIM과 같은 구조로 되어 있다. 따라서 EDI-MS 저장 측면을 고려할 때 이러한 중첩 구조를, 트리 구조로 표현하는 것이 바람직하다.[12] 그림 2는 회송 메시지를 포함하는 EDIM 특유의 애트리뷰트들이 어떻게 EDI-MS에 대응되는지를 보여주고 있다.[1]

EDIN은 수신측 UA가 송신측 UA에게 보내는 응답 메시지다. EDIN은 수행하는 역할에 따라 기질 통지, 수용 통지, 회송 통지의 세가지 형태로 구분된다. EDIN은 메시지 식별자, EDIN 송신자, EDIN 수신자, 최초 수신자, 보안 요소, EDIN 식별자 등의 공통 필드와 각각의 EDIN과 관련된 특정 필드로 구성된다.[21]

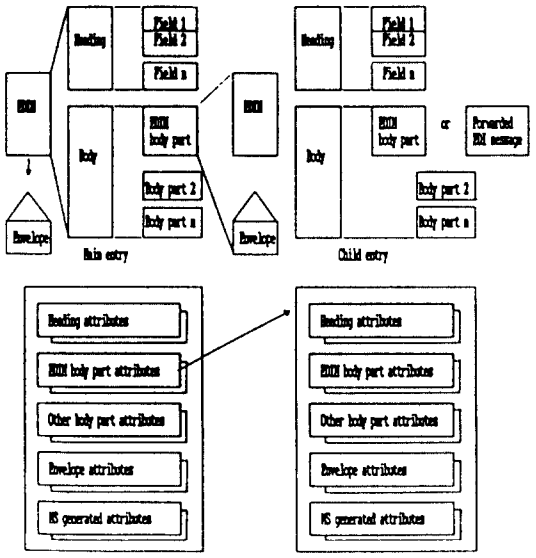

그림 2. EDIM을 포함하는 EDI 메시지 및 MS로의 대응[1]
Fig 2. EDI message that contains EDIM and its mapping to MS[1]

EDIM과 EDIN이 MS에 저장될 때 EDI-MS는 EDIM의 머리 부분의 필드들이나 EDIN의 대부분의 필드들을 MSIB 애트리뷰트(attribute)로 변형시킨다. EDI 통신 시스템상의 전달 메시지들은 데이타베이스로 간주되는 MSIB의 엔트리(entry)로 취급한다. 각각의 메시지는 이미 권고안에서 제시한 애트리뷰트들의 결합이라고 할 수 있다. 이렇게 메시지에 관련된 정보를 해당되는 애트리뷰트로 분할하여 저장함으로써 메시지를 조회 처리할 때 사용한다. 예를 들면 송장(Invoice)을 가지는 모든 EDIM을 검색하고자 하면, 머리 부분의 'EDI Message Type' 필드에 해당하는 MSIB 애트리뷰트를 검색하여 이 요구 사항에 적합한 결과값을 반환하면 된다.

III. 객체-관계 대응 함수를 이용한 MSIB 설계 방법

3.1 MSIB 설계 단계

본 논문은 EDI MS용 MSIB 구축을 위해 상용 RDBMS를 이용함을 전제로 하였다. 따라서 일반적인 RDBMS의 설계 방법을 이용하여서도 MSIB 구축이 가능하다. 그러나 EDI MS의 검색 서비스를 효과적으로 지원하기 위해서는 모델링 단계에서부터 MSIB 저장 정보 객체인 EDIM, EDIN의 특성을 고려하는 것이 바람직하다.

기존의 설계 방법은 계층 구조이고, 하부 객체들을 포함하는 복합 객체인 EDIM, EDIN을 ECR 모델을 이용하여 개념적으로 모델링한다. 관계 테이블을 이용하여 논리적으로 설계한다. 해당 객체내에 존재하는 애트리뷰트가 원자적일 때까지 개념 설계 및 논리 설계를 반복해야 한다.

이 경우는 MSIB를 구성하는 논리 테이블 갯수가 상당히 증가한다. 각 테이블마다 대리 인식자가 존재하며, 하부 구조의 테이블일 경우는 상부 구조의 테이블의 대리 인식자를 포인팅하는 외래 키(foreign key)를 추가로 가져야 하는 단점이 있다.[17] 이를 해결하기 위한 별도의 지원이 필요하다.

본 논문에서 제안한 설계 방법은 RDB 설계시 OMT 모델을 도입하는 기존의 연구[10]를 확장한 설계 방법으로 EDI MS용 설계라는 특성을 충분히 반영한다.

개념적 설계 단계에서 OMT 다이어그램을 이용하여 MSIB 저장 정보 객체를 모델링한다. 논리적 설계시 OMT의 객체 클래스를 관계 테이블로 대응한다. MSIB는 하나의 EDIM 엔트리를 저장하기 위해 EDIM 구성 특성을 고려하여 계층 구조화된 여러개의 객체 클래스로 표현하였다.

OMT 모델을 RDB 설계시 도입하는 기존의 연구를 이용할 경우는 계층화된 객체 클래스를 관계 테이블로 대응할 때, 일반화, 연관화, 집단화, 관계중 하나의 관계성만을 나타낸다. OMT 다이어그램에 의해 계층 구조로 표현되는 객체 클래스를 관계 테이블로

대응하기 위해서는 상위 객체 클래스와의 관련성 뿐만 아니라 하위 객체 클래스와의 관련성 또한 관계 테이블 내에서 표현할 수 있어야 한다.

본 논문은 OMT 모델을 RDB 스키마(schema)로 대응하는 대응 법칙을 함수로 표현하고 합성 함수를 정의하였다. 이를 객체-관계 대응 함수라 한다. OMT 모델에 객체-관계 대응 함수를 적용하는 단계를 개념 설계와 논리 설계 중간에 도입한다.

본 논문에서 제안한 객체-관계 대응 함수를 이용한 설계 방법은 EDIM 특성을 고려하였기에 EDI-MS용 MSIB 설계에 적합하다.

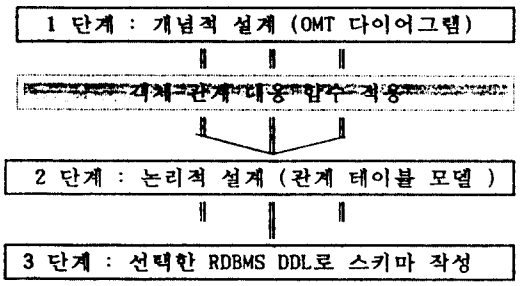

그림 3. 객체-관계 대응 함수를 이용한 MSIB 설계 방법
Fig 3. The design methodology on the MSIB using the mapping function of the object-relation

이 설계 방법은 MSIB를 개발함에 있어 상용 RDBMS를 사용하되 객체 지향 패러다임(object-oriented paradigm)을 이용하여 저장되는 정보의 추상화 수준을 높인다. OMT 모델에서 RDB 스키마로의 대응 과정을 합성 함수화 하였기 때문에 OMT 다이어그램의 각각의 객체 클래스에 객체-관계 대응 함수를 한번씩만 적용하면 된다. 이는 OMT 다이어그램에서 RDB 스키마를 자동적으로 생성해 주는 툴(tool)을 개발할 경우 생성 알고리즘을 쉽게 작성할 수 있도록 한다.

저장 정보 객체들간의 관계성을 관계 테이블안에서 관리할 수 있으므로 다중 값 처리시 관계성 애트리뷰트를 사용한다. 별도의 정규화 과정 없이 제 3 정규형을 만족하는 관계 테이블 모델을 얻을 수 있는 장점을 가진다.

본 논문의 설계 방법은 EDI-MS용 MSIB 설계라는 측면을 고려하여 개발하였지만 이 경우에만 국한된 방법론은 아니다. OMT 모델을 이용한 RDB설계시 객체 클래스들간에 두가지 관계성을 고려해야 할 경우, 제안한 객체-관계 대응 함수를 이용할 수 있다.

3.2 OMT를 이용한 MSIB의 개념적 설계

MSIB 개념적 설계에서 RDB 설계시 일반적으로 많이 사용하는 ECR 모델을 채택하는 것 보다 다음과

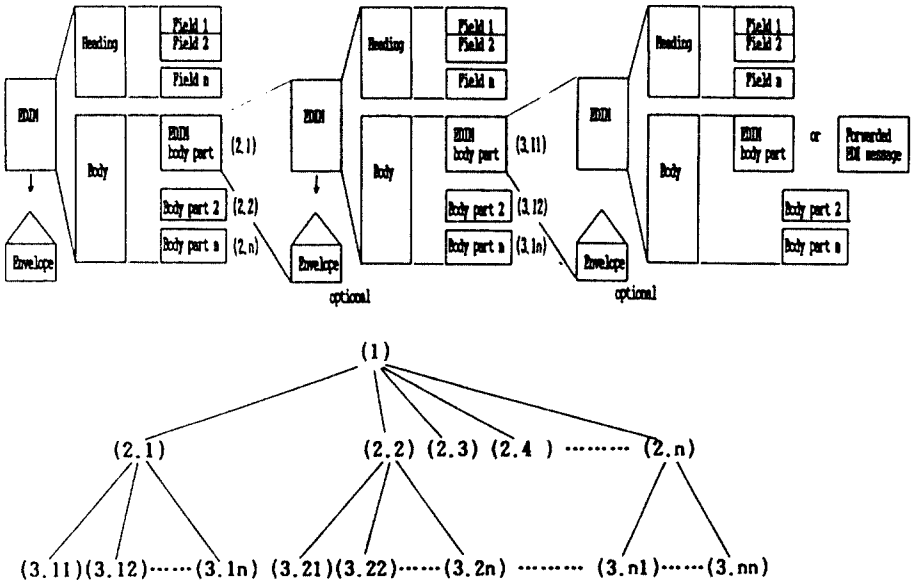

그림 4. EDIM 중첩 구조와 대응 트리
Fig 4. EDIM nested structure and mapping tree

같은 이유로 OMT 모델을 채택하는 것이 적합하다.

첫째, EDIM은 MSIB의 주요 엔트리로 머리 부분과 몸체 부분으로 나뉘어진다. 몸체 부분에는 회송 EDIM을 포함한다. 회송 EDIM은 또 다른 회송 EDIM을 포함할 수 있다. 이러한 EDIM의 중첩 구조는 그림 4에서와 같이 계층화된 트리 형태로 표현된다. 몸체 부분은 전송되는 EDI 문자 정보와 함께 음성 정보, 화상 정보 등 멀티미디어를 위한 데이타를 포함할 수 있다. 따라서 MSIB 개념 설계는 상위 클래스의 속성을 상속받을 수 있는 OMT 모델을 이용하는 것이 적합하다.

둘째, EDIM의 몸체 부분에 포함되는 회송 EDIM은 이를 포함하는 EDIM과 동일한 구조, 즉 재귀적(recursive) 구조임을 알 수 있다. 이는 OMT 모델의 'Recursive Aggregates'을 사용하면 정확하게 나타낼 수 있다.

셋째, MSIB의 저장 정보 객체인 EDIM, EDIN, EDI-MS의 추상적 서비스는 X.435 권고안에서 ASN.1 구문 표현으로 표기되어 있다. ASN.1 문법이 표현하는 데이타 유형은 단순형과 구조형으로 분류한다.[1] [9] 단순형은 일반 프로그래밍 언어에서 제공하는 데이타 유형과 유사하므로 이를 RDBMS에 저장하는 데 어려움이 없다. 그러나 구조형은 ASN.1 표기에서만 볼 수 있는 독특한 유형으로 집합형(set type), 연속형(sequence type), 선택형(choice type), 임의형(any type), 태그형(tag type)의 다섯가지 유형이 있다.[1, 9] 이러한 유형 중 집합형, 연속형, 선택형은 여러개의 단순형이 복합적으로 결합되어 구성된다. 구조형은 표 1에서와 같이 OMT에서 제공하는 집단화(aggregation), 일반화(generalization), 연관화(association)가 정확하게 대응되므로 MSIB 개념 설계에 OMT 모델을 채택한다.

표 1. ASN.1 데이타 유형과 OMT와의 관계
Table 1. ASN.1 data types and the relationship with OMT

ASN.1 데이타 유형	OMT 다이어그램
SEQUENCE	aggregation
SET	aggregation
CHOICE	generalization
SEQUENCE OF	association
SET OF	association

MSIB에 저장 정보 객체인 EDIM과 EDIN의 애트리뷰트들과 MS 자체 생성 애트리뷰트들을 OMT 모델을 사용하여 모델링한 결과다. ASN.1으로 표현된 정보 객체를 개념적으로 나타내었다. 유사한 특성을 가지며 공통적으로 P7 프로토콜의 검색 서비스 오퍼레이션이 적용 가능한 객체의 집합을 객체 클래스로 모델링하였다. 그림 5의 모델링은 객체 클래스의 이름만 표현하였다. 세부 객체를 나타낸 모델링은 그림 6~그림 7에서 표현된다.

3.3 관계 모델에 의한 MSIB 논리적 설계

3.3.1 객체 지향 패러다임(paradigm)과 대응 법칙 (mapping rule)

RDB의 각 릴레이션의 열값은 시스템에서 정의한 정수, 스트링, 실수, 날짜, 시간 등의 형태로 제한된다. 이러한 제한점을 극복하기 위해 OODB의 한 애트리뷰트가 다른 튜플을 가질 수 있도록 OID(Object IDentification : 객체 식별자)를 사용하는 객체 스킴(object scheme)을 도입한다.

RDB에서는 테이블을 구성하는 애트리뷰트 중 각 튜플을 유일하게 식별할 수 있는 애트리뷰트를 기본 키로 선택하였다. 그러나 OMT를 사용한 RDB 설계에서는 OID 개념을 도입하여 각각의 테이블마다 튜플을 유일하게 나타낼 수 있는 OID라 부르는 애트리뷰트를 첨가하여 이를 기본 키로 선택한다.

이러한 OID와 함께 OMT의 각 객체 클래스들간의 일반화, 연관화, 집단화 관계를 테이블의 애트리뷰트로 첨가하여 클래스 계층성(class hierarchy), 복합 객체(complex object)등의 객체 지향 패러다임을 수용할 수 있다.

객체 지향 개념을 RDBMS 설계시 도입함으로써 추상화의 수준을 높이고, 정규화 과정을 용이하게 하며 데이타베이스와 응용 사이의 무결성 문제를 감소시킬 수 있는 장점이 있다.[6, 10]

OMT의 각 객체 클래스의 관계 테이블 모델로의 대응 과정은 다음과 같다.[7, 10]

(1) 각 객체 클래스는 각 관계 테이블로 대응된다. 각각의 객체는 테이블의 애트리뷰트로 대응된다.

(2) '객체 클래스 OID'를 테이블의 애트리뷰트로 첨가한다. 이 애트리뷰트는 후보 키로가 된다.

(3) 하나의 슈퍼 클래스(superclass)에 대해 여러개의 서브 클래스(subclass)사이에 일반화 관계가 존해라 때는 다음과 같다.

(1)슈퍼 클래스, 서브 클래스 구별없이 대응되는 테

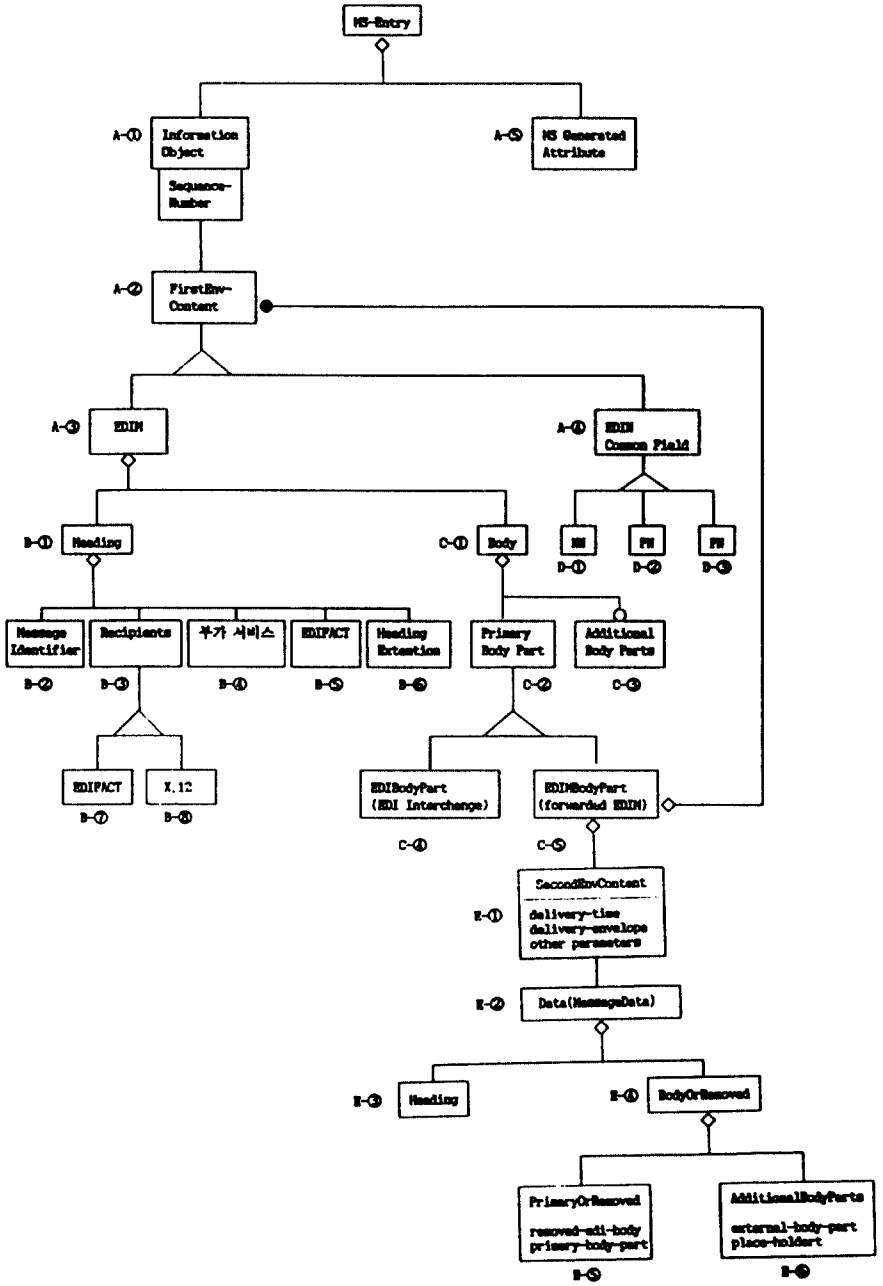

그림 5. MSIB 저장 정보 객체의 OMT 다이어그램

Fig 5. The OMT diagram of the MSIB storage information object

이블에 '슈퍼 클래스 이름 OID' 애트리뷰트를 첨가한다. 이를 후보 키의 하나의 요소로 한다.

② 슈퍼 클래스와 서브 클래스의 분류 기준인 구별자(discriminator)를 슈퍼 클래스의 대응 관계 테이블에 첨가한다. 이 애트리뷰트를 후보 키의 요소로 첨가한다.

(4) 객체 클래스들 간에 일대 다의 연관화 관계가 존재하고 연관성이 링크(link) 애트리뷰트에 의해 표현될 때는 다음과 같다.

① 일대 다의 연관화 관계 중 '일(one)'에 해당하는 객체 클래스를 테이블에 대응할 경우 '일 객체 클래스 이름 OID' 애트리뷰트를 첨가한다.

② 링크 애트리뷰트에 의해 대응되는 일대 다의 '다(many)'에 해당하는 객체 클래스는 일 객체 클래스와의 관련성 때문에 '일 객체 클래스 이름 OID' 애트뷰트를 첨가한다. 연관성을 나타내는 '링크' 애트리뷰트 또한 테이블 애트리뷰트로 첨가한다. 이 두 애트리뷰트의 결합을 후보 키로 한다.

(5) 여러개의 구성 클래스들(component classes)과 이들의 집합으로 이루어진 하나의 모임 클래스(assemblg class)사이에 집단화 관계는 연관화 관계의 특별한 경우이므로 (4)의 대응 법칙과 동일하다.

3.3.2 대응 법칙의 함수화

OMT 모델로 표현된 MSIB의 객체 클래스들은 계층 구조화 되어 있다. 객체 클래스를 관계 테이블로 대응시킬 때는 대응시킬 객체 클래스의 상위 레벨과의 관계성 뿐만 아니라 하위 레벨과의 관계성도 고려해야 한다. 따라서 대응 법칙을 두 번 적용해야 한다. 3.3.1의 대응 법칙을 함수로 정형화하고 정형화된 함수를 합성하여 합성 함수를 생성한다.

대응 법칙을 함수화 하기 위해 다음과 같은 수학적인 기호를 도입한다.

(1) 수학적 기호 정의

① $ID(N_K)$: 임의의 클래스 K에 대해 클래스 K의 이름을 사용한 ID 객체로 관계 테이블의 한 애트리뷰트

예) 객체 클래스 EDIM에 대해 ID(EDIM) = *EDIM OID*

② X_K : 임의의 객체 클래스 K, 이는 객체(인스턴스: instance) $x(x \in X_K)$들로 이루어진 집

③ $T(X_K)$: 임의의 객체 클래스 K에 대해 K의 모든 객체 x를 애트리뷰트로 하는 관계 테이블 단, X_K는 모든 객체 x가 테이블의 모든 애트리뷰트로 대응됨을 의미

④ C_K : 임의의 객체 클래스 K를 대응시킨 관계 테이블의 후보 키

⑤ + : 관계 테이블에서 애트리뷰트의 첨가

⑥ + AT(att) : 관계 테이블에서 'att'라는 애트리뷰트의 첨가

(2) 수학적 기호에서 유추된 식

수학적 정의에 의해 다음과 같은 자명한 식을 유추할 수 있다.

① $T(T(X_K)) = T(X_K)$

관계 테이블로의 대응을 두번 대응한 결과나 한번 대응한 결과가 동일하다. 왜냐면 두 경우 모두 동일한 애트리뷰트들로 이루어진 관계 테이블이 되기 때문이다.

② $T(ID(N_K)) = + ID(N_K)$

클래스 K의 이름을 사용한 ID 객체를 관계 테이블의 애트리뷰트로 대응한 결과와 $ID(N_K)$애트리뷰트를 첨가한 관계 테이블은 동일하다.

③ $T(AT(N_K)) = + AT(N_K)$

②의 식에서 $ID(N_K)$대신 $AT(N_K)$의 경우도 동일한 논리로 성립한다.

④ $T(T(X_K) + ID(N_K)) = T(T(X_K)) + T(ID(N_K))$

두번의 대응이 행해질 때는 배분 법칙이 성립한다.

(3) 대응 법칙의 함수화

다음과 같은 공통역과 치역을 정의한 후 대응 법칙을 함수로 표현할 수 있다.

정의역 : 그림 5의 OMT 다이어그램상에서 객체 클래스들의 집합 *OC*

$OC = \{X_K \mid K \in$ OMT 다이어그램의 객체 클래스$\}$

공변역 : OID와 필요 사항이 테이블의 애트리뷰트로 첨가된 관계 테이블의 집합을 *RT*라고 한다. T_{APP}는 애트리뷰트가 첨가된 관계 테이블이다.

$RT = \{T_{APP} \mid T_{APP} =$ OID 및 discriminator, link, qualifier, type 등 필요 사항을 애트리뷰트로 첨가한 관계 테이블$\}$

대응 법칙 (1), (2)를 다음과 같이 '단일 객체 클래스 대응 함수(single object class mapping function) s

: $OC \rightarrow RT$'로 정의한다.

$$\begin{cases} s(X_K) = T(X_K) + ID(N_K) & (5) \\ C_K = ID(N_K) & (5)' \end{cases}$$

대응 법칙 (1), (3)과 같이 객체 클래스간에 존재하는 일반화 관계를 고려한 대응 법칙을 '일반화 대응 함수(genaralization relationship mapping function) g : $OC \rightarrow RT$'로 정의하면 다음과 같다.

$$\begin{cases} g(X_{SP}) = T(X_{SP}) + ID(N_{SP}) + AT\ (disc) & (6) \\ C_{SP} = ID(N_{SP}) : \text{임의 슈퍼 클래스 SP 대응} & (6)' \\ g(X_K) = T(X_K) + ID(N_{SP}) & (7) \\ C_k = ID(N_{SP}) : \text{임의의 서브 클래스 K 대응} & (7)' \end{cases}$$

대응 법칙 (1)(4)와 (1)(5)는 각각 연관화 관계와 집단화 관계의 대응 법칙이다. 집단화 관계는 연관화 관계중 클래스간에 종속성이 존재하는 특별한 경우다. 따라서 대응 함수는 연관화 관계 함수를 집단화 관계 함수에도 적용한다.

객체 클래스간에 존재하는 연관화, 집단화 관계를 고려한 대응 법칙 '연관화 대응 함수(association relationship mapping function) a : $OC \rightarrow RT$'로 정의하면 다음과 같다.

$$\begin{cases} a(X_{AS}) = T(X_{ONE}) + ID(N_{ONE}) & (8) \\ C_{AS} = ID(N_{ONE}) : \text{일대 다 연관 관계 중 '일'에 해당하는 임의의 클래스 ONE 대응} & (8)' \\ a(X_K) = T(X_K) + ID(N_K) + ID(N_{ONE}) + AT(type) & (9) \\ C_K = ID(N_K) \text{ or } ID(N_{ONE}) + AT(type) & (9)' \\ \quad : \text{일대 다 연관 관계 중 임의의 '다' 관계 클래스 K 대응} & \end{cases}$$

(4) 합성 함수

객체 클래스의 관계 테이블로의 대응시 상위 객체 클래스와의 관련성과 하위 객체 클래스와의 관련성을 고려해야 한다. 다음과 같이 대응 함수 a와 g를 이용한 4가지 합성 함수 g∘a, g∘g, a∘g, a∘a를 정의할 수 있다. 이렇게 정의한 합성 함수를 그림 5의 OMT 다이어그램상의 A-①~E-⑥ 객체 클래스에 적용한다.

①g∘a

상위 객체 클래스와 연관화 관계가 있고 동시에 하위 객체 클래스와 일반화 관계가 있을때 이 객체 클래스는 합성 함수 g∘a에 의해 관계 테이블로 대응된다. 대응하고자 하는 객체 클래스는 K로 표시한다. 상위 객체 클래스는 일대 다 관계중에 일 객체 클래스이므로 ONE로 나타낸다. 하위 객체 클래스는 K 객체 클래스를 슈퍼 클래스로 하는 서브 클래스 SB로 표현한다.

합성 함수는 ONE 객체 클래스와의 a 관계를 대응한 후 SB 객체 클래스와 g 관계를 대응한다.

1단계 : 집단화 관계 대응

$g \circ a(X_K) = g(T(X_K) + ID(N_K) + AT(type))$ (식(9)에 의해)

$g(C_K) = g(ID(N_K))$ or $g(ID(N_{AS}) + AT(type))$ (식(9)'에 의해)

: 임의의 다 관계 클래스 K 대응

2단계 : 일반화 관계 대응

$$g(T(X_K) + ID(N_K) + ID(N_{ONE}) + AT(type))$$
$$= T(T(X_K) + ID(N_K) + ID(N_{ONE}) + AT(type)) + ID(N_K) + AT(disc) \quad (\text{식(6)에 의해})$$
$$= T(T(X_K)) + T(ID(N_K)) + T(ID(N_{ONE})) + T(AT(type)) + ID(N_K) + AT(disc) \quad (\text{식 (4)에 의해})$$
$$= T(X_K) + ID(N_K) + ID(N_{ONE}) + AT(type) + ID(N_K) + AT(disc) \quad (\text{식 (1)(2)(3)에 의해})$$
$$= T(X_K) + \underline{ID(N_K)} + \boxed{ID(N_{ONE}) + AT(type)} + AT(disc) \quad (ID(N_K)\text{는 중복})$$

$\therefore Ck = ID(N_K)$ or $ID(N_{ONE}) + AT(type)$

: 임의의 슈퍼 클래스 K 대응

1, 2 단계에 의해 g∘a 함수와 그에 따른 테이블의 후보 키는 다음과 같이 정의할 수 있다.

$$\begin{cases} g \circ a(X_K) = T(X_K) + \underline{ID(N_K)} + \boxed{ID(N_{ONE}) + AT(type)} + AT(disc) \\ Ck = \underline{ID(N_K)} \text{ or } \boxed{ID(N_{ONE}) + AT(type)} \end{cases}$$

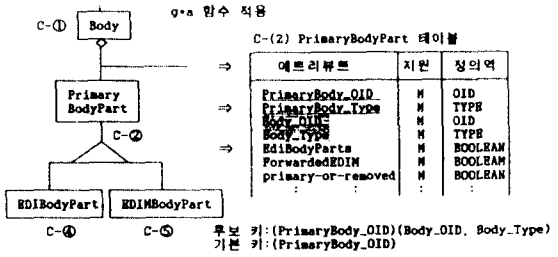

그림 7. C-② 객체 클래스에 g∘a 함수를 적용한 관계 테이블
Fig 7. Relational table that is a result of mapping g∘a function to C-② object class

C-② 객체 클래스에 g∘a 함수를 적용시켜 그림 7과 같은 테이블을 만든다. 각각의 객체 클래스를 구성하는 객체는 테이블의 애트리뷰트로 대응되며, 어떠한 객체의 대응인지를 나타내는 OID 애트리뷰트와 관계성 애트리뷰트가 첨가된다. 밑줄과 음영 부분은 수식이 관계 모델에서 어떤 애트리뷰트로 첨가되는지를 표시한다.

②g∘a

합성 함수 g∘g는 상위 객체 클래스 SP와 일반화 관계를, 하위 객체 클래스 SB와도 일반화 관계를 고려한 함수이다. ① g∘a의 경우와 동일하게 1, 2단계로 대응 함수를 합성한다. 다음과 같은 g∘g 합성 함수를 정의할 수 있다.

$$\begin{cases} g\circ a(X_K) = T(X_K) + \underline{ID(N_{SP})} + \boxed{ID(N_K) + AT(disc)} \\ \quad Ck = \underline{ID(N_{SP})} + \boxed{ID(N_K)} \end{cases}$$

A-④ 객체 클래스에 g∘g 함수를 적용시켜 그림 8과 같은 테이블을 생성한다.

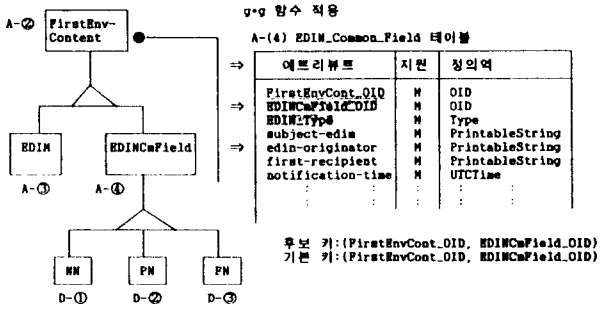

그림 8. A-④ 객체 클래스에 g∘g 함수를 적용한 관계 테이블

Fig 8. Relational table that is a result of mapping g∘g function to A-④ object class

a∘g와 a∘a의 경우는 ① g∘a, ② g∘g와 동일하게 함수의 합성에 의해 정의되며, 이를 OMT 다이어그램의 객체 클래스에 적용하여 관계 테이블을 생성한다.

3.4 관계 테이블의 RDBMS DDL 스키마 표현

설계 3 단계에서는 설계 2 단계의 관계 테이블 모델을 실제 구현하기 위해 선택한 RDBMS의 DDL로 표현한다. RDBMS를 위한 표준 언어인 SQL(Structured Query Language)로 MSIB의 테이블을 생성한다. SQL은 테이블의 이름, 테이블의 열인 애트리뷰트를 지정함으로써 테이블을 생성할 수 있다. 테이블 생성문의 일반적 형식에 따라서 그림 8의 (A)-4 EDIN_Common_Field 테이블을 생성하면 그림 9와 같다. 이와 같은 방법으로 MSIB를 구성하는 모든 테이블을 생성한다. 생성된 테이블을 기반으로 선택한 RDBMS를 이용하여 MSIB를 구축한다.

```
CREATE TABLE EDIN_Common_Field
(   FirstEnvContent_OID          ID              NOT NULL,
    EDINCmField_OID              ID              NOT NULL,
    EDIN_Type                    CHAR(10)        NOT NULL,
    subject-edim                 CHAR(20)        NOT NULL,
    edin-originator              CHAR(50)        NOT NULL,
    first-recipient              CHAR(50)        NOT NULL,
    notification-time            TIME            NOT NULL,
     :                :          :               :
     PRIMARY KEY (FirstEnvContent_OID, EDINCm_Field_OID),
     FOREIGN KEY (FirstEnvContent_OID)
     REFERENCES  FirstEnvContent_OID (FirstEnvContent))
```

그림 9. SQL 코드에 의한 EDIM 테이블 생성

Fig 9. Generating EDIM table using SQL code

Ⅳ. 객체-관계 대응 함수를 이용한 설계 방식 분석

기존의 설계 방법을 사용할 경우 반복적인 개념 설계와 논리 설계를 수행함으로 제 1 정규형을 만족하는 MSIB 구성 테이블을 구축하였다. 제안한 설계 방법은 객체-관계 대응 함수를 이용함으로써 별도의 정규화 과정 없이 제 3 정규형을 만족하는 MSIB 구성 테이블을 구축한다.

OMT 모델의 'Recursive Aggregates'을 사용하여 동일한 구조의 회송 EDIM 포함관계를 명시함으로써 회송 EDIM을 저장하기 위한 테이블의 중복 생성을 방지한다. 특히 다중 값을 저장하고 저장된 다중 값을 접근하는데 잇점을 제공한다.

4.1 대응 함수와 정규화

객체-관계 대응 함수를 MSIB 설계에 도입함으로써 별도의 정규화 과정 없이 MSIB 구성 테이블을 설계한다.

(1) 제 1 정규형 MSIB 관계 테이블

X.435 권고안에서 ASN.1으로 표현된 EDIM, EDIN 구성 요소 중 구조형으로 표현된 복합 객체가 있다. 구조형은 OMT에 의해 객체 클래스들 사이의 관계로 표현하였다. 즉, 구조형은 단순형의 값을 가지는 객체로 분해되고 관련된 객체가 클래스화 되어 표현된

다. 각 객체 클래스에 객체-관계 대응 함수를 적용하였다. 적용 결과와 관계 테이블 애트리뷰트는 ASN.1 단순형을 정의역으로 하는 애트리뷰트다. 그러므로 MSIB의 관계 테이블은 제 1 정규형을 만족한다.

(2) 제 2 정규형 MSIB 관계 테이블

제 3 정규형이 만족함을 보이기 위해 다음과 같은 관계를 정의한다.

① R : "객체-관계 대응 함수" s, a, g에 의해 생성된 관계 테이블에서 '기본 키→기본 키를 제외한 애트리뷰트들'의 관계

② $T(X_K)$-PK : "객체-관계 대응 함수" s, a, g에 의해 생성된 관계 테이블에서 기본 키를 제외한 애트리뷰트들의 집합

s, g, a와 합성 함수 g∘g, g∘a, a∘g, a∘a에 의해 생성된 관계 테이블의 후보 키의 유형을 표 2와 같이 분류할 수 있다. 제 2 정규형은 기본 키와 기본 키를 제외한 애트리뷰트 모두가 기본 키에 완전 함수적 종속임을 보이면 된다.[18] 그런데 완전 함수적 종속의 문제는 기본 키가 복합 애트리뷰트일 때만 발생한다. 표 2에서 4의 경우의 기본 키가 복합 애트리뷰트인 g∘g, a∘g 함수를 적용시킨 관계 테이블의 경우만을 증명하면 된다.

표 2. 함수와 후보 키
Table 2. Function and candidate key

1	s	$C_K = ID(N_K)$
2	g	$C_{SP} = ID(N_{SP})$ Ck = $ID(N_{SP})$
3	a	$C_{AS} = ID(N_{AS})$
4	g∘g a∘a	$Ck = ID(N_{SP}) + ID(N_K)$
5	g∘a a∘a	$Ck = ID(N_K)$ or $ID(N_{AS}) + AT(type)$

g∘g, a∘g 함수를 적용시켜 생성한 테이블의 기본 키는 복합 애트리뷰트인 $\langle ID(N_{SP}), ID(N_K)\rangle$이다. 관계 R에서 $\langle ID(N_{SP}), ID(N_K)\rangle \rightarrow T(X_K)-PK$이며, 이때의 관계 R이 완전 함수적 종속임을 보이면 된다.

$\langle ID(N_{SP}), ID(N_K)\rangle \xrightarrow{R} T(X_K)-PK$가 성립함을 증명하면 다음과 같다.

만일 $ID(N_{SP}) \rightarrow T(X_K)-PK$가 관계 R에서 성립한다고 하자. $T(X_K)-PK$에 속하는 임의의 한 애트리뷰트를 a_K라 하자. 그러면 $ID(N_{SP}) \xrightarrow{R} a_k$ 또한 성립해야 한다. 그러나 K 객체 클래스에 의해 생성된 관계 테이블의 한 애트리뷰트인 a_k는 슈퍼 클래스를 나타내는 애트리뷰트 $ID(N_{SP})$만으로는 K 객체 클래스의 애트리뷰트 임을 알 수 없다. $ID(N_{SP}) \not\xrightarrow{R} a_k$다. 그러므로 관계 R을 위해서는 $ID(N_K)$ 애트리뷰트를 첨가한 복합 애트리뷰트가 필요하다.

위와 동일한 방법으로 $ID(N_K) \xrightarrow{R} T(X_K)-PK$의 경우는 $ID(N_K)$만으로는 슈퍼클래스와의 관계성에 의한 생성을 표현할 수 없다. 이를 위해 $ID(N_{SP})$ 애트리뷰트를 첨가한 복합 애트리뷰트를 기본 키로 해야 한다. 그러므로 $\langle ID(N_{SP}), ID(N_K)\rangle \xrightarrow{R} T(X_K)-PK$로의 함수 R은 완전 함수적 종속이다.

(3) 제 3 정규형 MSIB 관계 테이블

제 3 정규형은 관계 R이 제 2 정규형이며 기본 키와 기본 키를 제외한 애트리뷰트 모두가 기본 키에 이행적 함수 종속이 아닐 때 성립한다. 이행적 함수 종속이란 함수 종속 관계 R.A→R.B와 R.B→R.C가 성립되면 논리적 결과로 R.A→R.C가 성립할 때를 의미한다.

대응 함수를 적용한 관계 테이블의 관계중 R.B→R.C가 존재하지 않는다면 이행적 함수 종속도 존재하지 않는다.

$T(X_K)$테이블의 임의의 애트리뷰트를 b_K를 제외한 임의의 애트리뷰트를 c_K라 하자. $b_K \xrightarrow{R} c_K$가 성립하지 않는다. 왜냐하면 대응 함수에 의하여 테이블의 애트리뷰트들은 권고안에서 제시한 필드들을 테이블의 애트리뷰트로 대응시켰다. 권고안의 각 필드들은 각각이 필요한 정보를 가지며 이 필드들은 의미상 종속성을 가지지 않는다. 임의의 애트리뷰트 b_K, c_K간에는 종속성이 존재하지 않는다. 이는 이행적 종속의 성립을 위한 관계 R.B→R.C가 존재하지 않음을 의미하므로 MSIB의 관계 테이블들은 제 3 종규형을 만족한다.

4.2 재귀적 구조의 EDIM 처리를 위한 관계 테이블

EDIM의 몸체 부분은 머리 부분과 같이 정형화된 크기로 고정된 것이 아니라 상황에 따라서 가변 크기를 가진다. 몸체 부분은 EDI 표준 문서에 따라 작성한 EDI 인터체인지를 포함하거나 다른 곳으로 전송되기 위한 회송 EDIM을 포함한다. 그림 5의 다이어그램과 그림 2의 EDIM의 MS로의 대응에서 볼 수 있듯이 회송 EDIM일 경우는 이를 포함하는 EDIM과 동일한 구조인 재귀적(recursive) 구조가 된다.

이는 OMT 모델의 'Recursive Aggregates'을 사용하여 정확하게 표현된다. EDIM의 몸체 부분에 포함되는 회송 EDIM이 EDI-MS에 애트리뷰트별로 저장될 때 회송 EDIM을 위한 별도의 테이블을 만들 필요가 없다. E-(1)~E-(6) 객체 클래스는 각각 객체 클래스 A-(2), A-(3), B-(1), C-(1)~C-(5)와 대응되므로 객체 클래스 A-(2), A-(3), B-(1), C-(1)~C-(5)에 "객체-관계 대응 함수(Object-Relation mapping function)"를 적용시켜서 만든 관계 테이블을 이용하여 회송 EDIM의 정보를 저장한다.

4.3 대응 함수와 다중 값 처리

객체-관계 대응 함수를 이용함으로써 얻을 수 있는 가장 큰 장점은 다중 값을 저장하고 저장된 다중 값을 접근하는데 잇점을 제공하는 것이다. 다중 값에 대한 처리가 특히 중시되는 부분은 EDIM이 다중 수신자에게 전송되는 경우다.

EDI-MS는 MHS용 MS에 비해 통지와 책임에 관한 사항이 추가되었다. EDI-UA는 수신하고자 하는 메시지를 EDI-MS에서 제공하는 서비스를 이용하여 자신은 EDI 메시징 환경에 종속되지 않고 배달되는 메시지에 대한 책임과 통지를 설정할 수 있다. 이때 EDI-MS는 MSIB에 저장된 다중 수신자 필드를 체크(check)하여 이를 기준으로 자동 회송, 책임 통지를 생성한다. 따라서 MSIB는 다중 수신자에 관해 권고안에서 제시하는 항목들을 구조적으로 저장해야 한다.

4.3.1 다중 수신자 처리 시스템

X.435 권고안에서는 EDIM 머리 부분의 수신자 필드를 동일한 구조의 반복을 의미하는 ASN.1의 SET OF로 표현하였다. SET OF에 의해 표현된 수신자 필드는 다중 수신자에 대한 사항을 기술할 수 있는 필드들로 구성되어 메시지의 수신자가 여럿인 경우를 처리한다.

기존의 방법에 의해 MSIB의 수신자 테이블을 구성하였을 경우와 객체-관계 대응 함수를 이용하여 테이블을 구성하였을 경우에 대한 통지 생성 처리과정에서 차이가 발생함을 알 수 있다.

다음 알고리즘은 EDI-MS로 배달된 EDIM 머리 다음 알고리즘은 EDI-MS로 배달된 EDIM 머리 부분의 다중 수신자 필드를 처리하는 알고리즘이다. 예로 그림 10의 테이블에 각 항목별로 저장된다. 이 정보를 이용하여 다중 수신자에 대한 통지를 생성하며, 회송을 처리하는 알고리즘이다.

알고리즘

```
construct_MSIB_table();        /* these tables are constructed
                                  by the OTMF design methodology */
do{
   get(O/R_OID);
   check(recipients, O/R_table)
   if(recipients_field == Single_value)
       access_record(1, O/R_OID);  /* access only one record of
                  the same O/R_OID in recipient_info_table */
   if(recipinets_field == Multi_value)
       access_record(N, O/R_OID);        /* access n records of
                  the same O/R_OID in recipient_info_table */
   check(edim_requests_field);
   make_edim(request);  /* generate PN or NN or FN */
   get(recipient, recipient_info_table);
   check(recipient, profile);     /* Is the recipient EDI-UA of
                          the EDIM connected to the EDI-MS? */
   if(recipient != profile)
        forwarding_action(standard_type, Per_Recipient_OID)
                                  /*generate forwarding message */
}while(송/수신자 테이블의 입력이 끝날 때까지)
```

그림 10은 객체-관계 대응 함수를 적용하여 얻은 다중 수신자 처리 테이블의 연결 구조를 보여준다. 위의 알고리즘을 이용하여 테이블을 제어한다.

기존의 방법은 다중 값이 있는 경우 driven table을 이용하여 다중 값 저장 테이블에 접근하였다. 기존의 방법에 의해 다중 값을 처리할 경우에는 저장된 메시지 하나 하나에 대한 처리를 실시할 때마다 애트리뷰트 값 저장 테이블과 driven table을 접근해야 한다.

그림 11은 제안한 방법과 기존 방법을 이용하여 메시지 수신 통지를 생성할 때 필요한 연산 횟수의 실험 결과에 대한 비교 그래프다. 기존의 방법은 다중 값 처리를 위해 별도의 테이블 접근, 고유 인덱스 탐색 등의 과정이 필요하다. 반면 제안한 방법은 테이블의 각 튜플로 직접 접근 가능하도록 OID 애트리뷰트에 튜플이 저장된 위치를 저장한다. 따라서 이를 이용함으로써 원하는 튜플로의 직접 접근이 가능하다.

제안한 방법을 사용할 경우는 메시지 증가에 따른 연산 횟수가 산술적으로 증가한다. 반면 기존의 방법은 연산 횟수가 기하 급수적으로 증가한다.

계산에 의해 연산 횟수를 구하면 다음과 같다. 제안한 방법은 OID 애트리뷰트에 저장된 값을 이용한다. OID 애트리뷰트에 저장된 값을 이용하여 원하는 정보가 저장된 테이블의 튜플로 직접 접근한다. 따라

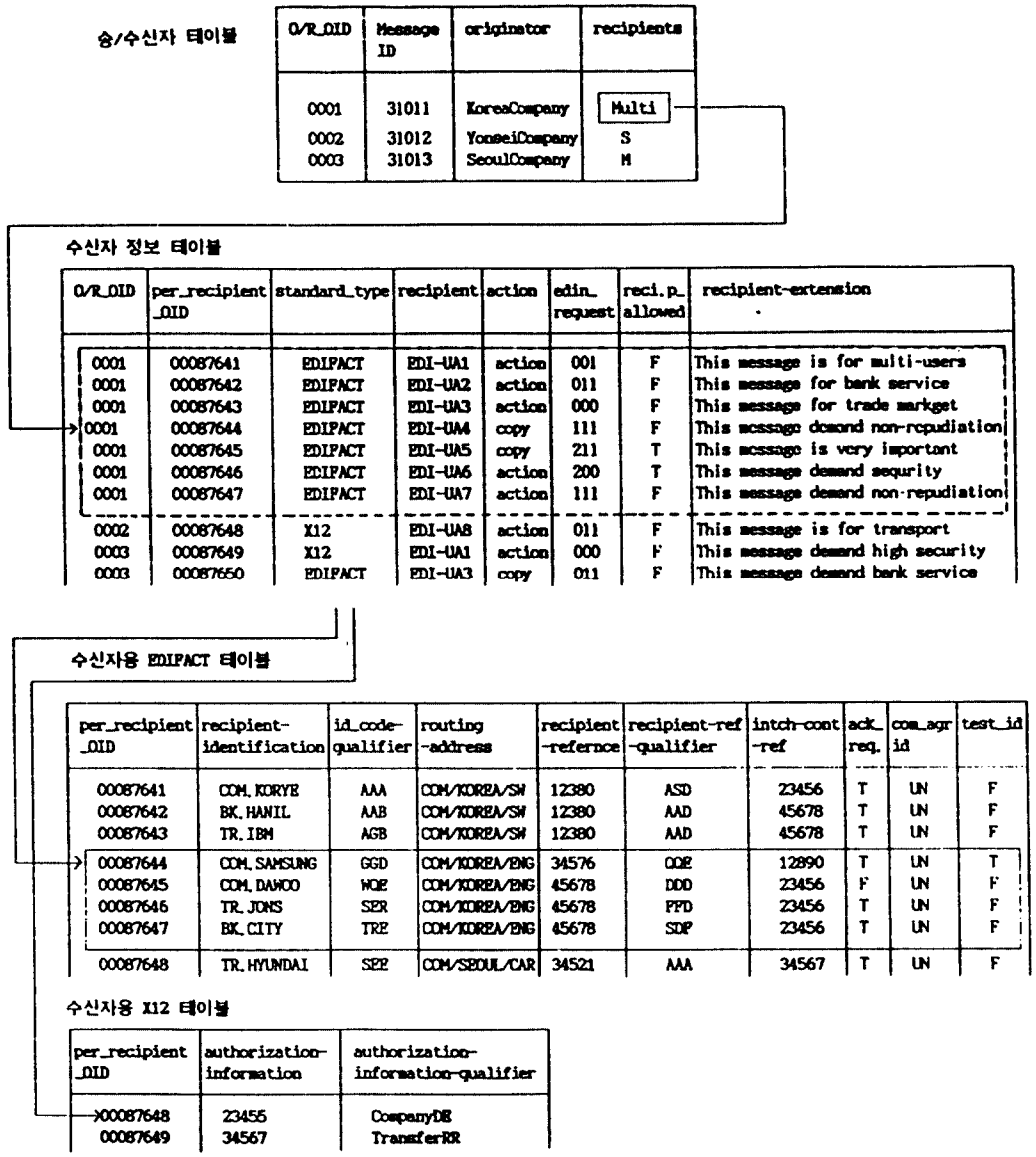

그림 10. 객체-관계 대응 함수를 이용하여 설계한 다중 수신자 처리 테이블 연결 구조

Fig 10. The connection mechanism of the processing tables for multi users designed by Object-Relation Mapping Function

서 통지 생성을 위한 연산 횟수는 다음과 같다. C는 튜플로의 직접 접근을 위한 계산 시간으로 일정한 상수다.

$$\sum_{i=1}^{n} C = Cn \qquad : O(n)$$

기존의 방법은 수신된 메시지 고유의 인덱스(msib_gen_id)를 저장 테이블 마다 보유하고 있다. 원하는 정보가 저장된 테이블의 튜플로 접근하기 위해서는 테이블이 보유하는 메시지 고유 인덱스를 탐색해야 한다. 탐색하여 처리하고자 하는 메시지의 번호와 테이블의 메시지 고유 인덱스가 일치하면 튜플의 값을

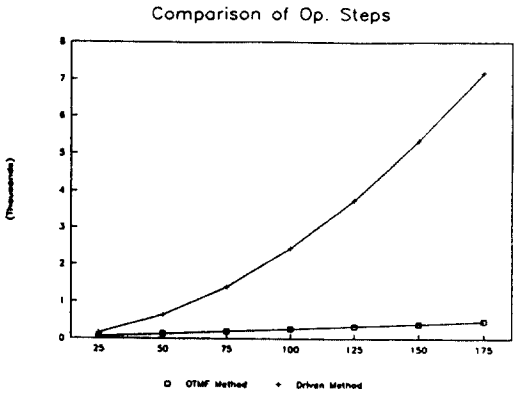

그림 11. 메시지 갯수에 따른 통지 생성을 위한 연산 횟수 비교

Fig 11. The comparison of the atomic operation for generating of the EDI Notification according of the number of message

얻을 수 있다. 따라서 필요한 연산 횟수는 다음과 같다.

$$\sum_{k=1}^{n} \frac{n}{2} = \frac{n_2}{2} \qquad : O(n^2)$$

계산에 의해 구한 연산 횟수는 그래프 형태와 대응됨을 알 수 있다.

4.4 설계 방법 분석

표 3에서 기존의 설계 방법을 이용하여 MSIB를 설계, 구축할 경우와 제안한 방법을 이용하여 설계, 구축할 경우 설계시 얻을 수 있는 장점과 설계의 효과로 얻을 수 있는 장점을 비교하였다.

기존의 설계 방법을 이용하여 EDI-MS용 MSIB를 구성할 경우와 제안한 설계 방법을 이용할 경우의 구성 테이블 갯수는 별 차이가 없다. 그러나 기존방법은 다중 값 처리를 위한 별도의 테이블이 필요하다. 제안한 설계 방법은 중첩 구조의 회송 EDIM을 MSIB 엔트리로 저장할 수 있도록 설계시 이를 고려하였다.

기존의 방법은 다중 값이 있는 경우 driven table을 이용하였으므로 테이블 접근 횟수, driven table을 위한 공간 낭비 등의 단점이 있다. 본 연구에서 제안한 방법은 별도의 driven table 없이도 기존 테이블에 OID 필드와 관계성 필드를 추가함으로 다중 값 처리를 위한 테이블들간에 가상적 링크(link)가 구성된다. 이는 기존 방법의 단점을 보안하면서 다중 값에 대한 접근 및 검색을 용이하게 한다. 또 전체 MSIB

표 3. 설계 방법 비교표

Table 3. The comparison table of the designe method

비교 기준	기존 방법	연구 방법
설계 과정	정규화를 위한 논리 설계, 개념 설계 반복	객체 관계 대응 함수 적용
정규형	제 1 정규형 만족	제 3 정규형 만족
필드 추가	메시지 고유 식별자, 다중값 저장 테이블의 자체 식별자	관계 테이블 이름을 사용한 OID 필드 추가, 관련성 필드 추가
다중 값을 갖는 애트리뷰트 처리를 위한 별도의테이블	2개	필요 없음
회송 EDIM 저장	고려 안함	고려
다중 값 처리를 위한 애트리뷰트 접근시 연산 횟수	$O(n^2)$	$O(n)$

엔트리를 구성하는 테이블간에 필요한 조인 연산(join operation)을 감소 시킨다. RDBMS에서 다른 연산에 비해 오버헤드가 큰 조인 연산을 감소 시킴으로 EDI-MS는 MSIB를 이용하여 수행하는 서비스를 빠르게 행할 수 있다.

V. 결 론

본 논문에서는 EDI-MS의 하부 모듈로 사용되는 MSIB 저장 테이블 구축시 효율적인 저장과 검색을 위해 모델링 단계에서 객체 지향 패러다임을 도입하는 설계 방법을 제안하였다. 전반적인 EDI 통신 시스템에 관해 연구하였고 EDI 전용 통신 프로토콜인 CCITT X.435를 연구 분석하였다. 특히 상용 RDBMS로 개발하는 독립형 EDI-MS에 연구 촛점을 맞추어 EDI-MS를 지원하는 MSIB의 역할과 기능을 연구하였다.

X.435는 1988년도의 X.400 프로토콜 스택을 기본으로 한다. 따라서 본 논문에서는 기존의 MHS-MS의 EDI-MS로의 확장성과 EDI MS의 MHS-MS의 수용성을 고려하였다. 기존의 MHS-MS의 하부 저장 모듈로 RDBMS를 이용하는 추세를 고려하여 EDI-MS용 MSIB 또한 상용 RDBMS를 이용하여 개발한다.

EDI-MS의 MSIB를 구축함에 있어서는 ASN.1으로 표현된 EDI 메시지의 빠른 검색을 위해 MSIB를 여러개의 테이블로 구조화하여 구축해야 한다. 그리고 EDI 시스템은 그 특성상 영리를 목적으로 하는 기업이 사용한다는 현실적인 특성을 고려해야 한다.

본 논문은 기존의 OMT 모델의 RDB 스키마로의 변환 기법을 확장하여 객체-관계 대응 함수를 이용한 새로운 변환 기법을 제안하였다. 이를 EDI-MS용 MSIB 설계의 논리적 단계와 개념적 설계 단계 중간에 도입함으로 MSIB 저장 객체인 EDIM의 특성을 충분히 고려한 MSIB를 설계하도록 한다.

객체-관계 대응 함수는 OMT 다이어그램의 객체 클래스들 간의 계층적 관계에서 발생하는 상·하위 클래스의 관계성을 관계 테이블의 애트리뷰트로 보유할 수 있도록 한다. 객체-관계 대응 함수는 기존의 OMT 모델의 RDB 스킴으로의 대응시 보장하였던 정규화 과정의 잇점 또는 보장한다.

EDI-MS용 MSIB 설계에 이 함수를 도입함으로 EDIM의 계층적 구조와 ASN.1으로 표현되는 구조형의 복합 애트리뷰트를 ECR 모델을 이용한 RDB 스키마로의 변환시 발생하는 정규화 문제를 해결하고 효과적인 다중 값 처리를 가능하게 한다.

추후 연구 과제로는 본 논문에서 제안한 설계 방법을 이용하여 기존의 상용 RDBMS를 사용하여 구축된 MHS-MS의 MSIB를 확장, 구현하는 문제다. 더 나아가서는 구현된 MSIB를 EDI-MS의 저장 모듈로 하여 EDI 서비스를 행할 수 있는 EDI-MS를 개발하는 문제다.

참 고 문 헌

1. Adrian Tang, Sophia Scoggins, Open Networking with OSI, Prentice-Hall, Inc., pp.231-254, 1992.
2. Barbara Nelson, X.400 Message Handling Systems : Standards and Practice, Retix, INTEROP 92 FALL.
3. CCITT, CCITT Recommendations X.400-X430, Data Communication Networks, Messaging Handling System, 1988.
4. CCITT, CCITT Recommendations X.435, Communication Networks, EDI Messaging System, 1992.
5. Fredric Alexandre, "Realizing The Message Store with a Relational Database," IFIP, 1990, pp E3.
6. James Martin, James J. Odell, Object-Oriented Analysis & Design, Prentice-Hall, Inc., pp. 435-439, 1992.
7. James Rumbaugh, Michael Blaha, William Premerlani, Frederick Eddy, William Lorensen, Object-Oriented Modeling and Design, Prentice-Hall International, Inc., pp.386, 1991.
8. Hawryszkiewycz I. T, Prentice-Hall, Inc., Relational Database Design, pp.160-166, 1990.
9. Marshall T. Rose, The Open Book, A Practical Perspective on OSI, Prentice-Hall, Inc., pp 171-191, 1992.
10. Michael R, Blaha, William J, Premerlani and James E.Rumbaugh, "Relational Database Design Using an Object-Oriented Methodology," Communication of the ACM, vol.31, No.4, pp.414-427, April, 1988.
11. Pinter, Set Theory, Addison-Wesley, pp. 51-69, 1986.
12. Platterner B., Lanz C., Lubich H., Muller M., Walter T., X.400 Message Handling Standards, Interworking, Applications, Addison-Wesley, pp.476-480, 1991.
13. Richard Hill, EDI and X.400 Using Pedi, Technology Appraisals, pp.22-28, pp.31-34, 1990.
14. Rolland F.D., Relational Database Management with ORACLE, Addison-Wesley, pp. 223-227, 1992.
15. 김진호, 최황규, "EDI 시스템에서 메시지 저장을 위한 모델링 기법," pp.111-135 EDI 시스템 기술 워크샵 '92, 1992.
16. 김태윤, 기업간 정보통신 전자거래정보교환, 집문당, pp.16-17, pp.37-42, 1991.
17. 이운용, 원병성, 최명수, 한상학 "관계형데이타베이스 시스템을 이용한 EDI 메시지 저장시 다중값을 갖는 애트리뷰트처리에 관한 연구," '92 가을 학술발표논문집, pp.47-54, 정보과학회, 1992.
18. 이석호, 데이타 베이스론, 정익사, pp.272, pp. 276, pp.279, pp.279-280, 1992.
19. 진영민, "전자정보거래(EDI) 서비스(VI)," 한국통신 경영과 기술, pp.32, 1991.5.
20. 한국통신, "KT-EDI 시스템 연구개발," '91년도 상반기 연구 결과 발표회 보고서, pp.52-55, 1991.9.
21. 한국통신, "EDI응용시스템 연구개발," 중간보고서, pp.50-53, 1992. 12.

白 勝 惠(Seung Hye Paik) 正會員
1968年 3月 22日生
1991年 2月 : 고려대학교 수학교육과 졸업
1994年 2月 : 고려대학교 전산과학과 석사
※주관심분야 : EDI, 객체 지향 시스템, 데이타베이스

金 泰 潤(Tai Yun Kim) 正會員
1956年 7月 13日生
1981年 2月 : 고려대학교 산업공학과 졸업
1983年 2月 : 미국 Wayne State University(공학석사)
1987年 12月 : 미국 Auburn University(공학박사)
1988年 3月 ~ 현재 : 고려대학교 전산과학과 부교수
※ 주관심분야 : EDI, ISDN, 위성통신, 이동통신, 통신 보안, 컴퓨터 그래픽스