# XML기반의 EDI 문서교환을 위한 미들웨어 설계 및 구현

최 광 미† · 박 수 영† · 정 채 영††


## 요 약

EDI를 이용한 전자문서 처리는 별도의 전용 소프트웨어와 부가가치망을 이용해 문서를 교환하게 되는데, 전용 소프트웨어의 사용은 새로운 문서에 대한 변경이 필요하며, 부가가치망의 사용은 문서교환과 유지보수에 많은 비용이 소요된다. 이러한 문제점 때문에 VAN 기반의 기존 EDI가 웹기반의 EDI로 옮겨지고 있다. 본 논문에서는 JDBC 브리지를 이용하여 두개의 관계형 데이터베이스에 존재하는 EDI 메시지를 XML로 변환하는 기법을 제안한다. 변환된 XML 파일을 사용하여 스키마를 그대로 복구하는 동시에 정의된 테이블에 원래 레코드를 그대로 삽입하는 방법을 제시하였다. 이는 동일한 데이터베이스 관리시스템을 사용해야 한다는 전제조건을 필요로했던 기존방식을 탈피했으며, 전자문서 교환시 환경에 따라 정상적으로 동작하지 않았던 경우의 문제점을 극복했다.


# Design and Implementation of Middleware supporting translation of EDI using XML

Gwang-Mi Choi† · Su-Young Park† · Chai-Yeoung Jung††


## ABSTRACT

Electronic document processing using EDI (Electronic Data Interchange) must exchange documents using VAN (Value Added Network). However, the use of exclusive software needs alteration of a new document and the use of VAN (Value Added Network) demands an exchange of document and high cost for maintenance. Due to these problems, the existing EDI (Electronic Data Interchange) is turning into Web-based EDI (Electronic Data Interchange). This paper suggests techniques that change EDI (Electronic Data Interchange) messages which exist in two relational databases into XML (extensible Markeup Language) using the JDBC bridge. Also this paper proposes a method that recovers schema using converted XML (extensible Markeup Language) file, and a method which inserts an original record into a declared table. This solves the limitation of an original method that have to use same database management system and also overcomes the problem in certain circumstances where the EDI (Electronic Data Interchange) exchange does not work.


**키워드 : XML(extensible Markeup Language), EDI : 전자자료교환(Electronic Data Interchance), DTD(Document Type Definition)**

## 1. 서 론

인터넷의 확산과 더불어 초고속망의 이용이 활발히 진행되고 있으며, 이에 따라 정보통신망을 이용하여 정보를 저렴하고 편리하게 교환하며 업무를 효율적으로 처리하기 위한 방안을 모색하고 표준화하려는 노력을 하고 있다. 이러한 노력이 결실을 맺기 위해 우선 전자문서의 표준화 작업이 필요하며, 따라서 각 표준기구가 제시한 표준안이 꾸준히 제안되고 있다[1]. 이러한 노력의 결과 가장 일반적으로 쓰이는 대표적인 형태가 바로 EDI(Electronic Data Interchange)이다. 현재의 EDI 서비스는 송신자, 수신자, VAN(Value-Added Network)간의 교환할 문서를 미리 정의하여 등록시켜 둔 EDI 전용 소프트웨어를 이용하여 전자문서를 주고받는다. 이러한 환경에서 EDI 서비스는 새로운 형태의 문서를 교환해야 할 경우 사용자마다 새로운 문서에 정보를 등록하여 EDI 전용 소프트웨어의 변경을 필요로 한다. 그러므로 끊임없이 변화하는 다양한 사용자의 욕구를 만족시키기에는 동적 기능이 부족한 것이 EDI 서비스 환경이다.

또한, 서로다른 기업간의 데이터를 전달하기 위해 VAN을 이용하였으나 VAN의 폐쇄성으로 인해 확장에 문제점이 제시되었다. 이러한 문제점을 해결하기 위해 다양한 문서구조 표현이 가능한 XML(eXtensible Markup Language)을 활용한 EDI 서비스 환경으로 옮겨가고 있다[2].

기존의 EDI 서비스가 인터넷 플랫폼으로 옮겨가고 다시 웹 플랫폼으로 옮겨감으로써 EDI 서비스 사용자는 기존의 서로 다른 EDI 전용 소프트웨어의 사용에 따른 불편함이 없어지고 통합된 환경으로 EDI 서비스를 사용할 수 있다. 또한 시스템 개발 및 유지보수가 기존의 복잡하고 확장성이 없는 EDI 전용 시스템보다 저렴해지는 장점이 있다. 또, 문

† 준 회 원 : 조선대학교 대학원 전산통계학과
†† 종신회원 : 조선대학교 전산통계학과 교수
논문접수 : 2002년 2월 8일, 심사완료 : 2002년 8월 26일

서를 구성하는 각 요소를 객체단위로 처리할 수 있으며, 손쉬운 시스템의 확장성을 제공할 수 도 있다.

단점을 다수 내포하고 있기는 하지만 EDI를 이용함으로써 얻을 수 있는 효과는 상당히 크기 때문에 많은 선진 기업들이 이를 도입하여 사용하고 있다. 대표적인 효과로는 처리시간 단축, 업무부대비용 감소, 업무오류 방지, 물류비 절감, 이미지개선, 인력절감 등을 들 수 있다. 그러나, 이러한 EDI의 효과에 비해 현재 EDI를 활용하는 비율은 낮은데 그 대표적인 이유는 기간업무를 처리하는 기존의 내부 응용 소프트웨어와 연계가 어렵고, EDI 소프트웨어가 갖는 폐쇄성 및 법령과 제도의 미비 등인 것으로 조사되었다. 소규모 기업의 경우 기업 규모에 비해 과다한 도입비용과 운영비용 역시 EDI 확산에 걸림돌이 되고 있다.

이에 따라 본 논문에서는 폐쇄적인 VAN을 이용하지 않고 기존에 구축되어 있는 인터넷을 기반으로 구조적이고 확장성이 뛰어난 인터넷 표준 언어인 XML을 기반으로 하여 EDI 문서를 처리하기 위한 미들웨어를 설계 및 구현하고자 한다.

본 논문의 연구목적은 XML기반 EDI 문서교환 미들웨어를 설계하고 구현하는 것이다. 기존 EDI의 문제점인 고정적이고 정형화된 데이터를 JDBC 변환기를 사용하여 관계형 데이터베이스 관리 시스템에서 스키마와 스키마에 의해 만들어지는 테이블에 포함된 레코드를 XML 파일로 변환한다. 네트워크를 통해 변환된 XML 파일을 사용하여 원래 스키마로 그대로 복구하는 동시에 이 스키마에 의해 정의된 테이블에 원래의 레코드를 그대로 삽입하는 기능에 대하여 구현하고자 한다.

## 2. 관련 연구

### 2.1 전통적인 EDI와 웹EDI

EDI는 조직간 시스템의 대표적 형태로 기계가 직접 읽고 처리할 수 있는 정형화된 문서에 대하여 자료의 내용을 표현하는 기호(Data Code 또는 Semantic) 및 자료의 항목별 표준 배열순서(Data Format 또는 Syntax)에 의해 표준화된 형태로 전자문서 통신매체를 통하여 교환하는 방식을 말한다. 전통적인 EDI 시스템의 구성은 크게 메시지, 통신 그리고 메시지를 처리하는 소프트웨어의 3부분으로 구분된다[1].

대부분의 메시지는 표준화된 양식과 구문을 따르게되며, 다양한 영역에서 사용되는 양식과 구문에 대한규정의 전자문서 표준안인 UN/EDIFACT, ANSI X.12, KEDIFACT 등의 국제 표준이 있고, 전문회사의 경우 자사와 거래처간에 문서교환을 위해 자체적으로 구현한 독자적인 양식을 사용하는 경우도 있다. EDI 메시지는 세그먼트(Segment)와 데이터 엘리먼트(Data Element)로 구성된다[4].

EDI에서 사용되는 통신 네트워크는 크게 2가지로 구분되는데 거래자를 직접 연결시키는 점대점(Point-to-Point) 연결방식과 중간의 통신서비스 업체인 VAN(Value Added Network)를 매개로 하는 제 3자 연결방식이 있다. 통신 프로토콜은 CCITT(Consultative Committee for International Telegraph and Telephone)에서 정의한 X.400, X.430, X.500 등과 같은 프로토콜이 있다[6].

EDI 메시지를 처리하는 소프트웨어는 사용자가 EDI를 이용한 작업처리를 할 수 있도록 사용자 인터페이스를 지원하는 부분, 실제 메시지를 처리하기 위한 부분, 그리고 하부구조로서 통신 소프트웨어로 구분된다. 사용자 인터페이스 지원 소프트웨어는 사용자가 표준전자문서를 상대방에게 송신하기 위하여 문서를 작성, 수정, 조회, 삭제하고 수신된 표준전자문서를 조회, 출력 할 수 있는 기능을 가지고 있다. 메시지 처리용 변환 소프트웨어는 전송될 내부파일 형태의 전자문서를 표준형태의 EDI 내부파일 형태로 바꾸어 주는 기능을 한다. 통신 소프트웨어는 컴퓨터와 컴퓨터간의 데이터 교환시 사용되는 소프트웨어로 통신회선의 종류에 따라 통신이 가능하도록 지원하는 기능을 가지고 있다[7].

EDI는 보통 고정된 태그집합으로 이루어지므로 전송되는 문서가 여러 형태의 데이터 필드를 포함하고 있는데, 기존 EDI는 이러한 필드를 추가하거나 삭제할 수 있는 유연성을 갖지 못한다. 즉, 고정된 태그집합은 새로운 제품과 서비스의 추가, 컴퓨터 시스템의 교체, 업무 처리과정 개선에 부정적인 영향을 주게 된다. 또한 VAN 중심인 EDI 서비스는 전송 메시지에 대해 일괄처리 및 압축전송 방식을 사용하기 때문에 전송속도의 증가요구, 실시간 응답이 필요한 업무에서는 요구조건을 모두 충족시킬 수 없다. 따라서, 기업에게는 빠른 기업환경 변화에 적응력 부족을 가져오고, 기업 업무환경 개선에 따른 타정보 시스템과의 연동을 어렵게 한다.

웹 환경에서 EDI 서비스를 구현하기 위한 방안은 변환 소프트웨어를 서버에 둘것인지 아니면 클라이언트에 둘것인지에 따라 두 가지 방법으로 구분할 수 있다. 첫 번째는 사용자가 웹 브라우저를 이용하여 양식의 내용을 입력한 다음 서버로 보내면 서버 내에서 변환과정을 거쳐 EDI 문서가 처리된다. 다운로드된 클라이언트 프로그램이 문서를 변환하여 그 내용을 서버로 송신하고 서버에서 후속 처리가 이루어진다. 따라서, 기존 EDI가 중요시하고 있는 데이터 처리의 개념에서 크게 벗어나지 못함으로서 EDI의 단점을 안고 있다[3].

### 2.2 미들웨어

미들웨어는 클라이언트 프로그램과 서버 프로그램 사이에서 클라이언트와 서버간에 연결을 유지/관리하며, 클라이언트의 작업 처리 요구를 서버에 전달하는 일을 하는 소프트웨어이다. 미들웨어가 정상적으로 수행되려면 클라이언트 측과 서버 측 미들웨어가 별도로 존재해야하며, 교환되는 자료는 반드시 각각의 미들웨어를 거쳐야 한다. 또한 클라이언트와 서버간에 통신이 가능하도록 데이터 통로를 제공하고, 양측간의 연결 세션을 유지/관리하는 기능을 하며, 클라이언트

의 작업 처리에 필요한 서비스를 찾아주는 기능을 한다.

또다른 미들웨어의 기능으로는 여러 서버에 흩어진 프로그램에 클라이언트 요청을 라우팅하며, 서버 프로그램이 실행중이면 클라이언트 요청을 기다리게 하고, 서버 프로그램을 감시하는 기능을 하며, 데이터베이스 트랜잭션을 관리하고, 데이터베이스와 공조하는 기능을 한다.

3계층 클라이언트/서버 구조에서는 미들웨어가 중간 계층을 형성한다. 이러한 미들웨어에는 TP monitor(Transaction Processing Monitor), DCE environment(Distributed Computing Environment or Data Communication Equipment), RPC system, ORB(Object Request Broker)가 있다.

#### 2.2.1 TP monitor(Transaction Processing Monitor)

TP monitor는 트랜잭션이 프로세스 내의 한 단계에서 다음단계로 정상적으로 진행되는지를 감시하는 프로그램이다. 즉, 트랜잭션 처리의 완전성을 보장하고, 오류가 발생하면 적절한 조치를 취하는 역할도 가지고 있다. 실제로 많은 TP monitor가 부하를 분산시키고 트랜잭션의 효율적 전송을 담당한다.

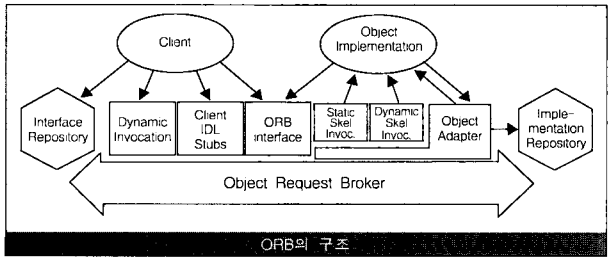
#### 2.2.2 DCE environments(Distributed Computing Environment ; or Data Communication equipment)

분산 컴퓨팅 환경(DCE : Distributed Computing Environment)은 시스템 내에서 데이터 교환 기능을 설정하고 관리하는데 필요한 산업표준 소프트웨어 기술이다. 일반적으로 DCE는 상이한 운영체제와 이기종의 서버가 지리적으로 분산되어 있는 네트워 내에서 사용된다. DCE는 보안기능을 지원하며, IBM의 CICS, IMS 및 DB2 데이터베이스 접근 기능을 제공한다.

또한 컴퓨터 통신에서 DCE는 모뎀이나 다른 직렬장치가 컴퓨터와 데이터를 주고받기 위해 사용하는 RS-232C 인터페이스를 의미하기도 한다.

#### 2.2.3 ORB(Object Request Broker)

ORB는 객체간의 클라이언트/서버 관계를 연결시켜주는 미들웨어로, 클라이언트 관점에서 서버객체의 존재위치에 관계없이 원하는 자료를 투명하게 호출할 수 있게 해준다. 또, 처리할 객체를 찾고, 매개변수를 전달하고, 메소드를 호출하며, 처리결과를 되돌려주는 일을 담당한다. 이러한 과정을 거쳐 ORB는 이질적인 분산 환경에서 서로 다른 컴퓨터에 있는 응용프로그램의 상호 운용성과 다중 객체시스템에 대한 상호연결성을 제공한다.




(그림 1) ORB의 구조

## 3. XML

XML은 기존의 HTML과 SGML이 갖는 단점을 보완하여 작성된 차세대 웹 언어 표준이다. HTML은 손쉽게 웹 문서를 만들 수 있고 이식성이 좋은 반면에 HTML DTD의 구조에 맞게 문서를 작성하므로 보다 구조화되고 다양한 레이아웃과 태그 명을 정의한 문서를 작성하는데는 문제점을 가지고 있다. 또한 SGML은 문서의 내용에 의한 논리구조를 정의하고 태그를 생성하여 다양한 응용들 사이에 구조화된 문서 데이터를 상호 교환할 수 있고 문서 구조를 기반으로 검색저장 등의 응용에 사용되며 CALS(Continuous Acquisition and Life-Cycle Support)에서 표준 문서 포맷으로 사용하는 등 널리 사용되고 있다. 하지만 SGML은 복잡하여 관련 소프트웨어 개발이 쉽지 않으며, 웹 상에서의 서비스를 위한 목적으로 만들어지지 않았기 때문에 웹상에서 일반적인 기능을 수용하고 있지 않다. 그리하여 XML은 HTML이 갖는 장점인 웹 상에서의 정보 제공과 SGML이 갖는 자주 사용되지 않는 복잡한 기능을 제거한 서브셋의 기능으로 복잡한 문서 구조를 제공하고 이에 따른 문서 태그명도 자유롭게 제공하는 SGML과 HTML의 장점을 수용하고 단점을 상호 보완하여 제공한다[8].

XML의 특징은 다음과 같다.

- SGML 특징을 수용하되 불필요한 사항을 줄여 가볍게 만든 메타(Meta) 마크업 언어이다.
- 여러 가지 종류의 마크업 표현이 가능하도록 확장성을 부여하고 있다.
- 구문과 구조에 대한 규칙을 갖고 있다. 그리고 이 규칙을 읽고 검사하기 위한 각종 제약 사항을 제시한다.
- 구조적인 문서 혹은 구조적인 자료를 저장하고 처리할 수 있는 매커니즘을 제공한다.
- XML 프로세서로 불리는 XML 처리 모듈에 대한 행동 양식을 정의한다.

XML을 활용할 수 있는 응용분야는 다음과 같다.

- 해당 분야 중심 XML 응용 - XML기술에 설계를 맞추는 것이 아니라, 해당 분야에서 오는 요구 조건에 맞추어서 XML 기술을 적용하는 방식을 사용한다.
- XML 표준 중심 XML 응용 - 응용 프로그램이 XML 관련 표준을 준수할 수 있도록 요구 사항 단계에서부터 설계에 이르기까지 철저하게 표준 관점에서 중요한 결정을 내리는 방법을 사용한다. DTD로 표현된 문서의 집합이 동일한지 XML 파서를 사용하여 자료가 올바른지 검증함으로서 이기종 데이터베이스간의 데이터 교환을 쉽게 할 수 있다.
- 문서 중심 XML 응용 - 문서에 대한 논리적인 요소와

물리적인 요소를 분리시키며, 논리적인 요소만이 존재하는 XML 문서에 물리적인 요소를 적용하여 사용자에게 동일한 데이터를 다양한 형태로 손쉽게 보여줄 수 있다.

- 자료 중심 XML 응용 - 데이터베이스 스키마와 레코드를 XML 파일로 표현하고, XML 파일을 다시 데이터베이스로 복원하는 기능을 수행 할 수 있다. 어떤 특정 데이터 베이스에 의존하는 대신 일반적인 데이터베이스를 XML 파일로 표현하는 기법으로 데이터 베이스 간에 자유로운 문서 교환이 가능하게 할 수 있다

이와 같이 XML은 다양한 응용분야의 요구사항을 충족시켜줄 수 있으며, 많은 응용 개발에서 사용된다[5, 9].

## 4. RDB와 XML간의 상호 변환 기법

### 4.1 RDB와 XML 파일에 대한 상호 변환 규칙

관계형 데이터베이스의 테이블 단위로 DTD를 만들고 이 테이블에 포함된 모든 레코드를 XML 파일로 표현하는 방법을 사용한다.

관계형 데이터베이스관리시스템은 여러 테이블에 분산되어 저장되므로, 각 테이블 단위로 XML로 변환할 수만 있다면 나중에 데이터베이스를 복원할 경우 향상성을 보장할 수 있다. 단, 전송시에 테이블 변동이 없다는 가정이 필요하다. 즉, 관계형 데이터베이스를 XML 파일로 옮길 경우 독자적으로 존재하는 테이블 단위로 이동할 수 있다. 제안한 시스템은 DTD로 스키마를 표현하므로 "필드 이름, 필드 타입, 필드 크기"를 자유롭게 표현할 수 있다.

필드 이름은 엘리먼트 이름으로 두며, 이럴 경우 테이블 스키마에 따라 DTD가 매번 달라지므로 주의가 필요하다. 그러나 필드 타입과 필드 크기에 대한 표현 방법은 저절로 해결되므로, 필드 타입과 필드 크기를 엘리먼트 속성으로 고정시켜두고, DTD를 정의하면서 필드에 관련된 정보를 부여할 수 있다.

관계형 데이터베이스 관리 시스템에 Person이라는 테이블이 있고, 이 테이블에 속한 필드가 Number, Name, Birth 세 가지이며, Field 타입은 각각 LONG(길이 : 4), TEXT(길이 : 50), DATE(길이 : 11)이라면, 관계형 데이터베이스 Person의 스키마는 <표 1>과 같다.

〈표 1〉 관계형 데이터베이스 Person의 Schema

| Table Name | Person | |
|---|---|---|
| Field Name | Field Type | Field Length |
| Number | LONG | 10 |
| Name | TEXT | 50 |
| Birth | DATE | 11 |

<표 2>는 Person에 속한 레코드이다.

〈표 2〉 관계형 데이터베이스 Person의 Record Set

| Number | Name | Brith |
|---|---|---|
| 200101 | 홍길동 | 1974-04-29 |
| 200102 | 이길동 | 1978-02-12 |
| 200103 | 신길동 | 1976-03-21 |

<표 2>를 보면 Person에 속한 레코드는 {200101, "홍길동", "1974-04-29"}, {200102, "이길동", "1978-02-12"}, {200103, "신길동", "1976-03-21"}이다. 이런 스키마 정보와 레코드 집합을 사용하여 데이터베이스를 XML로 변환하고, 변환된 XML 파일을 이용하여 원래 데이터베이스로 변환 복원하는 규칙은 다음과 같다.

### 4.2 RDB를 XML 파일로 변환하는 규칙

관계형 데이터베이스 파일로 변환하는 규칙은 크게 두 단계이다. 첫 번째 단계는 데이터베이스 스키마 정보를 사용하여 DTD를 만드는 작업이며, 두 번째 단계는 데이터베이스 레코드 집합을 사용하여 DTD에 순응하는 엘리먼트 트리를 만들어내는 작업이다.

#### 4.2.1 DTD를 만드는 작업

DTD에 포함되어야 할 내용은 <표 1>에서 FIELD NAME, FIELD TYPE, FIELD LENGTH이다. FIELD NAME은 ELEMENT NAME으로 취급하고 FIELD TYPE과 FIELD LENGTH는 이 ELEMENT 속성으로 정의하여 모델링한 것은 <표 3> Person.dtd와 같다.

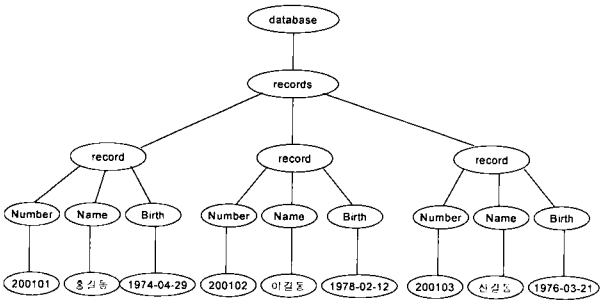
〈표 3〉 Person.dtd

```
<!ELEMENT database (records)>
<!ELEMENT records (record)*>
<!ELEMENT record (Number, Name, Birth)>
<!ELEMENT Number (#PCDATA)>
<!ATTLIST Number TYPE
(BIT|TINYINT|SMALLINT|INTEGER|BIGINT|FLOAT|LONG|
DOUBLE|NUMBERIC|
DECIMAL|CHAR|VARCHAR|LONGVARCHAR|TEXT|DATE|T
IME|TIMESTAMP|DATETIME)
#FIXED "LONG" MAXLEN CDATA #FIXED "10">
<!ELEMENT Name (#PCDATA)>
<!ATTLIST Name TYPE
(BIT|TINYINT|SMALLINT|INTEGER|BIGINT|FLOAT|LONG|
DOUBLE|NUMBERIC|
DECIMAL|CHAR|VARCHAR|LONGVARCHAR|TEXT|DATE|T
IME|TIMESTAMP|DATETIME)
#FIXED "TEXT" MAXLEN CDATA #FIXED "50">
<!ELEMENT Birth TYPE
(BIT|TINYINT|SMALLINT|INTEGER|BIGINT|FLOAT|LONG|
DOUBLE|NUMBERIC|DECIMAL|CHAR|VARCHAR|LONGVAR
CHAR|TEXT|DATE|TIME|TIMESTAMP|DATE|TIME)
#FIXED "DATE" MAXLEN CDATA #FIXED "11">
```

#### 4.3.2 ELEMENT TREE를 만드는 작업

Person.dtd를 만드는 작업이 완료되면, 이제 데이터베이스 레코드를 엘리먼트 트리로 표현하고 API를 사용하여 XML

파일을 작성한다. Person.dtd에 의하면 엘리먼트 트리의 뿌리는 database이다. 이 database 아래에 records가 자손으로 삽입되고 records 아래에 record로 묶어둔 각각의 레코드 Number, Name, Birth가 차례로 들어가는 방식으로 DOM (Document Object Model) 트리가 구성된다. (그림 2)은 <표 3>에서 정의한 Person의 record set에 순응하는 DOM 트리로 만든 엘리먼트 트리의 구성이다.




(그림 2) Person.dtd 의 element tree

4.3 XML 파일을 RDB로 변환하는 규칙

XML 파일을 관계형 데이터베이스로 변환하는 규칙 또한 크게 두 단계를 거친다. 첫 번째 단계는 데이터베이스 스키마 정보를 복원하는 작업이며, 두 번째 단계는 복원된 스키마 정보를 사용하여 만든 테이블에 데이터베이스 레코드 집합을 넣는 작업이다.

4.3.1 스키마 정보를 복원하는 작업

XML 문서를 파싱하는 과정에서 엘리먼트 이름과 속성을 빼내어 스키마 정보를 복원하는 방법을 사용한다.

4.3.2 레코드 집합을 넣는 작업

복원된 스키마를 사용하여 테이블을 먼저 생성시킨 다음 characters 메소드에 넘어오는 records 엘리먼트 하부에 엘리먼트의 값을 사용하여 SQL 문을 수행하는 과정을 반복하여 레코드를 넣을 수 있다.

## 5. 시스템 설계 및 구현

5.1 설계 목표

본 논문에서 구현하고자 하는 시스템의 목표는 다음과 같다.

5.1.1 관계형 데이터베이스와 XML 문서간의 상호 변환

관계형 데이터베이스에 존재하는 스키마와 레코드를 특정 XML 양식으로 변환한 다음, 관계형 데이터베이스에 스키마를 그대로 생성시키고, 레코드를 손실 없이 복원하려면 대부분 데이터베이스를 백업 또는 덤프 받은 다음 이를 다시 올려서 사용하는 방법을 택하고 있다. 이러한 방법은 반드시 동일한 데이터베이스 파일 시스템을 사용하는 전제 조건이 필요하다.

본 논문에서는 자료 전달관점에서 XML을 응용한 방법으로 DOM Builder를 이용하여 데이터베이스 스키마와 레코드를 XML 파일로 변경하고, 이를 SAX Parser를 이용하여 데이터베이스 스키마와 레코드로 변환시키는 시스템 구성을 제안한다.

5.1.2 XML 파일을 이용한 질의

검색을 원하는 엘리먼트 이름과 검색 조건을 입력하면 변경된 XML FILE을 DOM Builder를 이용하여 수정, 삽입, 삭제, 검색이 가능하도록 구성한다.

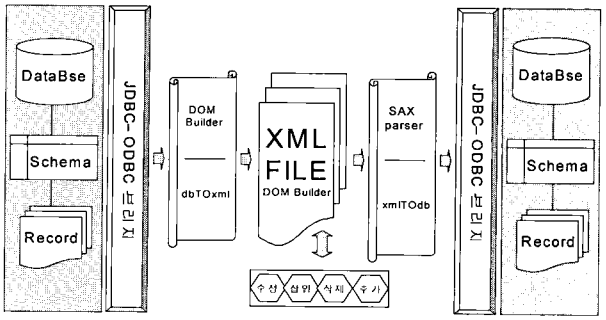
5.2 시스템 구조

(그림 3)은 본 시스템에서 제안한 XML 문서 처리 과정 미들웨어 구성도이다. 본 시스템은 크게 dbTOxml과 xmldb와 xmlTOdb로 크게 나누어진다.

dbTOxml은 자료전달관점에서 사용하는, XML 기술을 응용한 DOM Builder를 이용하여 데이터베이스 스키마와 레코드를 XML파일로 변경하는 기능을 하고 있다.

xmldb모듈은 변경된 XML FILE을 DOM Builder를 이용하여 XML 파일을 해석해서 DOM트리를 만들어 두고, 이 트리를 조작하는 방법으로 마치 XML 파일을 데이터베이스처럼 취급하는 수정, 삽입, 삭제, 검색이 가능하도록 구성하였다.

xmlTOdb은 SAX Parser를 이용하여 각종 연산이 수행된 XML문서를 데이터베이스 스키마와 레코드로 변환시켜 데이터베이스에 넣는 작업을 수행한다.




(그림 3) XML 문서 처리 과정

5.2.1 dbTOxml Module

(1) dbTOxml Class

JDBC설정과 XML 파일 생성에 필요한 객체를 생성한다. main 메소드는 dbTOxml 프로그램 자체에 대한 진입점을 제공하며, 명령행에서 JDBC 연결 URL과 테이블 이름을 제공받고, XML 파일 생성에 필요한 객체를 생성한다.

(2) schemaTOdtd

데이터베이스 스키마를 XML 파일을 위한 DTD로 변환하는 역할을 한다.

createDTD라는 메소드는 dbTOxml에서 획득한 데이터베이스 관련 메타 정보를 사용하여 DTD를 구성하는 기능을

한다.

(3) prettyDOM

prettyDOM은 DOM 메모리 출력을 담당하고, Document 객체를 받아서 하부 엘리먼트를 부모-자식-형제 관계에 맞추어 문자열로 만드는 역할을 한다. 인수로 넘어오는 엘리먼트의 타입이 org.w3c.dom.Node.ELEMENT_NODE일 경우 Recursive call을 수행하면서, 자손 엘리먼트에 대해 동일한 작업을 반복한다.

(4) recordTOdom

dumpXML 메소드는 데이터베이스 레코드 정보를 사용하여 DOM 트리를 구성하고 XML파일로 변환하는 기능을 제공한다.

5.2.2 xmldb 연산 Module

(1) xmlTOdb class

JDBC에 관련된 설정과 데이터베이스 복원에 필요한 객체를 생성한는 class이다. main 메소드의 역할은 진입점을 제공과 JDBC을 연결하고, URL 테이블 이름·XML 파일명을 받아들인다. 또한, 데이터베이스 복원에 필요한 객체를 생성한다.

(2) xmlTOrecords class

SAX 구동방식을 따르는 클래스인 xmlTOdbSAX 인스턴스를 생성하고 이 객체에게 파싱을 시작하도록 제어권을 넘긴다.

(3) xmlTOdbSAX class

xmlTOdbSAX class는 xmlTOdb 핵심논리처리루틴으로 SAX API를 사용하여 XML 파일을 파싱하면서 데이터베이스를 복원한다. xmlTOdbSAX class에는 xmlTOdbSAX, doParsing, startElement, endElement, characters 메소드가 있다.

- xmlTOdbSAX method : xmlTOSAX 클래스 생성자이며, 초기화를 수행한다.
- doParsing method : xmlTOdbSAX 클래스 진입점이며, SAX 파서를 초기화하여 XML 파일에 대한 파싱을 시작한다.
- startElement method : element 이름과 속성으로 테이블을 생성하고 레코드 삽입을 위해 필요한 스키마 정보를 뽑아낸다.
- eldElement method : 삽입 직전 스키마를 생성하고 레코드 단위로 삽입한다
- characters method : 레코드에 삽입될 실제 내용을 뽑아내고, 필요없는 개행문자를 제거한다.

5.2.3 xmlTOdb Module

(1) xmldb class

- main : xml 프로그램 자체 진입점을 제공하고 메뉴구동 방식의 무한 루프를 돌면서 사용자 입력을 처리한다.
- findRecords : 사용자가 입력한 조건을 사용하여 레코드를 찾아 삭제, 변경하고, 아무런 조건이 없다면 현재 존재하는 모든 레코드들을 출력한다.
- extractElement : record element 하부에 속한 모든 extractElement : record element 하부에 속한 모든 element 이름을 Vector 구조체에 넣는다.
- showRecord : record element 하부에 속한 모든 show Record : record element 하부에 속한 element들에 대해 엘리먼트 이름과 값을 뽑아내어 출력하고 앞뒤에 존재하는 개행문자를 제거한다.

record element 하부에 속한 모든 그외에도 레코드를 삽입, 삭제, 치환하는 insertRecord, deleteRecord, updateRecord가 있다.

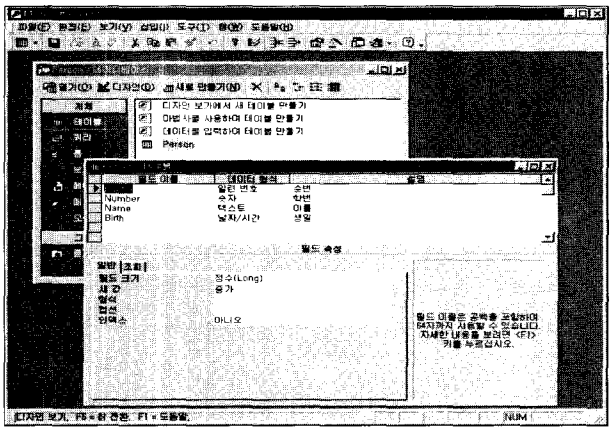
(2) lineinput class

한행에 걸쳐 사용자의 입력을 받아들여 getInput method 하나 만을 선언한다. getInput method는 인수로 문자열을 받아들여 이를 표준 출력으로 출력하고, 표준 입력에서 받아들인 문자열을 반환한다.

(그림 4)은 <표 4>을 사용하여 작성한 Person 테이블을

<표 4> Access에 사용할 Person 스키마

| 필드 이름 | 데이터형식 | 설 명 |
|---|---|---|
| Bunho | 정 수 | 순 번 |
| Number | 정수(Long) | 학 번 |
| Name | 문자열(50자 | 이 름 |
| Birth | 날짜/시간 | 생 일 |



(그림 4) Access를 사용하여 작성한 Person 테이블 스키마

<표 5> Person 테이블에 입력할 자료

| Bunho | Number | Name | Birth |
|---|---|---|---|
| 1 | 200103 | 홍길동 | 1974-04-29 |
| 2 | 200102 | 이길동 | 1978-02 12 |
| 3 | 200101 | 신길동 | 1976-03-21 |


(그림 5) Person 테이블에 입력한 자료

작성한 것이다. (그림 5)은 <표 5>을 사용하여 Person테이블에 입력한 자료이다. 데이터베이스가 다 완성되었으면 데이터베이스를 XML 파일로 변환시키기 위해 다음과 같이 도스창에서 실행시킨다.

**C:\dbxml\dbTOxml>java db2xml jdbc : odbc : Ps Person>person.xml**

위 명령문에서 첫 번째 인수(jdbc : odbc : )는 접속할 URL을 지정하고 두 번째 인수(Ps Person)는 테이블 이름을 부여하여 실행시킨다. 실행시킨 결과 아래 <표 6> preson.xml파일이 생성된다.

〈표 6〉 person.xml

```
< database >
< records >
< record >
< Number >
  200101
< /Number >
< Name >
  홍길동
< /Name >
< Birth >
  1974-04-29
< /Birth >
< /record >
    .
    .
< record >
< Number >
  200103
< /Number >
< Name >
  신길동
< /Name >
< Birth >
  1976-03-21
< /Birth >
< /record >
< /records >
< /database >
```

<표 7>는 <표 6> xml 파일을 데이터베이스로 변환한 것이다. xmlTOdbd을 도스상에서 실행하면 다음과 같다.

**C:\dbxml\xmlTOdb>java xmlTOdb jdbc : odbc : Ps Person> person.xml**

첫 번째( jdbc : odbc : ), 두 번째 인수(Ps Person)는 db Toxml 과 동일하며 dbTOxml에서 만들어진 XML파일 이름을 세 번째 인수(person.xml)로 추가하였다.

〈표 7〉 person.xml을 database로 변환

```
tag : database
tag : records

tag : record
tag : Bunho
contents : 1
tag : Number
contents : 200101
tag : Name
contents : 홍길동
tag : Birth
contents : 1974-04-29
creating schema : create table Person (Bunho, Number LONG,
Name TEXT(50),
Birth DATETIME)
inserting a record : insert into Person (Bunho, Number, Name,
Birth)
values(1, 200101, '홍길동', '1974-04-29')
tag : record

tag : Bunho
contents : 2
tag : Number
contents : 200102
tag : Name
cnotents : 신길동
tag : Birth
contents : 1978-02-12
creating schema : create table Person (Bunho, Number LONG,
Name TEXT(50),
Birth DATETIME)
inserting a record : insert into Person (Bunho, Number, Name,
Birth)
values(2, 200102, '신길동', '1978-02-12')
tag : record

tag : Bunho
contents : 3
tag : Number
contents : 200103
tag : Name
contents : 이길동
tag : Birth
contents : 1976-03-21
creating schema : create table Person (Bunho, Number LONG,
Name TEXT(50),
Birth DATETIME)
inserting a record : insert into Person (Bunho, Number, Name,
Birth)
values(3, 200103, '이길동', '1976-03-21')
```

## 6. 결론 및 향후 연구 과제

본 논문는 관계형 데이터베이스에 존재하는 표준 EDI문서들이 XML 기반의 문서로 변환되기 위해서는 관계형 데이터베이스에 존재하는 스키마와 레코드들을 특정 XML 양식으로 변환한 다음, 다시 관계형 데이터베이스에 스키마를 그대로 생성시켜야한다. 변환된 레코드를 손실 없이 복구하려면 대부분 데이터베이스를 백업 또는 덤프 받은 다음 이를 다시 사용해 왔던 기존의 방식을 JDBC 브리지를 사용하여 관계형 데이터베이스 관리 시스템에서 스키마와 스키마에 의해 만들어지는 테이블에 포함된 레코드를 XML 파일로 변환하는 dbTOxml 변환기를 사용하였다. dbTOxml 변환기에 의해 만들어지진 XML 파일을 사용하여 스키마를 생성하고 레코드를 삽입하는 xmlTOdb 변환기를 사용하였고 XML 파일을 데이터베이스처럼 사용하도록 한 dbxml를 구성하였다.

이는 반드시 동일한 데이터베이스 관리 시스템을 사용해야 한다는 전제 조건을 필요했던 기존 방식을 탈피했으며 외부로 전달할 경우 환경에 따라 제대로 동작하지 않을 가능성이 높았던 제한점을 극복했다.

향후 연구 과제는 기업의 EDI 문서는 보안을 유지하기 위한 시스템이 필요하므로 문서를 암호화하여 보낼 수 있는 연구가 필요하며, 둘째는 현재 DTD를 갖고서 얼마든지 XML 응용을 작성할 수 있지만 Schema 표준을 사용할 경우 훨씬 효과적으로 데이터베이스를 XML 파일로 표현 할 수 있기 때문에 Schema 표준에 대한 연구가 필요하다. 마지막으로 XML 문서가 가지고 있는 구조 정보는 일반적으로 트리 형태의 계층적 구조로 표현 될수 있기 때문에 객체지향 데이터베이스로 관리하는 것이 효율적이므로 이에 대한 연구가 요구되어진다.

## 참 고 문 헌


[1] 한국전자거래표준원, "한국EDI표준코드집", 한국전자거래표준원, 1998.

[2] T. Bray et al., "Extensible Markup Language (XML) 1.0," http://www.w3.org/TR/1998/REC-xml-19980210, 1998.

[3] 신동규 외 2인, "XML/EDI 시스템의 설계 및 구현", 정보처리학회논문지 D, 제8-D권 제2호, 2001.

[4] XML/EDIGroup, "XML/EDITransactionModels," http://www.geocities.com/WallStreet/Floor/5815/.

[5] Frank Boumphrey, Olivia Direnzo, etc., "XML Application," Wrox Press

[6] S. Abiteboul et al., "Active Views for Electronic Commrece," Proc. Int'l Conf. on VLDB, pp138-149, 1999.

[7] Y. Papakonstantinou and V. Vianu, "DTD Inference for Views of XML Data," Proc. of 19th ACM SIGACTSIGMOD _SIGART Symp. on PODS, pp.35-46, 2000.

[8] 임영태 외 2인 "XML에 기반한 EDI 문서교환 시스템 설계 및 구현", 정보처리학회논문지, 제7권 제11호, 2000.

[9] 신상철, "이질형 데이터베이스간의 문서교환을 위한 XML/EDI 설계 및 구현", 조선대학교 대학원 석사학위논문, 2001.



**최 광 미**

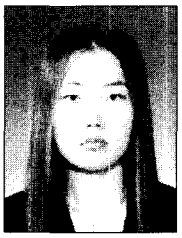
e-mail : iplab@hanmail.net

1990년 조선대학교 전자계산학과(이학사)

1995년 조선대학교 대학원 전산 통계학과 (이학석사)

2003년 조선대학교 대학원 전산 통계학과 박사과정 수료

관심분야 : 멀티미디어, 신경망, 멀티미디어 컨텐츠, 인공지능, 정보보호


**박 수 영**

e-mail : swiminpark@hanmail.net

2001년 조선대학교 전산통계학과(이학사)

2001년~현재 조선대학교 대학원 전산통계학과 석사과정

관심분야 : 멀티미디어, 신경망, 멀티미디어 컨텐츠, 인공지능, 정보보호



**정 채 영**

e-mail : cyjung@chosun.ac.kr

1983년 조선대학교 컴퓨터공학과(공학사)

1986년 조선대학교 대학원 전산 전공 (공학석사)

1989년 조선대학교 대학원 전산 전공 (공학박사)

1986년~현재 조선대학교 자연과학대학 수학·전산 통계학부 교수

관심분야 : 멀티미디어, 신경망, 멀티미디어 컨텐츠, 인공지능, 정보보호