XML기반의 B2B 문서교환 Broker System 설계 및 구현

최광미* 박수영*, 정채영*
조선대학교 자연과학대학 전산통계학과

Design and Implementation of Broker System supporting translation about B2B document using XML

Gwang-Mi Choi, Su-Young Park, Chai-Yeoung Jung
Dept of Computer Science and Statistic, Chosun University
E-mail : {iplab, swiminpark}@hanmail.net, cyjung@chosun.ac.kr

요 약

EDI를 사용한 전자문서 처리는 두 기업간의 데이터를 전달하기 위해 VAN을 이용하여 데이터를 전달했으나 VAN의 폐쇄성으로 인해 확장에 문제가 제시되었다. 이러한 문제점을 해결하기 위하여 다양한 문서구조 표현이 가능한 XML (eXtensible Markup Language)을 활용한 EDI 서비스 환경으로 옮겨가고 있다. 본 논문에서는 JDBC 브리지를 사용하여 두 관계형 데이터베이스에 존재하는 EDI 메시지를 브러커시스템를 이용한 XML 변환기법을 제안한다. 변환된 XML을 사용하여 스키마를 그대로 복구하는 동시에 정의된 테이블에 원래 레코드들을 그대로 삽입하는 방법을 제시하면서, 반드시 동일한 데이터베이스 관리시스템을 사용해야 한다는 전제조건을 필요했던 기존방식을 탈피했으며 외부로 전달할 경우 환경에 따라 제대로 동작하지 않을 가능성이 높았던 제한점을 극복했다.

1. 서 론

전세계적으로 인터넷의 확산과 더불어 초고속정보통신망의 구축 작업이 활발히 진행되고 있으며, 이에 따라 이러한 정보통신망을 이용하여 정보를 저렴하고 편리하게 교환하며 업무를 효율적으로 처리하기 위한 방안을 모색하고 표준화하려는 노력을 하고 있다.[1] 이러한 노력이 결실을 맺기 위해서는 우선 전자문서의 표준화 작업이 필요하며 각 표준기구들이 제시한 전자문서에 대한 표준들이 꾸준히 제시되고 있다. 이러한 노력의 결과로 가장 일반적으로 쓰이고 있는 대표적인 형태가 바로 EDI(Electronic Data Interchange)이다. 현재의 EDI서비스는 송신자, 수신자, VAN(Value-Added Network)간의 교환할 문서를 미리 정의하여 등록시켜 둔 EDI 전용 소프트웨어를 이용하여 전자문서를 주고 받는다. 이러한 환경에서 EDI 서비스는 새로운 형태의 문서를 교환해야 할 필요가 생겼을 경우 사용자마다 새로운 문서에 정보를 등록하여 EDI 전용 소프트웨어의 변경이 필요하다. 그러므로 끊임없이 변화하는 다양한 사용자의 욕구를 만족시키기에는 동적 기능이 부족한 것이 EDI 서비스 환경이다.

본 논문의 연구목적은 XML기반 EDI문서교환 브러커시스템 를 설계하고 구현하는 것이다. 기존 EDI의 문제점인 고정적이고 정형화된 데이터를 JDBC 변환기를 사용하여 관계형 데이터베이스 관리 시스템에서 스키마와 스키마에 의해 만들어지는 테이블에 포함된 레코드들을 XML파일로 변환한다. 네트웍을 통해 변환된 XML 파일을 사용하여 원래 스키마로 그대로 복구하는 동시에 이 스키마에 의해 정의된 테이블에 원래 레코드들을 그대로 삽입하는 기능에 대하여 구현하였다.

2. XML

XML은 기존의 HTML과 SGML이 갖는 단점을 보완하여 작성된 차세대 웹 언어 표준이다. HTML은 웹 상에서 손쉽게 하이퍼미디어 문서를 만들 수 있고 이식성이 뛰어나 많은 사람들이 사용하여 왔다. 반면에 현재 HTML DTD가 제공하는 구조만의 문서를 만들므로 보다 구조화되고 다양한 레이아웃과 태그 명을 정의하여 문서를 만드는데는 문제점을 가지고 있다. 그러나 HTML이 갖는 장점인 웹 상에서의 정보 제공과 SGML이 갖는 자주 사용되지 않는 복잡한 기능을 제거한 서브셋의 기능으로 복잡한 문서 구조를 제고하고 이에 따른 문서 태그명도 자유롭게 제공하는 SGML과 HTML의 장점을 수용하고 단점을 상호 보완하여 제공한다. 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 행동양식을 정의한다.

XML을 활용할 수 있는 응용분야는 다음과 같다.

- 해당 분야 중심 XML
- XML 표준 중심 XML응용
- 문서 중심 XML응용

· 자료 중심 XML 응용

이와 같이 XML은 다양한 응용분야의 요구사항을 충족시켜줄 수 있으며, 많은 응용 개발에서 사용된다.

3. RDB와 XML간의 상호 변환 기법

3.1 RDB와 XML 파일에 대한 상호 변환 규칙

관계형 데이터베이스의 테이블 단위로 DTD를 만들고 이 테이블에 포함된 모든 레코드를 XML파일로 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 XML파일로 옮길 경우 독자적으로 존재하는 테이블 단위로 이동할 수 있다. 제안한 시스템은 DTD로 스키마를 표현하므로 "필드 이름, 필드 타입, 필드 크기를 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 고정시켜(#FIXED)두고, 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.

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

<표 1> 관계형 데이터베이스 Person의 Schema

<표 2>을 따르는 Person에 속한 레코드가 다음과 같다.

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

<표 2> 관계형 데이터베이스 Persons ecord Set

[표 2]를 보면 Person에 속한 레코드는200101,"홍길동"19744-04-29"), (200102, "이길동", "1978-02-12"), (200103, "신길동", "1976-03-21")이다. 이런 스키마 정보와 레코드 집합을 사용하여 데이터베이스를 XML로 변환하며, XML 파일을 사용하여 원래 데이터베이스로 변환 복원하는 규칙은 다음과 같다.

3.2 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.

3.2.1 DTD를 만드는 작업

DTD에 포함되어야 할 내용은 [표 1]에서 FIELD NAME, FIELD TYPE, FIELD LENGTH 이다. FIELD NAME은 ELEMENT NAME으로 취급하고 FIELD TYPE과 FIELD LENGTH는 이 ELEMENT 속성으로 정의하여 모델링 한게 아래 <표 3> Person.dtd이다.

```
<!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|TIME|TIMESTAMP|DAT
ETIME)
#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|TIME|TIMESTAMP|DAT
ETIME)
#FIXED "TEXT" MAXLEN CDATA #FIXED "50">
<!ELEMENT Birth TYPE
(BIT|TINYINT|SMALLINT|INTEGER|BIGINT|FLOAT|LONG|DOUBLE|NUMBERIC|DE
CIMAL|CHAR|VARCHAR|LONGVARCHAR|TEXT|DATE|TIME|TIMESTAMP|DATE |
TIME)
#FIXED "DATE" MAXLEN CDATA #FIXED "11">
```

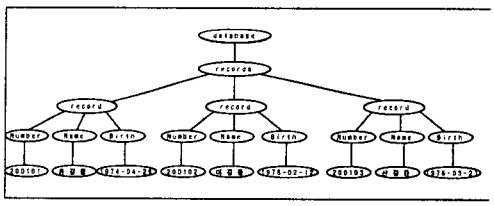
<표 3> Person.dtd

3.3.2 ELEMENT TREE를 만드는 작업

person.dtd를 만드는 작업이 완료되면, 이제 데이터베이스 레코드를 엘리먼트 트리로 표현하고 API를 사용하여 XML 파일을 작성한다.

person.dtd에 의해서 엘리먼트 트리의 뿌리는 database이다. 이 엘리먼트 database 아래에 records가 자손으로 삽입되며, records 아래에 record로 묶어둔 각각의 레코드 Number, Name, Birth가 차례로 들어가는 방식으로 DOM트리가 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

3.3 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 이렇게 복원된 스키마 정보를 사용하여 만든 테이블에 데이터베이스 레코드 집합을 넣는 작업이다.

3.3.1 스키마 정보를 복원하는 작업

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

3.3.2 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.

4. 시스템 설계 및 구현

4.1 설계 목표

본 논문에서 구현하고자하는 시스템의 목표는 다음과 같다.

(1) 관계형 데이터베이스와 XML 문서간의 상호 변환

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

는 전체 조건이 필요하다. 자료를 전달하는 관점에서, 사용하는 XML 기술을 응용하여, DOM Builder를 이용하여 데이터베이스 스키마와 레코드를 XML 파일로 변경하고, 이를 다시 SAX Parser를 이용하여 데이터베이스 스키마와 레코드로 변환시키는 시스템 구성을 제안한다.

(2) XML 파일을 이용한 질의

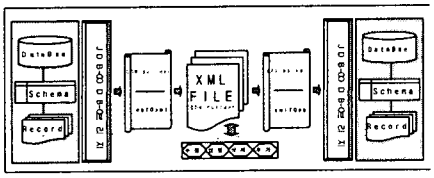
검색을 원하는 엘리먼트 이름과 검색 조건을 입력하면 변경된 XML FILE을 DOM Builder를 이용하여 자유로이 수정, 삽입, 삭제, 검색이 가능하도록 구성한다.

4.2 시스템 구조

[그림 5.1]은 본 시스템에서 제안한 XML 문서 처리 과정 브러커 이다. 본 시스템은 크게 dbTOxml과 xmldb와 xmlTOdb로 크게 나누어진다.

dbTOxml은 자료를 전달하는 관점에서 사용하는 ,XML 기술을 응용하여, 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 각종 연산 즉, 수정, 삽입, 삭제, 검색이 가능하도록 구성되었다. xmlTOdb은 SAX Parser를 이용하여 각종 연산이 수행된 XML문서를 데이터베이스 스키마와 레코드로 변환시켜 데이터베이스에 넣는 작업을 수행한다.



(그림 3) XML 문서 처리 과정

4.2.1 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 받아들인다. XML 파일 생성에 필요한 객체를 생성한다.

(2) schemaTOdtd

데이터베이스 스키마를 XML 파일을 위한 DTD로 변환하는 역할을 한다. createDTD라는 메소드가 하나 있는데, 이 메소드는 dbTOxml에서 획득한 데이터베이스 관련 메타 정보를 사용하여 DTD를 구성하는 기능을 한다.

(3) prettyDOM

DOM Level 1에는 표준적인 메모리 출력 방식이 정의되어 있지 않다. prettyDOM은 DOM메모리 출력을 담당하고 있는데, 메모리 출력 방식의 호환성 문제를 극복할 수 있다.

Document 객체를 받아서 이 문서 엘리먼트 하부 엘리먼트를 부모-자식-형제간 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

dumpXML이라는 메소드가 하나 있는데, 데이터베이스 레코드 정보를 사용하여 DOM 트리를 구성하며 XML파일로 변환하는 기능을 제공한다.

4.2.2 xmldb 연산 Module

(1)xmlTOdb class

JDBC에 관련된 설정과 데이터베이스 복원에 필요한 객체를 생성한다. main메소드는 진입점을 제공하고, JDBC 연결 URL테이블 이름, XML 파일 이름을 받아들인다. 또한, 데이터베이스 복원에 필요한 객체를 생성한다.

(2) xmlTOrecords class

SAX구동방식을 따르는 클래스 xmlTOdbSAX인스턴스를 생성하고 이 객체에게 파싱을 시작하도록 제어권을 넘긴다.

(3) xmlTOdbSAX class

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

· xmlTOdbSAX method : xmlTOSAX클래스 생성자이며. 초기화를 수행한다.

· doParsing method : xmlTOdbSAX클래스 진입점이며, SAX 파서를 초기화하여 XML파일에 대한 파싱을 시작한다.

· startElement method : element이름과 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 제거 한다.

4.2.3 xmlTOdb Module

(1) xmldb class

· main : xml프로그램 자체 진입전을 제공하며, 메뉴 구동 방식의 무한 루프를 돌면서 사용자 입력을 처리한다.

· findRecords : 사용자가 입력한 조건을 사용하여 레코드를 찾아 삭제, 변경하고, 아무런 조건이 없다면 현재 존재하는 모든 레코드들을 출력하는 기능을 한다.

· extractElement : record element하부에 속한 모든 element 이름을 Vector 구조체이 넣어둔다.

· showRecord : record element 하부에 속한 element들에 대해 엘리먼트 이름과 값을 뽑아내어 출력하며, 앞뒤에 존재하는 개행 문자를 제거한다.

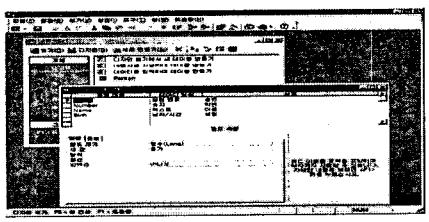
· 그외에도 레코드를 삽입, 삭제, 치환하는 insertRecord, deleteRecord, updateRecord가 있다.

(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.

필드 이름	데이터형식	설명
Bunho	정수	순번
Number	정수(Long)	학번
Name	문자열(50자	이름
Birth	날짜/시간	생일

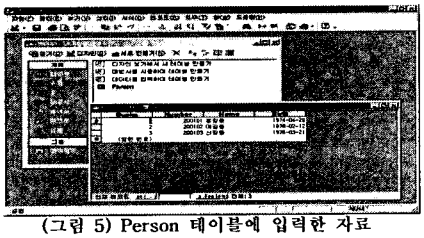
<표 4> Access에 사용할 Person 스키마



(그림 4) Access를 사용하여 작성한 Person 테이블 스키마

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

<표 5> Person 테이블에 입력할 자료



(그림 5) Person 테이블에 입력한 자료

(그림 4)은 <표 4>을 사용하여 작성한 Person테이블을 작성한 것이다. (그림 5)은 <표 5>을 사용하여 Person테이블에 입력한 자료이다. 데이터베이스가 다 완성되었으면 데이터베이스를 XML파일로 변환시키기 위해 첫 번째 인수 접속할 URL을 지정하고 두 번째 인수 테이블 이름 Preson을 부여하여 실행시킨다. 실행시킨 결과 아래 <표 6> Preson.xml과일이 생성된다.

```
<database>
<records>
<record>
<Number>
 200101
</Number>
<Name>
 홍길동
</Name>
<Birth>
  1974-04-29
 </Birth>
 </record>
  .
  .
 </database>
```

<표 6> Person.xml

<표 7>는 <표 6> xml 파일을 데이터베이스로 변환한 것이다. 첫 번째, 두 번째 인수는 dbToxml 과 동일하며 dbTOxml에서 만들어진 XML파일 이름을 세 번째 인수로 추가하였다.

```
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: 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')
```

<표 7> Person.xml을 database로 변환

5. 결론 및 향후 연구 과제

본 연구에서는 관계형 데이터베이스에 존재하는 표준 EDI 문서들이 XML기반의 문서로 변환되기 위해서는 관계형 데이터베이스에 존재하는 스키마와 레코드들을 특정 XML 양식으로 변환한 다음, 다시 관계형 데이터베이스에 스키마를 그대로 생성시키야하고, 그 레코드를 손실 없이 복구하려면 대부분 데이터베이스를 백업 또는 덤프 받은 다음 이를 다시 올려서 사용해 왔던 기존의 방식을 JDBC 브리지를 사용하여 관계형 데이터베이스 관리 시스템에서 스키마와 스키마에 의해 만들어지는 테이블에 포함된 레코드를 XML 파일로 변환하는 dbTOxml변환기를 사용하고, dbTOxml 변환기에 의해 만들어지진 XML 파일을 사용하여 스키마를 생성하고 레코드를 삽입하는 xmlTOdb 변환기를 사용하여, 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 될 수 있기 때문에 객체지향 데이터베이스로 관리하는 것이 효율적이므로 이에 대한 연구가 요구되어진다.

[참고문헌]

[1] Brian LOwe, Justin Zobel, ron Sacks-davis "A Formal Model for Databases of Structured Text,
"Proceedings of the Fourth International Conference on Database Systems for Advanced Application (DASFAA '95), pp.449-456,1995.

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

[3] W3C, "Document Object Model", http://www.w3.org/DOM/

[4] James Tauber, Linda van den Brink, "XML SOFTWARE", http://www.xmlsoftware.com/

[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.