바다-IV/XML 검색기 설계 및 구현

이명철*, 김상균*, 이미영**, 김명준**, 이규철*
*충남대학교 컴퓨터공학과
**한국전자통신연구원 컴퓨터•소프트웨어 기술연구소
e-mail : mclee@ce.cnu.ac.kr

Design and Implementation of BADA-IV/XML Query Processor

Myungcheol Lee*, Sang-Kyun Kim*, Kyu-Chul Lee*
Mi-Young Lee**, Myung-Joon Kim**
*Dept. of Computer Engineering, Chungnam National University
**Computer & Software Technology Lab., ETRI

요 약

XML 은 차세대 인터넷 전자 문서 표준으로서 주목을 받고 있다. 최근에 기존의 문서를 XML 로 변환하거나 신규 문서를 XML 로 작성하게 되면서 대량의 XML 문서가 생성되고 있고, 따라서 대량의 XML 문서를 효율적으로 검색하기 위한 XML 문서 검색기가 필수적 시스템으로 부각되고 있다. 본 논문에서는 XML 문서에 대한 내용기반, 구조기반, 속성기반 검색을 지원하는 XQL 과 객체지향 데이터베이스 표준언어인 OQL 을 결합한 형태의 XML 질의 언어를 설계하고, 한국전자통신연구원에서 개발한 바다-IV 내에 저장된 XML 문서에 대해 검색을 수행하는 바다-IV/XML 검색기의 설계 및 구현에 대하여 기술하였다.

1. 서론

XML[1]이 1998 년에 W3C (World Wide Web Consortium)에서 차세대 인터넷 표준으로 채택된 이후, XML 관련 응용이나 도구, 구문 분석기와 XML 을 지원하기 위한 다른 표준들이 속속 나오고 있고, 점차 기존의 웹 문서 포맷인 HTML 을 대체하고 있을 뿐만 아니라, 워드프로세서 문서, 전자상거래 문서, 전자교범 문서 등도 XML 문서로 만들어 지고 있다.

이러한 추세라면 향후 XML 로 만들어지는 문서의 양은 기하급수적으로 많아질 것이다. 따라서, 이런 대량의 문서를 저장하고 관리하는 기능은 필수적인 요소가 될 것이고, 대량의 저장된 문서에 대해서 원하는 문서를 찾는 검색 기능 또한 매우 중요하다.

XML 문서는 기존의 문서와는 달리 문서의 의미적인 구조를 엘리먼트와 애트리뷰트로 나타내고 엘리먼트 안에 원하는 문서의 내용을 표현한다. 예를 들어, 책의 서론, 본론, 결론, 제목, 장, 절, 구 등을 엘리먼트로 나타내어 책의 구성이나 구조가 어떻게 되어 있는 지를 한 눈에 알 수 있으며, 서론에 대한 내용은 서론 엘리먼트에 기술되어 있으므로 서론 엘리먼트에서 찾을 수 있다. 이런 특성을 가진 XML 문서에 대한 검색을 위해서는 기존의 검색기에서 제공하던 문서의 내용에 대한 검색 뿐만 아니라 문서의 논리적인 구조 정보와 속성 정보에 대한 검색도 할 수 있어야 한다.

본 논문에서는 객체 지향 데이터베이스 관리 시스템(OODBMS)인 바다-IV[2]에 저장이 되어 있는 대량의 XML 문서를 효율적으로 검색하기 위한 연구로서 XML 문서에 대한 검색어의 설계와 검색기 구현에 대한 내용을 다룬다.

본 논문에서 구현한 XML 검색기는 XML 문서의 구조적인 특성이 잘 유지될 수 있는 OODB 내에, 또한 W3C 의 표준인 DOM(Document Object Model)[3]을 기반으로 하는 저장 구조를 가지고 있는 시스템 내에 저장된 XML 문서에 대한 검색을 수행한다.

XML 검색기에는 XML 문서가 지닌 내용 정보, 구조 정보, 속성 정보에 대해 처리가 가능하고, 사용자에게 쉽고 사용이 편한 질의 언어가 필요한데, 본 논문에서는 OODB 의 OQL(Object Query Language)[4]과 XML 질의 언어인 XQL (XML Query Language)[5]을 결합하여 OODB

† 본 논문은 한국전자통신연구원의 " 멀티미디어 문서 모델링 도구 개발 (위탁계약번호:99-44)" 과제의 일부로 수행된 결과임.

내에 저장된 XML 문서를 효율적으로 검색할 수 있는 질의어를 설계한다.

본 논문의 구성은 다음과 같다. 먼저 2 장에서 XML 관련 질의 언어를 살펴 본다. 3 장에서는 바다-IV/XML 질의 언어의 설계 내용을 기술한다. 4 장에서는 바다-IV/XML 검색기의 설계 내용을 기술하고, 5 장에서는 구현 내용을 기술한다. 마지막으로 6 장에서는 본 논문의 결론을 기술한다.

2. 관련 연구

2.1 XQL

XSL(Extensible Stylesheet Language)[7]은 XML 문서에 대한 스타일시트 언어인데, XSL 패턴 언어를 통해 특정 노드를 찾아서, 스타일시트를 적용한다. XQL[4]은 XSL 의 패턴 언어를 확장한 형태로 XSL 의 노드에 대한 처리에 불리언 논리, 필터, 노드의 집합에 대한 색인을 추가했고, 질의와 패턴을 사용하는 데에 단순한 구문을 사용하여 간결하고 간단하다는 것이 특징이다.

2.2 XML-QL

XML-QL[8]은 현재 W3C 에 Note 로 제출된 상태이며, 대용량의 문서에서 데이터를 추출하고, 여러 문서 사이에 데이터를 주고 받고, 한 문서에서 다른 문서로 내용을 변환하고, 문서간의 통합을 어떻게 할 것인가에 초점을 맞춘 언어이다.

XML-QL 은 크게 WHERE 절과 CONSTRUCT 절로 구분 되며, WHERE 절에는 변수와 태그의 생략(</>)을 사용하여 검색 조건을 나타내고 대상이 되는 XML 문서를 "IN" 다음에 지정하여 여러 문서에 대한 검색을 수행할 수 있다. CONSTRUCT 절에서는 WHERE 절의 조건을 만족하는 결과를 추출하여 새롭게 결과를 재구성하는 부분을 기술한다.

3. 바다-IV/XML 질의 언어의 설계

XQL 은 XML-QL 에 비해 데이터베이스 기능을 제외하고는 많은 장점이 있다. XQL 에 데이터베이스 기능을 보완하기 위한 방법으로 객체 지향 데이터베이스의 OQL 을 이용하여 모델링을 한다면 기존의 객체 지향 데이터베이스에서 XML 문서를 검색하기 위한 질의 언어의 설계가 가능하다.

기존의 OODBMS 의 OQL 에는 XML 문서에 대한 검색을 지원하는 부분이 없으므로, 본 바다-IV/XML 검색기에서는 XML 문서에 대한 검색을 위해 OQL 을 확장하여 XQL 을 OQL 과 긴밀하게 결합시켜 사용할 수 있게 한다. 즉, <그림 1>과 같이 OQL 구문의 WHERE 절에 <xql_predicate>를 추가하여 XQL 질의를 입력하는 것이다. 이렇게 OQL 과 XQL 이 긴밀하게 결합된 형태를 본 논문에서는 OXQL 이라고 부른다.

```
OXQL :=
'select' <select_list>
'from' <class_name> [inherited_class_flag>]
['where' <search_condition>]
['order by' <sort_specification_list>]
search_condition :=
<xql_predicate> | <comparison_predicate> |
<in_predicate> | <like_predicate>
xql_predicate :=
<value expression>   'contains'   '('   <Bada-
IV/XQLQuery>  ')'
```

그림 1. OQL 과 XQL 을 결합한 OXQL 의 구문 정의

OXQL 에서 SELECT 절과 FROM 절은 OQL 의 문법과 같고 WHERE 절에 <xql_predicate>이 추가되었다. <xql_predicate>의 검색 대상은 객체나 객체의 애트리뷰트가 되고, 이 검색 대상에 'contains'를 적용하여 XQL 질의를 하게 된다.

<그림 1>에서 <Bada-IV/XQLQuery>는 XQL 에 엘리먼트의 내용이나 애트리뷰트의 내용이 문자열인 경우, 와일드 문자를 포함한 패턴에 대한 비교 연산을 위한 LIKE 연산과 지정된 엘리먼트의 전문 내용에 대한 유사도 검색을 위한 RELATED 연산을 지원하도록 본 논문에서 확장한 것이다.

OXQL 을 사용한 질의의 예로는 다음과 같은 것이 있을 수 있다.

```
논문 DTD 의 모든 문서에서 제목에 'XQL'이란 단어가
포함된 모든 문서를 찾아라.
SELECT *
FROM 논문_Document
WHERE this CONTAINS ( "제목 RELATED
'XQL' " )
```

4. 바다-IV/XML 검색기의 설계

바다-IV/XML 검색기는 크게 XML 검색 관리기, XML 질의 처리기, XML 결과 처리기, XQL 구문 분석기의 네 부분으로 구성되며, 구조는 <그림 2>와 같다.

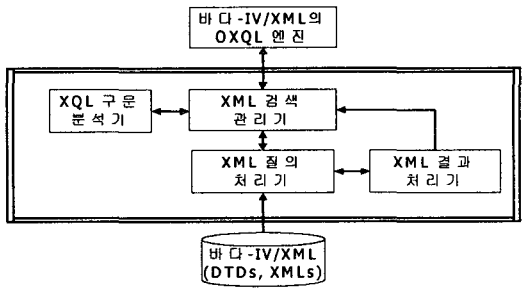

그림 2. 바다-IV/XML 검색기의 구조

4 개의 모듈에 대해 간략히 설명을 하면 다음과 같다.

4.1 XML 구문 분석기

바다-IV/XML 의 OXQL 엔진 모듈에서 전달된 질의문을 구문 분석하여 XQL 파스 트리를 만들어 XML 검색 관리기를 통하여 XML 질의 처리기에 전달한다. XQL 파스 트리를 구성하는 기본 클래스는 XQLNode 클래스이며 각 노드는 가지고 있는 정보에 따라 Unit, Rval, Path, Invocation, Filter, Bang, Param 클래스 등으로 구분된다.

4.2 XML 검색 관리기

XML 검색 관리기는 OXQL 엔진 모듈로부터 XQL 질의 문을 입력 받고, 결과를 OODB 의 OXQL 엔진에 전달해 주는 역할을 한다. 그리고 입력 받은 XQL 질의에

대해 XQL 파스 트리를 만들기 위해 XQL 구문 분석기를 이용하고 질의를 처리하기 위해 XML 질의 처리기를 이용하며 필요한 결과 형태를 만들기 위해 XML 결과 처리기를 이용한다.

4.3 XML 질의 처리기

XQL 구문 분석을 통하여 나온 XQL 파스 트리를 XML 검색 관리기로부터 전달 받고, 전달 받은 XQL 파스 트리를 필요에 따라 질의 처리 계획을 이용하거나 트리를 순회하며 기본 술어 처리나 정보 검색 및 DOM API 에서 제공하는 기본적인 조건 처리 함수들을 차례로 수행하여 결과를 구한다. 이때, XQL 파스 트리의 각 노드를 Unit, Path, Rval, Invocation, Filter, Bang, Param 등을 구분하여 각각 적절한 방식으로 처리한다. 처리 결과는 XML 결과 처리기를 통해 XML 검색 관리기로 전달 된다.

4.4 XML 결과 처리기

XML 질의 처리기에 의해 수행된 결과를 전달 받는다. 이때 전달 받은 내용은 단일형이거나 정보 검색의 결과로 나오는 가중치 값 등을 포함하는 복합형이 될 수가 있다. 그리고, 결과가 하나의 객체가 될 수도 있고 여러 객체의 집합이 될 수도 있는데 이런 여러 가지 결과 유형에 대한 처리를 한다.

5. 바다-IV/XML 검색기의 구현

5.1 바다-IV/XML 의 전체 구조

<그림 3>은 바다-IV 가 XML 저장관리기/검색기와 결합된 전체 구조이다. XML 저장관리기는 클라이언트 부분에 바다-IV/C++ API 를 이용해서 구현이 되어 있고, XML 검색기는 서버 부분에 바다-IV/OK 엔진과 통합되어 구현이 되어 있다.

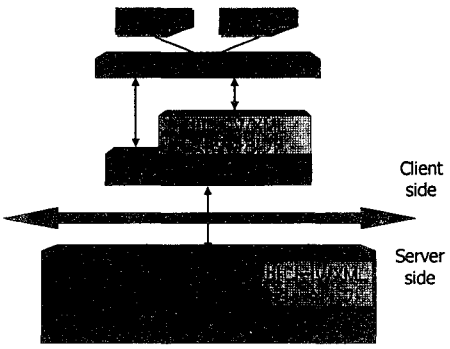

그림 3. 바다-IV/XML 의 전체 구조

5.2 XQL 구문 분석기의 클래스 정의

XQL 구문 분석기는 입력 질의를 노드들에 대한 트리로 분리를 한다. 각 노드의 타입으로는 QUERY, UNIT, RVAL, PATH, BANG, SUBSCRIPT, FILTER, INVOCATION, PARAM 등이 있다. 이들 타입에 대해 각각 클래스로 정의하여 기능을 분리했다. XQL 구문 분석기의 결과인 XQL 파스 트리의 각 노드는 <그림 4>와 같이 XQLNode 를 상속 받는 여러 클래스들로 이루어 진다.

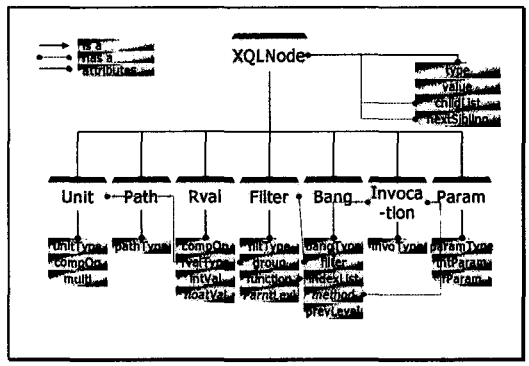

그림 4: XQL 구문 분석기의 클래스 구조

XQL 파스 트리의 루트 노드는 Query 노드이며 전체 질의를 나타낸다. 질의는 Unit 노드의 집합으로 이루어지고, Unit 노드 자체로도 의미있는 질의를 구성할 수 있다. 각 Unit 노드는 다음 타입 중에 하나이다.

- PATH_UNIT : 계층 정보만으로 이루어진 질의를 나타낸다. 예를 들면 /book/title 은 하나의 PATH_UNIT 으로 표현 가능하다.
- COMP_UNIT : 비교 연산자를 포함하는 질의를 나타낸다. 예를 들면 eq ('='), ne ('!=')를 포함하는 질의어는 COMP_UNIT 으로 표현 가능하다.
- AND_UNIT : and 연산자를 포함하는 질의를 나타낸다.
- OR_UNIT : or 연산자를 포함하는 질의를 나타낸다.
- UNION_UNIT : $union$ ('|') 연산자를 포함하는 질의를 나타낸다.
- INTERSECT_UNIT : $intersect$ 연산자를 포함하는 질의를 나타낸다.
- NOT_UNIT : not 연산자를 포함하는 질의를 나타낸다.

Path 노드는 절대 경로와 상대 경로 둘 중에 하나를 표현한다. Path 노드는 질의의 부분 질의 내용을 갖고 있는 Bang 노드로 구성된다. 예를 들면, 하나의 Path 노드로 나타낼 수 있는 /book/title 의 경우 "/"를 기준으로 book 과 title 두개의 Bang 노드를 가지고 있다. Bang 노드는 Filter 노드에 대한 포인터를 가지고 있다. 각 Filter 노드는 다음 타입 중에 하나이다.

- DOT_FILTER : 현재까지의 검색 결과(current search context)를 나타낸다.
- INVOCATION_FILTER : ancestor()와 같은 함수나 메소드를 나타낸다.
- ELEMENT_FILTER : book 이나 title 같은 엘리먼트를 나타낸다.
- STAR_FILTER : 계층 구조에서 한 레벨을 건너 뛴다. 예를 들어, /book/*/title 은 book 의 자식의 자식인 title 엘리먼트를 나타낸다.
- ATTRIBUTE_FILTER : @price 와 같이 애트리뷰트를 나타낸다.
- GROUP_FILTER : 괄호와 같은 그룹 연산자

를 나타낸다.

예를 들어 하나의 Path 노드로 나타낼 수 있는 /book/./*/title 의 경우, <그림 5>에서와 같이 4 개의 Bang 노드와 2 개의 ELEMENT_FILTER, 1 개의 DOT_FILTER, 1 개의 STAR_FILTER 로 구성된 트리로 표현할 수 있다.

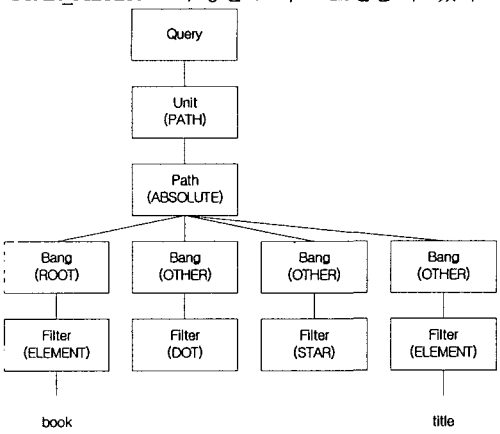

그림 5. XQL 파스 트리 예

Comp 노드는 비교 연산자를 포함하는 질의를 나타내며 Rval 노드를 가리키고 있다. Rval 노드는 비교 연산자를 중심으로 오른쪽에 있는 값을 나타낸다. 각 Rval 노드는 다음 타입 중에 하나이다.

- PATH_RVAL : 오른쪽에 있는 값이 Path 를 나타내는 표현식이다.
- FLOAT_RVAL : 오른쪽에 있는 값이 부동 소수점이다.
- INT_RVAL : 오른쪽에 있는 값이 정수이다.
- TEXT_RVAL : 오른쪽에 있는 값이 문자열이다.

Invocation 노드는 메쏘드 또는 함수 호출을 포함하는 질의를 나타내며, Param 노드를 가리키고 있다. Param 노드는 메쏘드 또는 함수 호출의 인자를 나타낸다. 각 Param 노드는 다음 타입 중에 하나이다.

- UNIT_PARAM : 인자가 하나의 완벽한 질의 형태를 갖는다.
- INT_PARAM : 인자가 정수 형태이다.
- FLOAT_PARAM : 인자가 부동소수 형태이다.
- TEXT_PARAM : 인자가 문자열 형태이다.

위에 정의된 클래스들을 사용하여 모든 가능한 XQL 질의를 트리 형태로 구성할 수 있고, XML 질의 처리기에서는 <그림 5>와 같은 XQL 파스 트리를 이용하여 검색을 수행한다.

6. 결론

본 논문에서는 XML 문서의 내용 정보와 다양한 구조 정보를 검색할 수 있는 검색기를 설계 및 구현하였다. 이 XML 검색기에서 무엇보다 필요한 것은 질의 언어인데, 본 논문에서는 쉽고 사용하기 편한 XQL 을 객체 지향 질의 언어인 OQL 과 결합해서 XML 질의 언어로 제안했다. 이것은 기존의 관계형 SQL 이나 객체형 OQL 에서 XML 을 지원하지 못하는 문제점을 보완한 것이다. 구조 정보인 부모와 자식간의 관계, 조상과 후손간의 관계, 형제들간의 관계에 대한 검색과 속성 검색은 XQL 을 통해서 하고, 내용 검색과 데이터베이스 기능인 문서간의 통합과 조인은 OQL 을 통해 지원한다.

본 논문에서 설계한 XML 검색기는 저장 구조가 DOM 기반으로 되어 있기 때문에 기본 연산은 부모와 자식 노드를 찾는 연산과 이전 형제 노드와 다음 형제 노드를 찾는 연산으로 검색을 한다. 그러므로, DOM 기반의 저장 구조에서 XML 문서를 검색할 수 있는 시스템으로 구현이 되었다. 그리고 DOM 기반의 저장 구조에서 각 노드의 정보에 노드의 ID 를 두어 구조 정보를 검색할 때에 효율을 높였다.

향후 연구 과제로는 XML 문서의 링크 정보에 대한 검색과 결과를 추출할 때 결과를 사용자가 원하는 형태의 XML 문서로 재구성하는 기능과 XML 문서의 그림 등 외부 개체의 검색에 대한 연구가 필요하다.

참고문헌

[1] Tim Bray, Jean Paoli, C. M. Sperberg-McQueen, "Extensible Markup Language (XML) 1.0", http://www.w3.org/TR/REC-xml-19980210/, 1998

[2] 이미영, 허대영, 김명준, "바다-III 멀티미디어 DBMS 설계", 한국전자통신연구소

[3] V. Apparao, S. Byrne, et al., "Document Object Model (DOM) Level 1 Specification", http://www.w3.org/TR/REC-DOM-Level-1/, 1998

[4] R. G. G. Cattell, "The Object Database Standard : ODMG-93", Morgan Kaufmann, San Francisco, California, 1994

[5] Jonathan Robie, Joe Lapp, and David Schach, "XML Query Language (XQL) ". http://www.w3.org/TandS/QL/QL98/pp/xql.html, 1998

[6] 김용훈, "다양한 구조 검색을 지원하는 XML 검색기의 설계 및 구현", 충남대학교 석사학위 논문, 1999

[7] Stephen Deach, "Extensible Stylesheet Language (XSL) 1.0", http://www.w3.org/TR/2000/WD-xsl-20000301/, 2000

[8] Alin Deutsch, Mary Fernandez, Daniela Florescu, Alon Levy, Dan Suciu, "XML-QL : A Query Language for XML", http://www.w3.org/TR/NOTE-xml-ql, 1998