Grove를 이용한 구조적 SGML 문서의 저장 및 검색

(A Storage and Retrieval System for Structured SGML Documents using Grove)

김 학 균 † 조 성 배 ††

(Hak-Gyoon Kim) (Sung-Bae Cho)

요 약 플랫폼에 관계없이 한번 작성된 문서의 정보를 이기종 시스템간 공유하고 다양한 문서 형식을 지원하기 위해 SGML(ISO 8879)이 사용되고 있다. SGML 문서는 내용뿐만 아니라 구조정보를 가지고 있다. SGML 문서가 널리 보급됨에 따라서 구조적 정보를 이용한 데이타베이스의 구축 및 검색 시스템에 대한 필요성이 고조되고 있다. 그러나, 기존의 색인어를 이용한 전문 검색 엔진으로는 문서의 구조정보를 활용할 수 없다. 본 논문에서는 DSSSL 및 HyTime의 문서 모델인 Grove를 변형한 데이타 모델을 이용하여 문서 형식에 독립적이면서, 문서 형식과 내용을 분리하여 저장하는 SGML 문서 저장 시스템을 개발하였다. 구조정보를 손실없이 저장할 수 있도록 객체 지향형 데이타베이스 시스템인 오브젝트 스토어(Object Store)를 이용하였다. 또한 엘리먼트에 대해 관계형 DBMS와 유사한 인덱스 구조를 생성하여 검색 성능을 향상시켰고, 내용기반 검색과 구조기반 검색을 효율적으로 결합한 사용자 인터페이스를 구축하였다.

키워드 : SGML, Grove, 구조기반 검색, 객체지향형 DB

Abstract SGML(ISO 8879) has been proliferated to support various document styles and to transfer documents into different platforms. SGML documents have logical structure information in addition to contents. As SGML documents are widely used, there is an increasing need for database storage and retrieval system using the logical structure of documents. However, traditional search engines using document indexes cannot exploit the logical structure. In this paper, we have developed an SGML document storage system, which is DTD-independent and store the document type and the document instance separately by using Grove which is the document model for DSSSL and HyTime. We have used the Object Store, an object-oriented DBMS, to store the structure information appropriately without any loss of structural information. Also, we have supported a index structure for search efficiency like the relational DBMS, and constructed an effective user interface which combines content-based search with structure-based search.

Key words : SGML, Grove, Structure-based search, object-oriented DB

1. 서 론

컴퓨터를 이용한 문서의 생성 및 교환이 보편화되어감에 따라, 플랫폼에 관계없이 한 번 작성된 문서 정보를 이기종 시스템간에 공유할 수 있는 데이타베이스의 구축 및 검색의 중요성이 날로 증가하고 있다. 또한, 문서는 메모, 전자 메일, 작업 매뉴얼, 공문서 등의 다양한 형식으로 되어 있기 때문에, 이를 형식에 따라 효율적으로 관리 및 공유하기 위해서는 문서를 일관성 있게 구조화하는 기술이 필요하다. 이에 ISO에서는 SGML(Standard Generalized Markup Language)을 문서의 구조 표준안으로 제정하였다[1]. 이를 이용하여 현재 HTML(Hyper Text Markup Language), HyTime(Hypermedia/Time-based Structuring Language)[2], TEI(Text Encoding Initiative)[3] 등이 표준 문서안으로 되어있다.

SGML 문서는 내용이외에도 문서의 구조와 레이아웃 등의 문서 형식을 정의하는 정보를 가지고 있다. 이러한 정보를 문서에 추가한 것을 마크업(markup)이라 하는데,

· 이 논문은 2002년도 한국학술진흥재단의 지원에 의하여 연구되었음 (KRP-2000-005-C00012).

† 비 회 원 : KT 서비스개발연구소 연구원
playbug@candy.yonsei.ac.kr
†† 종신회원 : 연세대학교 컴퓨터과학과 교수
sbcho@cs.yonsei.ac.kr

논문접수 : 1999년 4월 16일
심사완료 : 2002년 6월 25일

그 역할은 문서를 구성하고 있는 논리적인 단위들을 분리하고 각 단위에 적용되어야 할 기능을 지정하는 것이다. SGML은 이와 같은 마크업 언어를 정의하는 메타 언어로서, 다양한 마크업 언어를 정의할 수 있도록 한다.

SGML 문서는 구조적으로 연관된 엘리먼트가 서로 복잡하게 연결된 계층형 모델이기 때문에, 이를 효과적으로 모델링하기 위하여 다양한 모델이 제시되고 있다. 이때 SGML 문서의 데이타모델은 다음과 같은 조건을 만족시켜야 효과적인 구조 기반 저장 및 검색이 가능하다 [4]. 첫째, 엘리먼트 단위의 문서 구조정보 탐색이 가능해야 한다. 둘째, SGML 문서와 데이타베이스의 저장구조가 유사해야 한다. 셋째, 모든 문서 형식을 지원해야 한다. 넷째, 대용량의 데이타 관리가 가능해야 한다. 위의 조건을 만족시키기 위해 본 논문에서는 문서를 효과적으로 관리할 수 있도록 엘리먼트 단위로 SGML 문서를 객체지향형 데이타베이스에 저장 및 검색한다. SGML 문서는 다양한 형태의 문서 형식에 따라 생성되기 때문에, 모든 문서 형식을 지원할 수 있는 DSSSL(Document Style Semantics and Specification Language)[5] 및 HyTime의 문서 모델인 Grove(Graph Representation Of property ValuEs)를 사용하여 문서를 모델링한다.

본 논문의 구성은 다음과 같다. 2장에서는 논문에 관련된 배경 지식에 대해 설명하고, 3장에서는 전체 시스템에 대해 간략하게 기술하고, 4, 5장에서는 SGML 문서를 데이타베이스에 저장하고 검색하는 방법에 대해 정리한다. 그리고, 6장에서는 구현 결과를 고찰하여, 이에 대한 결론을 맺는다.

2. 배 경

2.1 SGML

SGML은 ISO 8879-1986에서 정의된 규칙들을 이용하여 마크업 언어의 구조를 정의하고, 그에 따라 문서 내용에 마크업을 하는 것이다. SGML 문서는 크게 선언부, 형식 정의부, 실제 문서부로 이루어져 있다[1].

(1) 선언부

SGML 문서의 선언부는 문서에서 사용되는 문자집합, 특수 문자에 대한 설명, 그리고 태그 삭제기능과 같은 여러 가지 기능 정보를 가지고 있다. SGML 문서의 맨 앞에 위치하여 정보를 전달하기 때문에 시스템 사이의 호환성을 유지할 수 있다.

(2) 문서 형식 정의부

문서 형식 정의부(Document Type Declaration: DTD)는 엘리먼트, 엔터티 및 속성 등의 세 가지 주요 구성요소를 사용하여 문서의 논리적인 구조를 정의한다. 엘리먼트는 문서의 논리적 단위를 나타내며, 태그를 통하여 문서 내용을 마크업한다. 각 엘리먼트는 현재 엘리먼트 안에 포함되는 엘리먼트 혹은 엘리먼트의 집합체, 즉 모델그룹을 이용하여 내용모델을 정의한다. 엔터티는 하나의 단위로서 참조되는 문자의 집합으로 문서 내용 중 일부분을 코드화하여 이름을 붙일 수 있도록 하며 특수 문자의 표현도 가능하다. 저장 위치에 따라 내부 엔터티와 외부 엔터티로 나뉘어진다. 속성은 엘리먼트의 시작 태그에 첨가되는 것으로, 엘리먼트 고유의 정보를 담고 있다.

세 가지 구성요소 이외에도 엘리먼트나 엔터티의 처리 방법을 나타내는 처리 명령어(Processing Instruction) 선언, 노테이션(Notation) 선언, 마크업을 편하게 하기 위한 단축참조표(Short Reference Map) 선언 등이 있다. 그림 1은 문서 형식 정의부의 예이고, 그림 2는 정의된 형식을 트리 형태로 보여준 것이다. 트리의 노드는 내용 모델의 정의에 따라 재귀적으로 반복될 수 있는 속성을 가지고 있다.

(3) 실제 문서부

실제 문서부(Document Instance: DI)는 SGML 문서의 내용이 들어가는 부분으로, DTD에 따라 작성된 구조화 문서이다. 각 엘리먼트에 대해 해당되는 문서 내용을 포함하여, 전체 문서는 엘리먼트들의 노드로 구성되는 일종의 트리로 표현된다.

```
<!DOCTYPE article[
<!ELEMENT article                    - -     (title, author+, abstract, section+, ack)>
<!ATTLIST article status (final|draft) draft>
<!ELEMENT title                      - -     (#PCDATA)>
<!ELEMENT author, abstract           - O     (#PCDATA)>
<!ELEMENT section                    - O     (title,body*, subsection*)>
<!ELEMENT subsection                 - O     (title, body+)>
<!ELEMENT body          - O          (figure|paragraph)>
<!ELEMENT figure        - O          (artwork,caption?)>
<!ATTLIST figure        label        ID                  #IMPLIED>
<!ELEMENT artwork                    - O     EMPTY>
<!ATTLIST artwork                    sizex   NMTOKEN             "16cm"
                                     sizey   NMTOKEN     #IMPLIED
                                     file    ENTITY              #IMPLIED>
<!ELEMENT caption                    O O     (#PCDATA)>
<!ELEMENT paragraph, ack             - O     (#PCDATA)   > ]>
```

그림 1 형식 정의부의 예

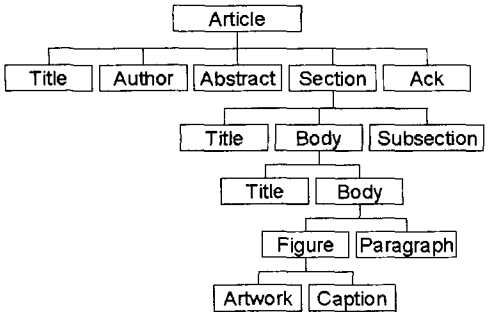


그림 2 트리로 표현된 형식 정의부

2.2 SGML 문서를 위한 데이타 모델

SGML 문서는 구조적으로 연관된 엘리먼트가 서로 복잡하게 연결되어 있기 때문에 다양한 모델들이 제시되었다. 특히, SGML 구조정보의 대상 데이타베이스로의 전환, SGML 문서의 빠른 처리를 위한 적절한 인덱스 구조의 생성 등이 큰 문제로 부각되고[6], 또한 구조정보에 대한 질의 인터페이스의 부재라는 문제가 있다[7]. 본 절에서는 현재 연구되고 있는 SGML 문서의 데이타베이스 모델에 대해 문서기반 모델과 엘리먼트 기반 모델로 나누어 살펴본다.

(1) 문서 기반 모델

기존의 전문 검색 시스템들은 문서에 색인어를 부여하여 검색을 가능케 한다. 문서로부터 색인어를 추출하고, 사용자의 질의에 사용된 단어와 문서 색인어 사이의 유사성을 계산하여 결과를 제공한다[6]. 이러한 시스템은 문서를 단순히 단어의 집합으로 보기 때문에 구조정보가 무시되는 문제가 있다[7].

(2) 엘리먼트 기반 모델

엘리먼트 기반 모델은 전체 문서를 엘리먼트 단위로 분해하고, 엘리먼트에 대해 직접 질의를 수행하는 방법을 취한다. 문서의 구조에 기반하여 정보를 저장하기 위해 많은 수의 테이블과 튜플이 필요하다는 단점이 있기 때문에[8], 이에 대한 해결책으로 확장 관계형 모델[4, 8,9]이나 오브젝트 모델[10,11,12]로 표현하는 방법이 있다.

관계형 모델로는 일반적인 스키마를 이용해 문서의 DTD에 따라 관계형 모델의 레코드 구조를 정의하는 방법이 있다. 이 방법은 자동으로 DTD를 분석하여 모델을 생성해주기 때문에 사용자는 DTD에 기술된 것 이외에는 문서를 엘리먼트로 분해하는 것에 대한 정보를 알 필요가 없으나 DTD에 나타난 문서구조 전체가 레코드의 구조로 직접 사용되기 때문에 검색범위가 효율적이지 못하다[9]. 또한 구조의 네스팅(nesting)과 참조(reference)를 지원하는 확장 관계형 모델은 서브 엘리먼트가 없는 엘리먼트에 대한 필드의 반복은 지원하나, 서브 엘리먼트를 포함하고 있는 엘리먼트의 반복은 지원하기 어렵다[4].

INRIA의 VERSO 프로젝트는 O2 객체지향형 데이타베이스를 이용하여 SGML 문서의 DTD를 엘리먼트 단위의 O2 스키마로 변환하고 문서를 관련 객체와 값으로 매핑시켰는데, DTD에 정의된 각각의 엘리먼트는 타입과 제약조건 및 디폴트 연산자를 갖고 있는 클래스로 해석된다[10]. 독일의 GMD-IPSI에서는 다양한 DTD 구조를 동적으로 데이타베이스에 반영하는 연구를 하였다. DTD에 대한 정보를 표현하기 위하여 DTD에 대한 상위 DTD(Super-DTD)를 정의하여 DTD를 상위 DTD의 인스턴스 형태로 표현하였다. 이를 통해 특정 DTD에 관계없이 SGML 문서를 엘리먼트 단위의 오브젝트로 저장하였다. 그리고, 문서의 엘리먼트를 구조적 엘리먼트와 비구조적 엘리먼트로 나누어 비구조적인 엘리먼트는 상위 구조적 엘리먼트의 오브젝트에 포함되어 있는 구조를 가진다[11]. 또한 SGML 문서의 논리적 구조 모델을 DTD 독립 스키마 모델과 DTD 의존 스키마 모델로 나누어 설계한 모델과[9], 질의는 주로 단말(terminal) 엘리먼트에서 발생된다는 판단하에 문서 내용 모델은 크게 단말 엘리먼트와 비단말 엘리먼트로 나누어 설계한 객체 지향형 시스템이 있다[13]. 이 모델은 객체 지향형 데이타베이스인 O2를 사용하여 계승이나 메소드 등과 같은 객체 지향형 개념의 여러 특성을 지원하며, O2에서 제공하는 질의어 OQL을 이용하였다.

2.3 Grove

Grove(Graph Representation Of property ValuEs)는 DSSSL 및 HyTime의 문서 모델로서, SGML문서의 추상 데이타 구조를 표현하는데 사용된다. 따라서, Grove 구조는 SGML 문서의 구조 및 내용정보를 추상화시키는데 적합하며 SGML 문서와 역변환될 수 있다는 장점이 있어 DSSSL 처리 시에 SGML을 나타내는 입력값으로 사용되고 출력에 있어서도 SGML 문서를 생성해 낼 중간 자료구조의 역할을 한다[5].

Grove는 방향성을 갖는 복잡한 트리 구조로 표현되며, 각 노드는 DSSSL 표준안의 SGML 특성 집합(SGML property set)에 정의된 클래스의 인스턴스이다. SGML 특성 집합은 SGML 문서의 모든 정보를 표현할 수 있도록 총 71개의 클래스로 구성되어 있으며, 이러한 클래스들은 크게 Declaration, Prolog, Instance 등의 모듈로 분류된다. 생성된 Grove의 각 노드는

SGML 특성 집합의 특정 클래스의 인스턴스로서, 속성 이름과 값을 가진 속성 집합으로 구성된다. 즉, 속성 집합은 특정 클래스의 구조정보 및 내용정보를 기술하는 역할을 한다. 속성의 정의로 가능한 데이타 타입에는 부울형이나 문자열등과 같은 기본적인 타입 이외에도 노드와의 관계를 이어주는 노달(nodal) 데이타 타입이 있다.

3. 문서의 구조기반 저장 및 검색 시스템 개요

SGML 문서와 같이 계층형으로 표현되는 문서에 대해 다양한 데이타 모델이 등장하여 앞서 제시된 문서 기반 모델과 엘리먼트 기반 모델 등이 있다. 특히, 엘리먼트 기반 모델은 크게 관계형 DBMS로 설계된 것과 객체지향형 DBMS로 설계된 것으로 나눌 수 있다. 관계형 DBMS로 설계된 모델을 계층형 모델로 표현하기 위해서는 많은 수의 테이블이나 튜플이 필요하기 때문에, 문서의 구조정보의 손실을 감내하면서 검색 기능에 초점을 맞추어 설계되고 있다. 반면에 객체지향형 DBMS로 설계된 모델은 참조(reference)를 이용하여 SGML 문서와 같은 데이타 구조를 쉽게 표현할 수 있다. 그러나 구조정보 및 내용정보에 대한 인덱스가 부재하기 때문에, 구조정보에 기반한 내용 검색의 성능이 떨어지는 단점이 있다.

따라서, 본 논문은 객체지향형 DBMS로 문서를 모델링하여 구조정보를 손실없이 저장하며, 적절한 형태의 인덱스 구조를 제공하여 효율적인 검색 효과를 보여주는 것을 목표로 한다. 이를 위해 먼저 구조정보와 내용을 효과적으로 표현하는 Grove를 사용한다. Grove는 SGML 문서의 모든 정보를 모델링하여 실제로 시스템에 적용하는데 무리가 있어, 본 논문에서는 Grove의 주요 클래스들을 추출 및 재배치한 데이타 모델을 생성한다. Grove는 구조정보와 내용정보를 관리하는 범용적인 클래스들로 구성되어 있기 때문에, 구조정보와 내용이 분리되어 SGML 문서를 관리 및 저장할 수 있으며, DTD에 독립적으로 문서를 관리할 수 있는 장점을 가진다. 그리고 검색 성능의 향상을 위해 주요 클래스에 인덱스 즉, *Extents*를 생성하였다. 인덱스에 질의를 보내 내용 검색을 한 뒤에 검색된 결과에 대하여 구조정보 검색을 하여 구조에 기반한 내용 검색을 효과적으로 수행한다. 또한, 시각적인 질의 인터페이스를 통해 문서의 구조정보 검색과 구조기반 내용 검색을 효과적으로 결합하였다. 다음은 본 논문에서 개발된 SGML 문서의 저장 및 검색 시스템에 적용된 특징들을 간략히 정리한 것이다.

- DTD에 독립적인 데이타 모델 제시
- 문서 형식과 내용을 분리하여 저장하여 효율적인 문서관리가 가능
- 문서의 구조정보 검색 및 구조에 기반한 내용 검색
- 구조정보와 이에 기반한 내용정보 검색이 결합된 인터페이스 제공
- 논리적으로 분리된 내용에 인덱스를 생성하여 검색의 효율 증대

개발된 SGML 문서의 저장 및 검색 시스템은 Object Store를 사용하여 Windows 98에서 Visual C++ 5.0으로 구현되었다.

4. 문서의 구조기반 저장

문서의 구조에 따른 저장은 그림 3과 같이 먼저 SGML 문서를 파싱하여 Grove를 생성하고 데이타베이스 생성 모듈에서 영속적인 데이타로 변환한다.

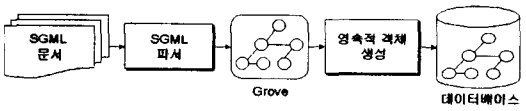


그림 3 구조에 기반한 저장 시스템

4.1 SGML 문서의 데이타 모델

SGML 문서의 파싱은 James Clark의 SP 파서[14]를 가공하여 SGML 문서의 데이타 모델을 생성하였으며, 구조는 그림 4와 같다.

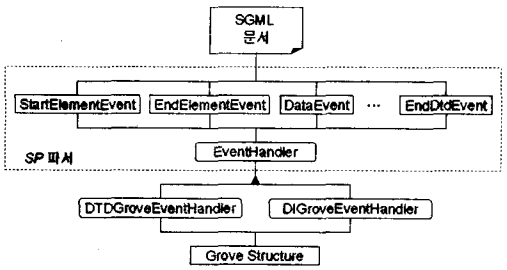


그림 4 SGML 문서의 파싱

SP 파서는 Event-driven 방식으로서, XML의 문서 모델인 SAX(Simple API for XML) 모델[15]과 유사한 인터페이스를 가진다. SP 파서의 Event 관련 클래스들은 DTD 및 DI 정보에 대한 Event를 발생한다. 즉, 특정 엘리먼트의 시작과 종료 등을 가리키는 Event를 발생하며, 그 정보는 EventHandler를 통해 얻을 수 있다. 따라서, EventHandler 클래스로부터 각각 DTD와 DI 정보를 가공하여 추출하기 위해서 DTDGroveEvent

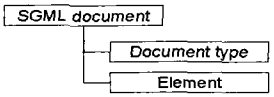


그림 5 SGML 문서의 구조

Handler와 DIGroveEventHandler를 상속받는다. 상속받은 EventHandler는 Event에 따라 물리적인 데이타 구조를 얻어내어, Grove의 형식에 따라 데이타 모델을 생성한다. 생성된 데이타 모델의 최상위 클래스인 SGML document 클래스의 구조는 그림 5와 같다. Document type은 DTD의 정보를 담고 있는 최상위 클래스이며, Element는 DI의 정보를 담고 있는 클래스이다.

(1) 문서 형식 모델

문서 형식 모델은 SGML 문서가 포함하고 있는 여러 구조정보를 손실 없이 표현하며 DI의 구조를 결정하는 역할을 한다. 문서 형식 모델은 Document type 클래스가 관리하며 Element type, Model group, Element token, PCDATA token, Attribute definition 클래스 등으로 구성되어 있다. Element type 클래스는 내용모델, 속성 정보, 배제 및 포함 엘리먼트 정보 등을 가지고 있으며, Attribute Definition 클래스는 엘리먼트의 속성을 정의한다. Model group 클래스는 그룹간의 연결자(connector) 및 출현 빈도수(occurrence) 정보를 가지고 있는데, 하위 엘리먼트의 내용 모델을 담고 있는 Model group 클래스와 단말 노드인 Element token 클래스 및 PCDATA token 클래스의 집합으로 구성되어 있다. 그림 6은 문서 형식 모델을 보여주는데, type은 문서의 DTD에 따라 동적으로 연관관계가 형성되는 것을 의미한다.

(2) 문서 내용 모델

문서 내용은 문서 형식 모델의 문법에 따라 저장되며, 계층적으로 연결된 Element 클래스의 집합으로 구성되

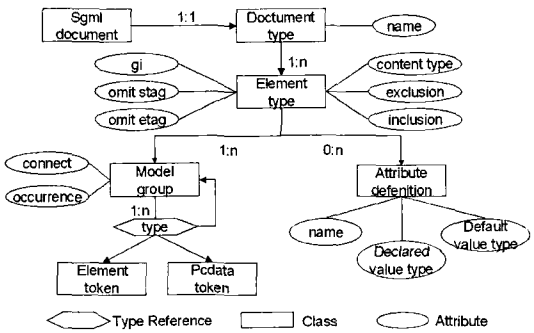


그림 6 문서 형식 모델

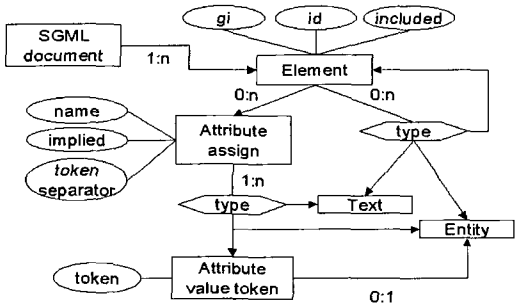


그림 7 문서 내용 모델

어 있다. Element 클래스는 엘리먼트의 속성 정보를 담고 있는 Attribute assign 클래스와 엘리먼트에 대한 내용을 담고 있는 Text 클래스, 내용 모델에서 정의된 순서에 따라 결정되는 하위 Element 클래스로 구성되어 있다. 위와 같이 구성된 클래스는 최상위 노드와 부모 노드 등의 공통된 속성 정보를 갖는다. 그림 7은 문서 내용 모델을 보여준다.

(3) 엔터티 모델

내부 및 외부 엔터티는 Entity 클래스가 관리하며, 외부 엔터티일 경우 External id 및 Notation 클래스가 사용된다. 외부 엔터티는 직접 저장되는 것이 아니라 이에 대한 ID만을 취하는 구조를 가진다. 엔터티는 Element 클래스 혹은 Attribute assign 클래스에 의해 참조된다. 그림 8은 엔터티의 구조를 보인다.

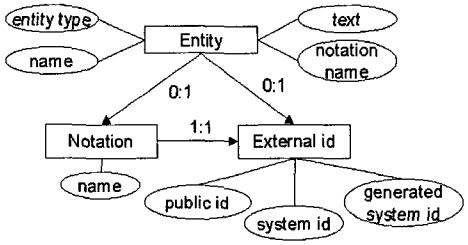


그림 8 엔터티 모델

4.2 데이타 모델의 저장

위와 같은 계층적인 데이타 모델을 기존의 관계형 데이타베이스의 테이블 구조로 전환하기는 어려우나 객체지향형 데이타베이스인 Object Store를 사용하면 호스트 프로그래밍 언어와 데이타베이스의 통합을 통해 호스트 프로그래밍 언어에서 만들어진 클래스의 구조를 바로 데이타베이스에 적용할 수 있다. 또한 객체간의 포인터(Object ID)를 통해 연관 관계가 정의되므로 유연

한 클래스 구조를 지원한다. 정의된 모델을 데이타베이스에 저장하기 위해서는 그림 9와 같은 과정을 거치고, 영속화된 객체를 생성하기 위해서는 다음과 같은 과정을 거친다.

1. 클래스 변수를 영속적 변수로 변환
2. 클래스간의 관계를 reference와 relationship으로 변환
3. 클래스 타입을 Object Store에 등록
4. 기존의 new 연산을 영속적인 new(persistent new) 연산으로 변환

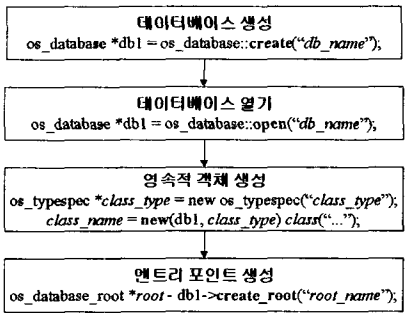


그림 9 Object Store 개발 인터페이스

5. 문서의 구조기반 검색

그림 10은 검색 시스템의 구조를 보인다. 구조 기반 사용자 인터페이스를 통해 질의를 입력받고 생성된 구조정보를 이용하여 질의어를 생성한다. 생성된 질의어는 데이타베이스를 통해 처리되고 결과는 사용자 인터페이스를 통해 확인할 수 있다.

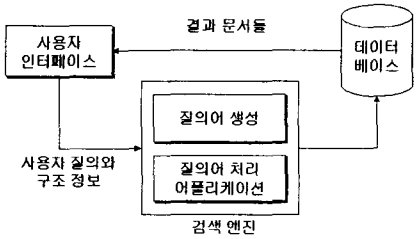


그림 10 검색 시스템

5.1 질의 언어

Object Store에서 제공하는 질의 언어는 기본적으로 객체간의 관계를 통해 이루어진다. SGML과 같이 구조적으로 이루어진 문서에 대한 질의는 기존의 전문 검색 엔진의 질의 이외에 여러 가지 질의 형태가 추가될 수 있다. SGML 문서의 질의 유형은 다음과 같다[4].

(1) 부울형 검색

문서 전체내용에 대해 기존의 불리언 검색을 수행하는 방법이다. 이와 같은 질의를 제공하기 위해서 Text 클래스를 이용한다. Text 클래스는 SGML 문서의 모든 데이타가 포함되어 있기 때문에 문서 전체에 대한 질의가 가능하다. 다음의 질의는 불리언 검색을 지원하는 형태이다.

[질의 1] "SGML"과 "OODBMS"를 담고 있는 모든 문서를 검색하라.

1. 모든 텍스트 오브젝트에 대해 그 값이 "SGML"과 "OODBMS"를 담고 있는지 검사

(2) 문서의 구조정보 검색

문서 형식이 어떻게 구성되어 있는지를 검색하는 질의 형태이다. 따라서 문서의 내용정보뿐만 아니라 문서의 논리적 구조정보도 데이타베이스에 저장되어야 한다.

[질의 2] <section>의 부모 엘리먼트를 검색하라.

1. 엘리먼트 타입 오브젝트 중에 section을 검색
2. 해당 오브젝트의 부모 이름을 출력

(3) 구조정보 기반 내용 검색

검색의 범위를 문서의 특정 지역에 제한하는 질의 형태이다. 즉, 문서의 특정 엘리먼트에 대해 질의를 수행한다. 이러한 질의는 문서를 계층적으로 엘리먼트 단위로 분리 저장함으로써 가능하다.

[질의 3] <section>의 <title>이 "Grove"를 담고 있는 문서를 검색하라.

1. 엘리먼트 오브젝트 중 이름이 title이고 그 부모가 section인 것을 검색
2. 선택된 엘리먼트 오브젝트에 종속된 모델그룹 오브젝트와 텍스트 오브젝트가 "Grove"를 포함하는지 검사

(4) 엘리먼트 속성 검색

엘리먼트의 내용뿐만 아니라, 엘리먼트의 속성에 대한 질의가 가능하다. 속성정보는 해당 엘리먼트의 Attribute assign 클래스에 저장되어 있다.

[질의 4] <picture>의 file 속성의 해당 값이 sgml.gif인 것을 검색하라.

1. 엘리먼트 오브젝트 중에 이름이 picture인 것을 검색
2. 해당 엘리먼트 오브젝트의 Attribute assign 오브젝트의 이름이 file인 것을 검색
3. Attribute assign의 값이 sgml.gif인 것을 검색

5.2 질의 인터페이스

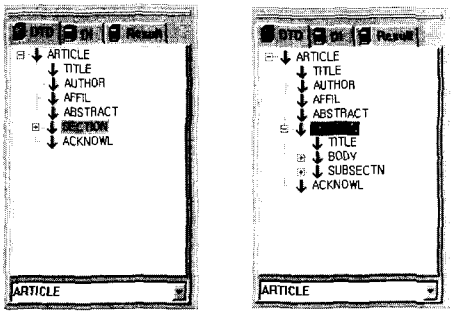
SGML 문서는 데이타베이스에 문서의 구조에 관한 정보와 구조에 따른 내용정보가 저장되어 있다. 이를 고

려하여 검색하는 방법에는 문서의 논리적인 구조정보의 검색과 문서의 구조정보에 기반한 내용정보의 검색이 있다. 이를 효과적으로 결합하기 위하여 문서 구조정보를 동적으로 생성하여 이에 대하여 질의를 제공하는 인터페이스를 제시한다.

(1) 구조정보의 검색

문서의 구조정보 검색은 SGML 문서의 DTD 정보를 최상위의 엘리먼트로부터 최하위의 엘리먼트까지 탐색할 수 있어야 한다. DTD 정보는 엘리먼트를 이용해 문서의 논리적 계층 구조를 표현하므로, 동적인 트리 구조로 표현될 수 있다. 트리의 노드는 Element type 오브젝트를 의미한다. 그림 11은 검색 인터페이스의 화면으로, 문서 트리 구조의 특정 엘리먼트를 클릭함으로써 하위 엘리먼트를 시각적으로 검색할 수 있다. 또한 검색 인터페이스는 등록된 여러 가지의 DTD를 선택함으로써 원하는 DTD에 대해 검색할 수 있다.

그림 12는 Section 엘리먼트의 내부 구조를 보여준다. *Section*이라는 *Element type* 오브젝트는 하위 엘리먼트 구조를 정의하는 모델 그룹을 가진다. 모델 그룹은 *Element token*의 리스트로 표현되어 하위구조의 관계를 정의한다. 각각의 Element token은 자신을 가리키는 Element type, Sibling 엘리먼트와의 연결 방법(*Connector*)과 엘리먼트의 발생 빈도(*Occurrence*) 등을 가지고 있다. 따라서, Element token 오브젝트에 적용되는 Element type 오브젝트를 참조하여 문서의 구조정보를 검색한다.



(a) Article DTD의 트리 구조 (b) Section 엘리먼트의 검색

그림 11 문서 구조 검색의 인터페이스

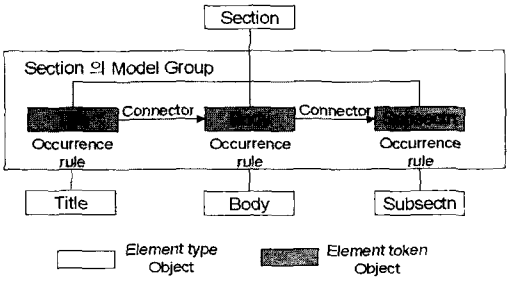


그림 12 문서의 구조정보 표현방법

(2) 구조기반 내용 검색

문서의 구조에 기반한 내용 검색은 기존의 전문 검색과는 달리 검색의 범위를 문서의 일부분으로 한정시키는 것을 의미한다. 앞서 제시된 Article DTD의 경우, 요약(abstract), 논문저자(author), 단락제목(section title) 등의 각각의 엘리먼트에 대해 질의를 수행할 수 있다. 예를 들어, Title 엘리먼트에 특정 검색어가 제시되었을 때 그림 13과 같이 구조적 정보를 가진 검색 패턴과 이에 해당되는 검색 범위를 구성한다. 효과적인 구조 검색을 위해 검색이 자주 일어나는 클래스에 대한 인덱스 즉, *Extents*를 생성하여, 관계형 DBMS와 유사한 형태의 질의를 수행할 수 있는 구조를 제공한다. 본 검색 엔진은 기존의 관계형 DBMS와 같은 질의 형태를 이용하여 Text 오브젝트에 대해 질의를 수행한 뒤, 검색된 결과에 대해 구조정보에 대한 질의를 매핑시켜 최종적인 결과를 얻는다.

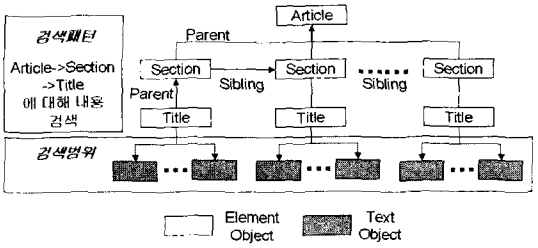
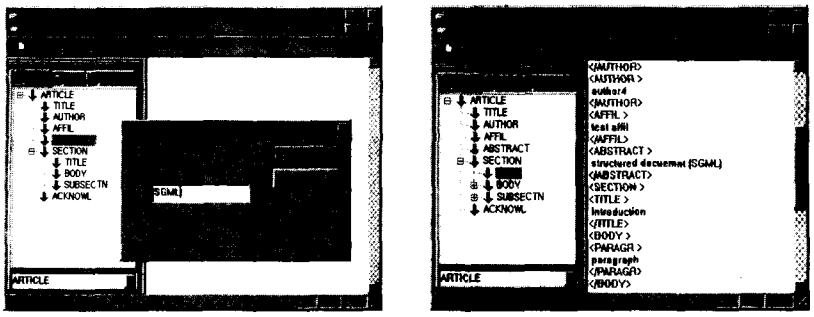


그림 13 검색패턴과 검색범위

사용자는 그림 14 (a)와 같이 문서의 구조적 정보 인터페이스를 통해 특정 엘리먼트에 질의어를 입력하여, 구조적 정보를 담고 있는 검색 패턴과 검색 범위와 일치하는 문서의 결과 목록을 제시한다. 결과 목록은 그림 14 (b)의 Result 탭에 제시된다. 문서의 목록에 있는 각각의 문서들은 데이타베이스에 저장된 논리적 단위를 재구성하여 보여질 수 있다.

(3) 엘리먼트 속성 검색

엘리먼트는 정의된 문서 형식에 따라 속성을 가질 수 있다. 속성을 가진 엘리먼트에 대한 질의는 구조에 기반한 내용 검색의 질의형태와 비슷하다. 즉, 특정 엘리먼



(a) 특정 구조 엘리먼트에서 검색 (b) 검색 결과

그림 14 문서의 구조에 따른 검색

트에 대한 다이얼로그 박스가 뜰 때, 속성 정보를 묻는 텍스트 박스가 함께 제시된다. 속성을 가진 엘리먼트는 Attribute assign 오브젝트를 가지고 있는데, 이 정보와 일치하는 문서를 검색한다.

6. 결 론

기존에 제안된 SGML 문서의 저장 및 검색 시스템은 문서의 구조 정보를 저장하는 것에 초점을 맞추어, 문서의 배치 정보는 무시되는 경향이 있다. 이러한 문제점으로 인해 하나의 SGML 문서에 대해 구조적으로 쪼개진 문서와 원래 문서 두가지 형태로 저장되었다. 그러나, 본 논문은 Grove라는 표준 문서 모델을 통해 문서의 구조, 배치, 내용 정보를 모두 저장하여 구조적으로 쪼개진 문서를 손실 없이 그대로 원래 문서로 저장될 수 있는 장점을 가진다. 또한, Grove라는 문서 모델은 XML(eXtended Markup Language)에도 적용이 가능하기 때문에 XML 문서에도 쉽게 적용될 수 있는 장점을 가진다.

본 논문에서는 DSSSL 및 HyTime의 문서 모델인 Grove를 이용하여 계층형 트리로 표현되는 SGML 문서를 모델링하고, 객체지향형 데이타베이스인 Object Store로 저장 및 검색하는 시스템을 개발하였다. 여기에서 개발한 시스템은 GMD-IPSI[12]나 SERI[13]에서 개발된 기존의 시스템과 차별되는 특징을 갖는데, 다음과 같이 요약할 수 있다.

첫째, 특정 DTD에 독립적인 데이타 모델을 제시하였다. 문서 모델은 크게 개별 DTD에 따른 것과 모든 DTD에 적용되는 범용적인 것으로 나눌 수 있다. 개별 DTD에 따른 모델은 구조정보를 효과적으로 저장할 수 있는 장점이 있는 반면에, 새로운 DTD가 삽입될 때마다 여러 제약 조건을 가지는 클래스를 생성해야하는 단점이 있다. 특정 DTD에 종속되지 않는 모델은 DTD를 자동으로 파싱하여 분석하는 장점이 있지만 효율적인 데이타 관리가 쉽지 않다. 본 논문은 특정 DTD에 종속되지 않는 모델을 제시하기 위해서 Grove를 이용하여 구조정보를 효과적으로 저장 및 관리하였다.

둘째, 문서 형식 모델과 문서 내용 모델을 분리하여 모델링하여 구조정보에 충실한 효과적인 데이타 모델을 제시하였다. 문서 형식과 내용 모델이 결합된 문서 모델은 문서를 저장할 때 문서구조를 반영하기 위해 과다한 제약 조건을 가지고 있어서 검색의 효율이 낮다. 본 논문에서는 문서 구조 모델과 문서 내용 모델을 분리함으로써 문서 내용 모델은 문서 형식 모델에 기반한 계층형 트리로 구성된다. 따라서 검색 인터페이스의 문서 구조 트리와 동일한 형태로 저장되어 있기 때문에 검색의 성능이 향상된다.

셋째, 시각적인 질의 인터페이스를 통해 문서의 구조정보 검색과 구조기반 내용 검색을 효과적으로 결합하였다. 구조정보 검색창을 통해 문서 구조정보를 동적으로 순회할 수 있으며, 생성된 구조정보 트리는 문서 내용 모델과 동일한 형태를 가지고 있기 때문에 엘리먼트별 검색이 가능하다.

넷째, 검색이 자주 일어나는 클래스에 *Extents*를 두어 보다 빠르게 데이타를 관리하고 검색할 수 있도록 하였다. 객체지향형 데이타베이스의 가장 큰 단점중의 하나는 관계형 데이타베이스처럼 특정 테이블을 뷰잉하기가 쉽지 않다는 점이다. 따라서 이러한 기능을 제공하기 위해 *Extents*를 지원하였다. Object Store는 *Extents*를 지원하지 않기 때문에, 검색이 자주 일어나는 클래스에 대해 *Extents*를 두어 기존의 관계형 데이타베이스와 같이 데이타에 접근할 수 있다.

현재 본 논문은 텍스트 기반의 SGML 문서를 대상으로

저장 및 검색 시스템을 제안하였으나, SMIL(Synchronized Multimedia Integration Language)을 비롯한 멀티미디어 문서의 저장 및 검색 시스템에 대한 요구가 증가하고 있다. SGML 문서에서는 멀티미디어 데이타는 엔터티로 표현되고 있는데, 이러한 엔터티를 관리하고, 검색하는 연구가 필요할 것이다. 또한, 현 시스템은 단일 문서에 대해 구조적 저장 및 검색을 하고 있어서, 문서간의 연관관계에 대한 정의는 부족하다. 이러한 문서간의 연관 관계를 저장하고 표현할 수 있는 인터페이스에 대한 연구가 필요하다.

참 고 문 헌

[1] International Organization for Standardization, "Information processing-text and office systems-Standard Generalized Markup Language(SGML)," *ISO/IEC 8879*, 1986.

[2] International Organization for Standardization, "Hypermdeia/Time-based Structuring Language (Hy-Time)," *ISO/IEC 10744*, 1996.

[3] *TEI(Text Encoding Initiative)*, URL: http://www.tei-c.org/.

[4] R. Sacks-Davis, T. Arnold-Moore and J. Zobel, "Database systems for structured documents," *IEICE Trans. on Information and Systems*, pp.1335-1342, 1995.

[5] International Organization for Standardization, "Document Style Semantics and Specification Languages(DSSSL)," *ISO/IEC 10179*, 1996.

[6] G. Salton and M. McGill, *Introduction to Modern Information Retrieval*, McGraw-Hill, Tokyo, 1983.

[7] I. Macleod, "Storage and retrieval of structured documents," *Information Processing and Management*, vol. 26. No. 2. pp. 197-208, 1990.

[8] A. Seungupta and A. Dillon, "Extending SGML to accommodate database functions: A methodological overview," *Journal of the American Society of Information Systems*, pp. 629-637, 1997.

[9] 김규태, 현득창, 이수연, 정광철, "관계형 데이타베이스를 이용한 SGML문서 처리", 정보과학회논문지(C), 제3권 제3호, pp. 238-247, 1997.

[10] G.E. Blake, M.P. Consens, P. Kilpelainen, P.A. Larson, T. Snider and F.W. Tompa, "Text/relational database management systems: Harmonizing SQL and SGML," *Proc. Applications of Databases*, pp. 267-280, 1994.

[11] V. Christophides, S. Abiteboul, S. Cluet and M. Scholl, "From structured documents to novel query facilities," *Special Interest Group on Management of Data(SIGMOD)*, 1994.

[12] K. Aberer, K. Bohm and C. Huser, "The prospects of publishing using advanced database concepts," *Conf. on Electronic Publishing*, 1994.

[13] 김용훈, 이원석, 류은숙, 이규철, 이상기, 김현기, 이혜란, 주종철, "SGML 문서 관리 시스템의 설계 및 구현", 한국문헌정보학회지, 제32권 제2호, pp. 157-177, 1998.

[14] J. Clark, *A Free, Object-oriented Toolkit for SGML Parsing and Entity Management*, URL: http://www.jclark.com/sp.

[15] D. Megginson, *The Simple API for XML*, URL: http://www.megginson.com/SAX/

김 학 균

1997년 연세대학교 컴퓨터과학과 졸업 학사. 2000년 연세대학교 컴퓨터과학과 졸업 석사. 현재 KT 서비스개발연구소 음성언어연구팀 전임연구원. 관심분야는 XML 응용 어플리케이션, 음성인식, 자연어처리

조 성 배

1988년 연세대학교 전산과학과(학사). 1990년 한국과학기술원 전산학과(석사). 1993년 한국과학기술원 전산학과(박사). 1993년 ~ 1995년 일본 ATR 인간정보통신연구소 객원 연구원. 1998년 호주 Univ. of New South Wales 초청연구원. 1995년 ~ 현재 연세대학교 컴퓨터과학과 부교수. 관심분야는 신경망, 패턴인식, 지능정보처리