구조적 문서 데이터베이스를 위한 컴포넌트 기반 문서 모델의 설계

조승기°, 고승규, 최윤철
연세대학교 컴퓨터과학과

Design of a Document Model Based on Components for Structured Document Database

SeungKi Cho°, SeungKyu Ko, YoonChul Choy
Department of Computer Science, Yonsei University

요 약

SGML은 구조적 문서를 나타내기 위한 표준으로 이기종간에 호환이 가능하며 다방면에 유연하게 이용 가능한 점이 장점이다. 전자도서관, 온라인 매뉴얼 등의 분야에서 SGML이 활용되면서, 대용량의 SGML 문서를 처리하고 사용자에게 정보를 제공하는 SGML 데이터베이스 시스템이 필요하게 되었다. 또한 웹 상에서의 문서교환의 표준으로 XML이 대두되면서 다양한 XML 문서 데이터를 저장, 관리할 수 있는 XML 문서관리 시스템도 다수 등장하고 있다. 본 연구에서는 이와 같은 구조적 문서 데이터베이스에 활용될 수 있는 데이터 모델로서, 엘리먼트 구조를 보완하고 문서의 개념을 표현할 수 있는 컴포넌트 기반 모델을 제시하고 기본적인 저장 시스템을 구현하였다. 실험을 통하여 검증한 결과 저장과 검색 면에서 성능 향상을 얻었다.

1. 서 론

SGML (Standard Generalized Markup Language)은 이기종간의 호환이 가능하며 논리적인 구조 정보를 포함한다는 장점 때문에 다양한 분야에서 널리 활용되고 있다. 현재 웹 문서의 주종을 이루고 있는 HTML역시 SGML에 그 뿌리를 두고 있다. 또한 HTML의 제한점을 극복하고 좀더 SGML에 가까운 자유로운 형식을 가지는 XML이 차세대 웹 언어로 제시되면서 SGML의 중요성이 더욱 부각되고 있다.

본 연구에서는 기존의 구조적 문서 데이터베이스 시스템에 새로운 컴포넌트(Component) 개념을 도입함으로써, 문서의 의미를 표현할 수 있도록 하였으며 효율적인 정보 획득이 가능한 문서 모델을 제시하고자 하였다.

2. 컴포넌트 (Component) 개념

2.1 정보검색 분야에서의 문서 컴포넌트 개념

문서를 컴포넌트의 모임으로 보는 접근 방법은 정보검색(Information Retrieval) 분야에서도 연구되어 왔다. 컴포넌트는 하나의 명사에서 문서의 단락까지 다양하게 정의되었는데, 문서의 구절들을 컴포넌트로 보아 전체 문서의 개념을 언어학적으로 보다 정확히 파악할 수 있는 방법을 제시하거나, 문서에 컴포넌트 개념을 도입하여 컴포넌트별 가중치를 이용해 색인과 검색을 향상시키는 연구[2] 등이 이루어져 왔다. 기존의 연구에서 컴포넌트의 정의는 다양하게 이루어져 왔으나 컴포넌트를 사용하는 목적은 대체로 일치한다. 그 목적은 문서를 특정한 단위로 나누어 생각함으로써 문서의 의미를 좀더 쉽게 표현하여 문서의 개념과 주제를 정확히 파악할 수 있고, 정보 획득에 있어서 효율을 높이기 위한 것이다.[3].

2.2 구조문서관리 시스템에서의 컴포넌트 개념

SGML, 또는 XML 등의 구조적 문서는 구조 정보를 가지고 있으며 문서의 각 부분이 나타내는 바가 엘리먼트(Element) 이름으로서 명시되어 독립적으로 구성되어 있으므로 기본적으로 컴포넌트 구조를 가지고 있다고 말할 수 있다. 최근 기업내 문서교환의 방편으로 XML이 부상하면서 다양한 컨텐트를 XML문서 형태로 저장, 관리하여 주는 XML 문서관리 시스템 (XML Document/Content Management System)들이 상용화되고 있는데, 이러한 시스템들 역시 XML 문서의 구조적 특성을 살려 문서의 각 부분을 변경, 저장, 재사용 하는 등의 기능을 제공하고 있다. 물론 이러한 기능 역시 컴포넌트를 지원하는 것이라고 볼 수 있다. 그러나 기존 시스템들의 경우 컴포넌트의 정의는 단순히 문서의 각 엘리먼트, 또는 이미지 등의 단일한 데이터에 국한된 정의에서 머물러 있으며 컴포넌트의 재사용이라는 측면 이외에는 별다른 의미를 갖지 않는다. 엘리먼트 정의만으로는 문서의 개념과 의미, 주제를 얻어내기 힘들다. 예를 들어 '논문지'라는 문서가 여러 개의 논문의 모임으로 이루어진다고 할 때, 엘리먼트 정의에는 단지 논문지, 논문이라는 엘리먼트 이름과 그 하위 구조, 반복 여부만이 나타날 뿐 '논문지가 논문의 모임'이라는 의

미와 개념이 정확하게 나타나 있지는 못하다. 대부분의 기존 SGML 데이터베이스 시스템 역시 XML 문서관리 시스템과 마찬가지로 엘리먼트 단위의 처리만을 기본으로 삼고 있어 위에서 언급한 바와 같이 문서의 개념을 잘 반영하지 못하고 있다. 또한, 저장시 모든 엘리먼트를 동일하게 취급하므로 차후 정보를 검색할 때 문서 구성 요소의 중요도를 파악할 수 없다.

3. 컴포넌트의 정의와 그에 기반한 저장 구조

3.1 정의

본 연구에서는 기존 데이터베이스 시스템의 컴포넌트 개념을 보완하는 컴포넌트 개념을 적용함으로써 발전된 문서 모델을 제안하고 성능 향상을 이루고자 하였다. 이를 위해 문서를 컴포넌트의 모임으로 보고 구조적 문서를 위한 컴포넌트의 개념을 정의하였으며 검증을 위해 저장구조 레벨에서 이를 지원하는 시스템을 구현하였다. 이 모델의 기본 개념은 DTD (Document Type Definition)를 이용하여 정의된 컴포넌트 정보를 DI(Document Instance) 저장에 반영하는 것, 즉 DTD에 의존적인 컴포넌트 모델이다. 정의된 컴포넌트 정보는 기존의 엘리먼트 저장 구조를 포함하는 하나의 레이어로서 작용하며 문서의 개념을 표현하고 검색 등의 정보 획득에 도움을 준다. 본 연구에서 제시하는 컴포넌트의 정의는 다음과 같다.

1. 엘리먼트보다는 상위, 문서보다는 하위 개념
2. 문서의 개념적인 정보를 반영한 구성 요소
3. 사용자가 자주 접근하는 구성 요소
4. 문서내에서의 중요도가 큰 구성 요소
4. 문서내에서 여러 번 반복되는 동일한 구조의 구성 요소

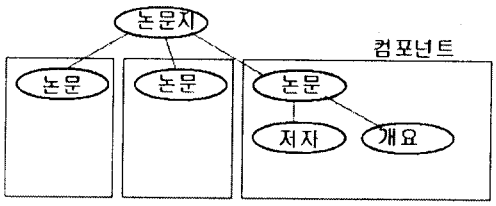

그림 1 간단한 컴포넌트 구조의 예

그림 1은 논문지를 논문의 모임으로 생각한 컴포넌트 구조의 예이다. 즉, 기존의 엘리먼트 저장 구조 위에 컴포넌트 정보라는 한 단계를 두어 문서의 개념적인 정보를 나타내고, 중요한 엘리먼트와 그렇지 못한 엘리먼트를 구별함으로써 정보 획득에 도움을 주게 된다.

기존 트리 모델에서는 각 엘리먼트마다 텍스트를 가지고 있어 검색 또는 브라우징 시 컴포넌트 내 텍스트를 하나로 합치는 과정을 거쳐야 했기 때문에 효율이 떨어지는 단점이 있었다. 본 모델 역시 엘리먼트 단위의 텍스트 저장을 기본으로 하고 있으나 컴포넌트로 정의된 부분은 하나의 텍스트로 저장, 관리하여 접근의 효율성을 높이려 하였다.

3.2 제안된 모델의 이점

3.1의 정의와 같은 컴포넌트를 적용함으로써 얻을 수 있는 이점은 다음과 같다.

1. 각 컴포넌트에 대한 빠른 접근과 검색
2. 동일한 구조의 컴포넌트간에 정보를 공유
3. 문서 처리에 대한 논리적인 단위를 제공하므로 문서 변화 탐지에 이용 가능
5. 문서가 가지는 의미와 개념을 나타내므로 문서 또는 DTD간 유사도 비교의 척도로 이용 가능

3.3 컴포넌트 기반 저장 과정

컴포넌트 저장 과정은 그림 2와 같다. 컴포넌트 추출기(Component Extractor)를 통해 얻어진 컴포넌트 정보가 파서(Parser)를 통해 얻어진 트리 모델(Tree Model)에 추가되어 하나의 문서가 생성되고, 객체지향 데이터베이스에 저장된다. 컴포넌트 추출기는 DTD정보를 이용하여 컴포넌트 정보를 생성해 내는 것을 의미한다. 본 연구에서는 문서의 구조를 잘 아는 전문가가 직접 컴포넌트 정보를 생성하는 것으로 가정하였다. 즉 필요할 때에는 사용자가 컴포넌트를 직접 정의하여 사용할 수도 있는 것이다. 컴포넌트 관리자(Component Manager)는 다양한 문서들이 데이터베이스에 혼재되어 있는 상황에서 각 DTD별, 문서별로 컴포넌트 정보를 관리하고 공유하기 위한 것이다.

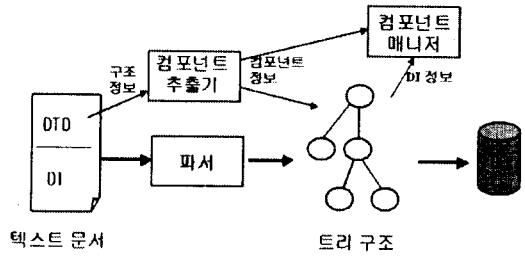

그림 2 컴포넌트 기반 저장 과정

실제로 저장된 문서는 WML (Wireless Markup Language) 문서를 저장한 예인 그림 3과 같다. 하나의 WML 문서는 여러 개의 card의 모임으로 구성되어 있기 때문에, card 엘리먼트를 하나의 컴포넌트로 정의하였다. card 컴포넌트 전체의 내용은 빠른 검색을 위하여 컴포넌트의 루트 엘리먼트 밑에 하나의 텍스트로 관리된다. 컴포넌트를 구성하는 엘리먼트들은 컴포넌트 텍스트 내에서 자신이 차지하는 부분을 나타내는 오프셋을 가진다. 참고로 다음은 WML 문서 DTD의 일부이다.

```
<!-- http://www.wapforum.org/dtd/ -->
<!ELEMENT wml ( head?, template?, card+ )>
<!ELEMENT card (onevent*, timer?, (do | p)*)>
```

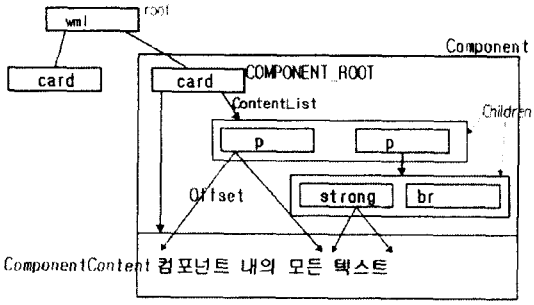

그림 3 저장된 실제 문서의 예

3.4 제안된 저장 모델의 문제점

앞에서 제시한 컴포넌트 모델의 문제점은 첫째, 저장시 오버헤드가 발생한다. 기존의 엘리먼트 구조 위에 컴포넌트 정보를 추가하는 방식이므로 저장 구조의 크기가 커지게 된다.

둘째는 데이터베이스 변경시의 문제점이다. 문서 또는 DTD가 갱신될 때 컴포넌트 정보 역시 변경되어야 하므로 추가적인 작업이 필요하고 따라서 오버헤드가 발생한다. 또한, 검색의 효율성을 위해 한 컴포넌트의 내용 전체를 하나의 텍스트로 관리하므로 내용 변경시 추가적인 작업이 필요하다. 한 엘리먼트의 내용을 변경하면 그 엘리먼트가 속한 컴포넌트 내의 다른 엘리먼트도 영향을 받기 때문이다. 물론 변경의 범위는 컴포넌트 내에 국한되므로 이 경우 정의된 컴포넌트의 크기에 따라 성능의 차이를 보이게 된다.

4. 구현과 실험

4.1 구현

시스템은 객체지향 데이터베이스 시스템인 Object Store를 이용하여 Windows NT 기반으로 구현하였다. 비교 대상이 되는 기존 시스템 역시 Object Store 데이터베이스이며, 문서 모델은 Grove에 기반하여 정의된 DTD 독립적인 모델이다 [5]. 현재 시스템에는 미리 정의된 컴포넌트정보를 반영하여 저장하는 기능과, 텍스트를 컴포넌트 단위로 관리하는 구조, 그리고 컴포넌트 내의 텍스트를 검색하는 기능이 구현되어 있다. 차후 컴포넌트 정보의 관리와 공유를 위해서는 컴포넌트 매니저가 구현되어야 한다.

4.2 실험 방법

실험에 사용된 문서 집합은 Patent(특허) 문서 200개이며, 각 문서에서 특허에 대한 상세한 설명을 포함하여 사용자의 검색 가능성이 높은 SPECIFICATION 엘리먼트를 컴포넌트로 정의하였다. 첫번째 실험은 특허문서 집합 중 100개를 이용해 저장시간, 용량, 검색시간 면에서 기존 저장방식과 새로 구현된 방식을 비교하였으며, 두번째 실험은 나머지 100개 문서를 추가하여 저장하고 첫번째와 동일한 실험을 수행하였다. 그림 4~그림 6은 실험의 결과를 도표로 나타낸 것이다. Component라 표기한 것이 새로 제안된 모델의 실험 결과이다.

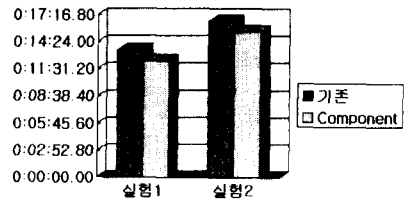

그림 4 저장시간의 비교
(단위: 시:분:초)

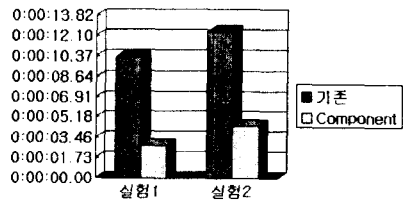

그림 5 검색(Retrieval)시간의 비교
(단위: 시:분:초)

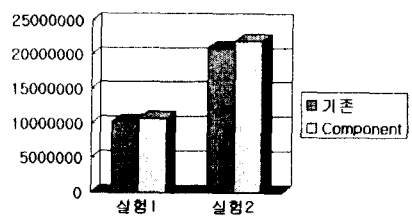

그림 6 데이터베이스의 크기 비교
(단위: Byte)

4.3 실험결과 분석

결과는 예상한 대로 일종의 Time-Space Trade-off로 귀결되었다. 컴포넌트 정보를 활용하여 검색기능은 향상되었으나 컴포넌트 정보가 오버헤드로 작용하여 데이터베이스의 용량이 다소 증가하였다. 그러나 저장구조의 증가율(약 5% 증가)에 비해 높은 검색 효율(기존 검색 속도의 약 36%)을 보였다. 물론 실험 자체가 컴포넌트로 정의된 부분에 대한 검색, 즉 Best-case라고 볼 수 있어, 실제 시스템의 평균적인 성능은 좀 더 낮을 것으로 예상된다. 또한 컴포넌트를 어떻게 정의하는가에 따라 시스템의 성능이 크게 달라질 것이므로 더 많은 실험이 요구된다. 저장시간 면에서는 오버헤드 때문에 시간이 더 길어질 것으로 예상하였으나 오히려 약간 줄어드는 모습을 보여주었다. 이것은 컴포넌트 내의 전체 텍스트를 하나로 저장하였기 때문에 각 엘리먼트의 텍스트 노드에 대한 저장 작업이 줄어들어 저장 효율이 향상되었기 때문이다.

5. 결론 및 향후 연구 과제

본 연구에서는 구조적 문서 데이터베이스 시스템의 성능을 개선하고 기존의 모델보다 발전된 문서 모델을 제안하고자, 문서의 개념을 표현할 수 있는 컴포넌트

모델을 제시하고 검증을 위한 시스템을 구현하였다. 같은 데이터베이스 시스템에 기반한 시스템에서 기존의 저장 방법과 제안된 컴포넌트 기반 저장 방법을 비교한 결과 컴포넌트 개념을 적용했을 경우 저장과 검색시 성능 향상을 얻을 수 있었다. 실험 측면에서는 더욱 다양한 데이터 집합(DTD)에 대한 실험, 동일한 DTD내에서 다른 컴포넌트 구조를 정의하는 실험 등이 필요하다. 컴포넌트 정의 방법과 그 대상이 성능에 많은 영향을 끼치기 때문이다. 정의된 컴포넌트의 특성과 속성을 분석하여 정량적인 평가를 내려야 한다. 또한 시스템의 전체적인 검색 성능을 알기 위해서는 문서내 엘리먼트와 컴포넌트에 대해 다양한 질의를 내리는 실험을 수행해야 한다. 현재 연구에서는 저장과 검색에만 초점을 맞추었으나 이와 같은 컴포넌트 개념은 사용자 인터페이스에 반영되어도 효과가 클 것으로 예상된다. 차후 사용자가 쉽게 컴포넌트를 정의하고 관리할 수 있는 인터페이스 측면에서도 연구가 진행되어야 할 것이다.

6. 참고 문헌

[1] R.Sacks-Davis, T.Arnold-Moore, and J.Zobel, "Database Systems for Structured Documents", November ADT94 (1994)

[2] K.L.Kwok, "Experiments with a Component Theory of Probabilistic Information Retrieval Based on Single terms as Document Components", ACM Trans.Inf.Syst.8,4 (Oct. 1990)

[3] K.L.Kwok, "An Interpretation Of Index Term Weighting Schemes Based On Document Components", Proceedings of 1986 ACM conference on Research and development in information retrieval (1986)

[4] 연제원, 이강찬, 이규철, 나중찬, 이미영, "효율적 XML 문서 변경 및 검색을 위한 페이징 기법" '99 가을 학술발표논문집(I), 한국정보과학회 (1999)

[5] 고승규,조승기,백승욱,이경호,최윤철, "SGML 문서 검색 시스템의 설계 및 구현", '99가을 학술발표논문집(I), 한국정보과학회 (1999)