유사 패턴을 갖는 HTML 문서의 XML 자동 변환

오 금 용[†]·황 인 준[††]

요 약

최근 들어, WWW(World Wide Web)의 급속한 보급으로 많은 양의 정보가 생성되고 있다. 이로 인하여 웹은 이제 정보 교환의 도구로서 뿐 아니라 정보의 저장소로 인식되게 되었다. 현재 웹상의 많은 문서들은 HTML(Hypertext Markup Language)을 사용하여 제작되었다. HTML은 간단하고 배우기가 쉬운 반면, 데이터에 대한 기술을 명확하게 하지 못하는 단점으로 인해 정보 검색에 있어서 효율성을 제공하지 못한다. 이를 보완하기 위한 방법 중에 하나가 구조적인 언어로 부상하고 있는 XML(eXtensible Markup Language) 문서로 변환하는 것이다. XML은 웹 상에서 데이터 교환을 위해 제안된 표준 메타 언어이다. 효과적인 데이터의 교환을 위해, XML은 DTD(Document Type Definition)를 통하여 문서의 구조를 기술할 수 있고 사용자가 원하는 대로 정의할 수 있다. 이러한 구조적 유동성은 웹에서 운용되는 모든 데이터를 통합, 저장, 처리할 수 있는 기반을 제공한다. 본 논문에서는 특히 유사한 패턴을 갖는 HTML 문서의 구조를 분석하고 그에 관련된 경로 정보를 인식하는 방식을 이용하여 XML 문서로의 변환을 자동적으로 수행할 수 있는 XML 변환기를 구현하였다.

Automatically Converting HTML Documents with Similar Pattern into XML Documents

Keumyong Oh[†]·Eenjun Hwang[††]

ABSTRACT

Recently, WWW (World Wide Web) has become a source of a large amount of information, and is now recognized not only as an information-sharing tool, but also as an information repository. Currently, the majority of documents on the web were created using HTML (Hypertext Markup Language). Although HTML is simple and easy to learn, its inherent lack of describing document structure makes it difficult to retrieve information effectively. One possible solution would be to convert such HTML documents into XML (eXtensible Markup Language) documents. XML is a standard markup language for exchanging data on the web. It can describe a document structure freely by defining its own DTD (Document Type Definition). This makes it possible to integrate, store, and retrieve data on the web efficiently. In this paper, we will propose a converter that automatically converts HTML documents with similar pattern into XML documents by analyzing the document structure and recognizing its path information.

키워드 : XML 변환기(XML converter), 유사 패턴(Similar Pattern), 웹 데이터(Web Data), 경로인식(Path Recognition)

1. 서 론

웹은 이미 단순한 정보 교환의 용도를 벗어나 정보 저장소의 기능까지 담당하고 있다. 현재 웹상에 존재하는 많은 문서들은 HTML을 사용하여 제작되었다. HTML은 문서를 보여주기 위한 목적을 갖고 만들어진 언어로 사용이 간편하고 쉬운 반면, 사용할 수 있는 태그가 한정되어 있고 필요한 태그를 임의로 정의할 수 없다는 단점이 있다. 또한 HTML로 제작된 문서는 평면적인 구조를 가지고 있어 사용자는 데이터의 의미적 정보를 직접 지정할 수 없고 데이터의 계층 구조 또한 표현할 수 없다. 그 결과로 웹 문서의 제작에 있어 HTML의 사용은 정보 축적 및 처리에 한계를 초래하였다.

이러한 HTML의 단점을 보안하기 위해 W3C(World Wide Web Consortium)는 1996년 SGML(Standard Generalized Markup Language)을 기반으로 하여 다양한 기능들과 구조적인 표현 능력을 가진 XML을 웹 문서의 표준으로 제안하였다. XML은 SGML의 범용성과 웹 문서의 구조적 표현성을 지원할 뿐 아니라 SGML이 가지지 못한 사용의 편리성을 제공하고 있다. XML의 범용성을 가능하게 한 요인으로는 첫째, 사용자는 문서상에 사용될 태그 세트와 속성을 필요에 따라 자유롭게 정의할 수 있으며 다른 사람들도 그 태그를 사용할 수 있고 둘째, XML 문서에서는 오직

† 정 회 원 : 아주대학교 정보통신전문대학원
†† 종신회원 : 아주대학교 정보통신전문대학원 교수
논문접수 : 2001년 7월 27일, 심사완료 : 2002년 2월 22일

문서의 구조와 의미에 관한 정보만 들어가며, 요소들을 꾸미는 부분은 스타일시트로 분리된다. 즉 XML의 태그는 실제로 문서의 내용을 설명하는 것이기 때문에 기존의 HTML의 용도보다 훨씬 더 광범위한 기능을 가지게 된다. 따라서 XML 문서는 웹 브라우저 상에만 표시하기 위한 것보다는 어떠한 종류의 응용 프로그램과 통합될 수 있는 범용적인 데이터베이스라고 할 수 있는 것이다.

XML이 가진 확장성과 편리함 때문에 많은 웹 문서들이 XML로 작성되고 있다. 하지만 이미 웹상의 많은 문서들이 HTML로 만들어져 있기 때문에 웹 데이터를 통합하고 효율적으로 관리하기 위해서는 HTML 문서를 XML 문서로 변환할 필요가 있다. 이러한 방향의 연구로는 W4F(Wysi-Wyg Web Wrapper Factory)[1], XWRAP(XML-Enabled Wrapper Generation System)[2, 3], XML Wrapper[4-7] 등이 있는데, 이들 연구에서는 HTML 문서의 일부를 자바 객체로 매핑(mapping)하고 단순한 정보 추출에 그치고 있다. 본 논문에서는 이러한 문서의 변환을 위해 공통적인 패턴을 갖는 HTML 문서의 구조를 분석하고 그에 관련된 경로 정보를 인식하는 방식으로 자동적인 XML 문서로의 변환을 가능하게 하는 자동 XML 문서 변환기를 구현한다.

본 논문의 구성은 다음과 같다. 2장에서는 HTML 문서를 XML 문서로 변환하는 것에 대한 관련 연구를 살펴보도록 하겠다. 3장에서는 구조화된 문서간의 유사성을 정의하는 과정을 설명하고 이를 기반으로 4장에서는 변환기가 HTML 문서를 XML 문서로 변환하는 과정에 대해 자세히 기술한다. 5장에서는 HTML 문서의 XML 변환 과정에 대한 예제를 보이며, 마지막으로 결론 및 향후 연구 방향에 대해 언급하겠다.

2. 관련 연구

W4F는 특정 웹에 있는 자원을 쉽고 간편하게 래퍼(wrapper)로 만들 수 있게 해준다. W4F는 웹 문서를 식별하기 위한 검색 언어(retrieval language)와 추출 규약을 표현하는 선언적 추출 언어(declarative extraction language) 그리고 추출된 정보와 사용자가 정의한 구조체를 연결시키기 위한 인터페이스로 구성된다. 검색 층(retrieval layer)은 문서에 get, post 등의 방식으로 접근을 제어하고 재전송, 접속 실패, 인증 등을 다룬다. 추출층(extraction layer)은 HTML 문서상의 일반적인 문법을 모두 다루는 기능을 하고 여기서 사용되는 추출 언어는 웹 문서의 구조를 충분히 표현할 수 있는 언어로 구성되어 있다. 이를 바탕으로 지정된 소스로부터 정보를 추출하고 이 정보는 문자열로 저장된다. 이때 추출 규칙은 비교적 자유로이 규정될 수 있다. 매핑 층(mapping layer)에서는 사용자가 원하는 대상 구조를 기반으로 추출된 정보를 매핑하게 된다. 각각의 층들은 서로 독립적인 구조를 가지고 있기 때문에 독립적인 컴포넌트 요소로서 재사용할 수 있다. W4F는 반자동으로 웹 문서의 데이터를 자바 클래스로 생성하는 것이다. 사용자는 이 클래스 파일을 변형하여 다른 어플리케이션에 재사용할 수 있다. 그렇기 때문에 웹 문서에 대한 접근이 필요한 데이터베이스 시스템이나 소프트웨어 에이전트(software agent)의 부분으로 사용되기도 한다. 하지만 W4F는 HTML 문서를 파싱해서 그 태그를 자바 객체로 매핑시키는 정도에만 그치고 있어서 사용자가 필요한 용도에 따라 가공해야 한다는 단점이 있다.

XWRAP은 웹 문서를 위한 래퍼 프로그램의 반자동적인 생성을 위한 시스템이다. 이 시스템의 목적은 비 구조적인 HTML 문서를 구조적인 XML 문서로 변환하는 것이다. 또한 XML로 제작되지 않은 웹 문서를 XML 문서로 표현하는데 있어 불필요한 부분에 대한 제거를 통한 내용 선택 기능을 사용자에게 제공한다. 시스템은 내부적으로 두 단계의 코드 생성 구조를 가지고 있다. 첫 단계에서는 개발자가 정의한 정보 추출 규칙에 따라 메타 정보를 변환하고 다음 단계에서는 생성된 추출 규칙을 XWRAP 컴포넌트 라이브러리와 결합하여 주어진 웹 소스에서 실행 가능한 래퍼 프로그램을 생성한다. 자바로 구현된 이 프로그램의 장점은 사용자 인터페이스가 제공되고 래퍼의 생성이 용이하다는 것이다. 그러나 XML이 아닌 문서에서 구조를 분석하는 데 미흡한 면이 있고, 사용자가 개입하여 변환을 할 경우 다수의 문서를 위한 공통적인 DTD의 생성이 어려운 단점이 있다.

Taniar와 Jiang[8]은 효율적인 웹 정보 검색을 목적으로 평면적인 구조의 HTML 문서로부터 구조적인 XML 문서를 생성하는 시스템을 제안하였다. 이들이 제안한 방법은 웹 페이지의 분석을 통해서 구조 정보를 얻어내고 그 구조 정보를 이용해 스키마(schema)를 유도한 후 매핑 도구를 사용하여 XML 문서를 얻어내는 것이다. 하지만 변환하고자 하는 HTML 문서마다 사용자가 개입해야 하는 단점이 있다. Huck[9]은 특정한 프로그램에서 필요한 정보를 웹 문서로부터 추출하기 위한 다른 방법으로 여러개의 독립적인 웹 문서로부터 정보를 다룰 수 있는 Jedi(Java based Extraction and Dissemination of Information)를 제안했다. Jedi는 래핑 층(wrapping layer)과 중재 층(mediation layer)으로 구성되어 있는데 래핑 층은 이질적인 문서 모델을 규칙적인 객체 모델로 변경시킨다. 중재 층에서는 여러 개의 문서를 통합하는 뷰(view)를 정의할 수 있고 사용자는 이 층을 통해 데이터를 다룰 수 있다. 이질적인 문서 형태를 규칙적인 객체 모델로 변형시키기 때문에, 특정한 문서에만 존재하는 정보를 추출하고 싶을 때는 문서 모델을 고려하여 추출 규칙을 정해야 하는 단점을 지닌다.

위에서 설명한 시스템들은 비구조적인 문서 전체나 일부

를 XML 문서로 변환하는 방법에 대하여 제시하였다. 변환 과정이나 방법은 상이할 수 있으나 변환의 기본 골격은 비구조적인 문서의 구조를 분석하고 분석한 구조 중에 데이터를 중심으로 XML 문서로의 매핑을 사용자 개입을 통해서 이루게 하는 공통적인 방법을 취하고 있다. 즉 비구조적인 정보에서 구조적인 정보를 생성할 때 사용되어야 할 기본 과정이자 필수적인 단계가 되는 것이다.

본 논문에서 유사 패턴을 지니고 있는 다량의 웹 문서를 XML 문서로 변환하는데 있어 일단 변환된 문서와 유사한 구조를 지닌 문서에 대해서는 구조 분석 단계를 생략하고 변환된 문서의 경로 정보를 이용하여 사용자 개입 없이 자동적인 변환을 수행할 수 있는 자동 변환기를 제안하고자 한다. 변환기에서는 구조 분석을 위해 문서를 구조화할 때 사용하는 문서 객체 모델인 DOM(Document Object Model)[10]을 사용한다. 이것은 HTML 문서나 XML 문서와 같은 인터넷 문서에 대하여 문서의 내용 및 구조를 객체로 표현하고 그 객체를 핸들링 할 수 있고 동적으로 문서의 내용과 구조, 스타일을 바꿀 수 있게 하는 플랫폼에 독립적이고 언어 중립적인 인터페이스를 의미한다. 현재 DOM 레벨 2는 HTML 4.0과 XML 1.0을 지원하고 있으며 본 변환기에서 사용된 DOM 기술은 레벨 2 수준이고 이를 이용하여 HTML 문서의 구조 분석과 XML 문서의 조작에 사용하고 있다.

3. 문서의 유사성 정의

3.1 구조화된 HTML 문서

변환기에서 문서의 구조적 정보를 얻어내기 위한 기술 중 하나로 DOM을 이미 언급했다. 문서가 적격(Well-formed)[11]일 경우 DOM을 이용하면 문서를 트리 형태로 표현할 수 있는데, 이를 기반으로 데이터를 삽입하고 추출할 수 있다. 적격 문서는 다음과 같이 정의될 수 있다.

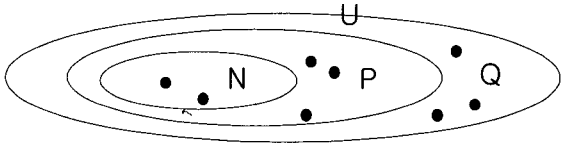
(i) 전체적으로 문서 양식을 갖추어야 한다.
(ii) 관련된 명세에 기술된 모든 적격성(well-formedness) 규약을 만족한다.

문서의 양식을 갖추어야 한다는 말은 다음을 의미한다.
(i) 하나 또는 그 이상의 엘리먼트를 포함하고 있다.
(ii) 루트 또는 문서 엘리먼트라 불리는 엘리먼트가 정확히 하나 존재하며, 그 엘리먼트의 어느 부분도 다른 엘리먼트의 내용에 포함되어서는 안 된다. 그 밖의 모든 엘리먼트들의 경우, 시작 태그가 다른 엘리먼트의 내용 안에 있을 경우 종료 태그 역시 같은 엘리먼트의 내용 안에 있다. 간단히 말하면, 시작 태그와 종료 태그에 의해 범위가 한정되는 엘리먼트들은 각각의 내부에서 올바르게 중첩된다.

이와 같은 항목들을 잘 구성된 HTML 문서로서 적용하기 위해서 Vodink[11]은 다음을 정의했다.

- 모든 엘리먼트들은 시작 태그와 종료 태그를 가져야 한다.
- 모든 태그는 올바르게 중첩되어야 한다.
- 모든 속성값들은 따옴표로 묶여 있어야 한다.
- 모든 엘리먼트들이 빈 태그로 사용될 때는 "/>"로 끝나야 한다.

HTML 문서에 대해 위에서 언급한 유효성이 검증되면, 파싱 과정을 통해 트리 구조로 표현할 수 있다. 트리 구조로 표현된 HTML 문서 안에는 태그와 텍스트뿐 아니라 다른 웹 문서로의 링크나 테이블 그리고 이미지나 오디오, 비디오 파일과 같은 멀티미디어 데이터가 있을 수 있으며 이를 구조화된 HTML 문서(structured HTML document)라 부른다. 본 논문에서는 구조화된 HTML 문서에 대해 공통된 패턴을 고려하여 문서간의 유사도(degree of similarity)를 측정하고 이를 바탕으로 유사한 문서들의 집합을 정의한다.
구조화된 HTML 문서가 갖는 태그들은 구조를 이루는 중요한 요소들로 이루어져야 한다. HTML에 사용되는 모든 태그가 구조에 영향을 주는 것은 아니므로 우선 구조 분석 전에 불필요한 태그들은 제거되어야 한다. 이를 위하여 전체 태그 중에서 문서의 물리적 구조에 영향을 줄 수 없는 태그와 물리적 구조에는 영향을 줄 수 있으나 의미적 구조에는 영향을 줄 수 없는 태그, 그리고 물리적 구조와 의미적 구조에 모두 영향을 줄 수 있는 태그들로 분류한다. 첫 번째 분류에 속하는 태그의 예로서는
과 <HR> 등이 있다. 두 번째 분류에는 <FONT>나 <B>, <IT> 등의 태그들이 포함될 수 있는 데 이들은 데이터의 앞뒤에서 시작 태그와 종료 태그로 사용되지만 문서의 의미적 구조보다는 글자체에 관여하는 경우가 많다. 즉 물리적 구조에는 영향을 주지만 의미적 구조에는 전혀 영향을 주지 않는다. 마지



PUQ = U, N ⊂P, N⊄Q

U : HTML 문서에서 사용될 수 있는 모든 엘리먼트들의 집합
Q : 단일태그로 사용되는 엘리먼트들의 집합
P : 시작태그와 종료태그를 사용하나 의미적 구조에 영향을 주지 않는 엘리먼트들의 집합
N : 시작태그와 종료태그를 사용하고 의미적 구조에 영향을 주는 엘리먼트들의 집합

(그림 1) 엘리먼트의 분류

막 분류에는 <P>나 <H1>, <H2> 등의 태그와 테이블에 관계되는 <TD> 등이 될 수 있다. (그림 1)은 HTML 문서에서 사용되는 태그들에 대한 이러한 분류를 보여준다.

3.2 공통 패턴을 갖는 문서 집합

본 논문에서는 구조화된 문서를 기반으로 다른 문서와의 비교를 통해 서로의 유사성을 측정하고 어느 범위내의 유사성을 가진 문서에 대해서는 공통 패턴을 갖는 문서들의 집합이라 정의한다. 구조화된 HTML 문서는 트리 형태로 표현되기 때문에 두 문서간의 유사성은 트리 형태의 유사성으로 대치될 수 있으므로 문서의 유사성은 다음과 같이 정의될 수 있다.

노드들의 집합(set of nodes) V와 간선들의 집합(set of edges) E로 이루어진 그래프 $G=(V,E)$가 있다고 가정하자. $u,v\in V$가 하나의 간선에 의해 연결되어 있다면 u와 v는 서로 인접해 있다(adjacent)라고 하고 $u\sim v$로 표기한다. 정수 $i=1,\cdots,n$에 대하여 서로 다른 순차적 노드들의 연속인 $u_0u_1\cdots u_{i-1}u_i$를 경로(path)라 하며 이때 I는 경로의 길이가 된다. 특히 $u_0=u_i$일 때를 순회(cycle)라고 한다. 순회가 없는 그래프를 트리(tree)라고 하고 루트를 갖는 트리를 루트 트리(rooted tree)라 하며 루트로부터 u까지 연결된 경로의 길이가 그 노드의 레벨(level)이 되고 $lev(u)$라 표기한다. 두 노드의 관계가 $u\sim v$이고 $lev(v)-lev(u)=1$이라면 노드 u는 v의 부모(parent)가 되고, 반대의 경우 v는 u의 자식(child)이 된다.

$T_1=(V_1,E_1)$과 $T_2=(V_2,E_2)$을 루트 트리라고 하자. 이때 $H_1\subseteq V_1$, $H_2\subseteq V_2$를 만족하는 전단사함수 $\xi: H_1\rightarrow H_2$가 존재하고 이때 노드들이 이웃하고 계층적 관계가 유지된다면 이를 부트리 동형사상(subtree isomorphism)이라고 한다. 물론 $T_1[H_1]$과 $T_2[H_2]$ 역시 트리가 된다. H_1의 부집합인 H_1'에 대하여 $\xi': H_1'\rightarrow H_2'$인 전단사 함수가 있고 더 이상의 동형 사상이 없는 경우를 극대(maximal) 부트리 동형 사상이라고 한다. 바로 극대 부트리 동형 사상 문제(maximal subtree isomorphism)는 두개의 루트 트리에서 극대의 부트리 동형 사상을 찾아내는 것이다. 다음의 정의는 경로선(path-string)에 관한 것이다.

(정의 1) 루트 트리 T의 서로 다른 두 노드 u와 v가 있고, 라고 하자. u에서부터 v까지의 단일 경로가 있을 때, 두 노드간의 경로선은 모든 $i=1,\cdots,n$에 대하여 $s_i=lev(x_i)-lev(x_{i-1})$가 $\{-1,1\}$의 값을 갖는 선 $s_1s_2\cdots s_n$이며 $str(u,v)$으로 표시된다. 이때 루트 트리의 루트로부터 파생되는 단일경로에 대해 경로선의 시작점이 루트이고, $lev(x_i)$ 대해 $lev(x_{i+1})$이 존재하지 않을 때의 경로를 $\overline{P_i}$라고 한다.

두개의 루트 트리 $T_1=(V_1,E_1)$과 $T_2=(V_2,E_2)$에 대해 각각의 트리상에서 $V=V_1\times V_2$인 연관 그래프(association graph) $G=(V,E)$를 정의할 수 있다. 이때 V의 두 노드쌍 (u,w)와 (v,z)에 대하여 $(u,w)\sim(v,z)\Leftrightarrow str(u,v)=str(w,z)$인 관계가 성립할 수 있는데 두개의 노드쌍 (u,w)와 (v,z)가 G에서 이웃하다면 T_1에서의 u와 v와의 관계가 T_2에서 w와 z의 관계와 같을 때를 만족하게 된다. 물론 반대의 경우에도 성립하게 된다. G의 부집합 중 각각의 노드들이 상호 이웃한 경우를 *클리크(clique)*라 하는데, G에서 최대 클리크(maximum clique)는 두개의 루트 트리 사이에 크기가 최대인 클리크를 의미한다. 최대 클리크 문제(maximum clique problem)는 최대 클리크를 찾는 것을 말하며, Marcello[12]는 극대 부트리 동형 사상(maximal subtree isomorphism)과 최대 클리크 (maximal clique)의 관계를 다음과 같이 정리하였다.

(정리 1) 두 루트트리와 관계되는 연관 그래프에서 어떤 극대 부트리 동형사상은 최대 클리크를 유도한다.

두 트리 사이의 최대 클리크를 통해 유사도를 측정하기 위한 식을 소개한다. 차수가 n인 임의의 그래프 $G=(V,E)$가 있다고 가정하자. 그리고 S_n가 $\mathcal{R}^n$의 표준집합으로서 표시되고 σ가 1과 같은 요소를 갖는 벡터라면, $S_n=\{x\in\mathcal{R}^n: \sigma' x=1 \text{ and } x_i\geq 0, i=0,i,\cdots,n\}$로 정의 할 수 있다. G의 꼭지점들의 부집합 C가 주어졌다면, 점 S_n내의 x_c은 다음과 같은 지표 벡터(characteristic vector)로서 정의될 수 있다.

$$x_i^c=\begin{cases}1/|C|, & \text{if } i\in C\\ 0, & otherwise\end{cases} \tag{1}$$

이때 $|C|$는 C의 기수(cardinality)를 의미한다. 다음에서 소개하는 이차함수를 고려하자.

$$f(x)=x'Ax+\frac{1}{2}x'x \tag{2}$$

여기서 $A=(a_{ij})$는 $n\times n$을 갖고 대칭형을 이루며 다음과 같이 정의되는 요소로 이루어진 G의 인접 행렬(adjacency matrix)을 말한다.

$$a_{ij}=\begin{cases}1, & \text{if } v_i\sim v_j\\ 0, & otherwise\end{cases} \tag{3}$$

(2)를 고려하여, $x\in S_n$에 대하여 $f(x^*)\geq f(x)$를 만족하면, 한 점 $x^*\in S_n$을 S_n안에 있는 f의 전역 극대점(global maximizer)이라 한다. 임의의 $\varepsilon>0$이 존재하여 $x\in S_n$에 대하여 $f(x^*)\geq f(x)$를 만족하며 x^*가 ε보다 작은 값을 갖는다면, 지역 극대점(local maximizer)라 한다. 최대 부트리 동형 사

상 문제를 풀기 위해서, $T_1=(V_1,E_1)$와 $T_2=(V_2,E_2)$를 두 개의 루트 트리라고 하고, A를 연관 그래프 G에 관계되는 n-노드 인접 행렬이라면, Marcello[12]는 Hofbauer[13]에 의해 제안된 반복 방정식(replicator equation)을 사용하여 이 문제를 풀었다. I_N이 $N\times N$ 단위 행렬이고 인접 행렬 A와 더불어 다음과 같이 표기될 수 있다.

$$W=A+\frac{1}{2}I_N \tag{4}$$

반복 방정식은 반복적으로 수행되어 나가며 S_n상에서 정의되는 함수 f를 극대화하게 된다. 이 과정을 통해 결국 연관 그래프에 있는 최대 클리크의 지표 벡터와 관계되는 지역 극대점으로 수렴하게 된다. 이 순서대로라면, T_1과 T_2 사이의 극대 부트리 동형사상을 유도하게 된다. 여기서 공통 패턴을 갖는 문서들의 집합을 정의하는 데 사용될 유사도 값을 얻을 수가 있다. 두 구조간의 유사도를 측정하는 것을 두 문서간의 유사구조를 측정하는 수단으로 사용될 수 있고, 그에 관계되는 경계를 정하는데 이용된다. 공통 패턴을 갖는 문서 집합은 다음과 같이 정의 될 수 있다.

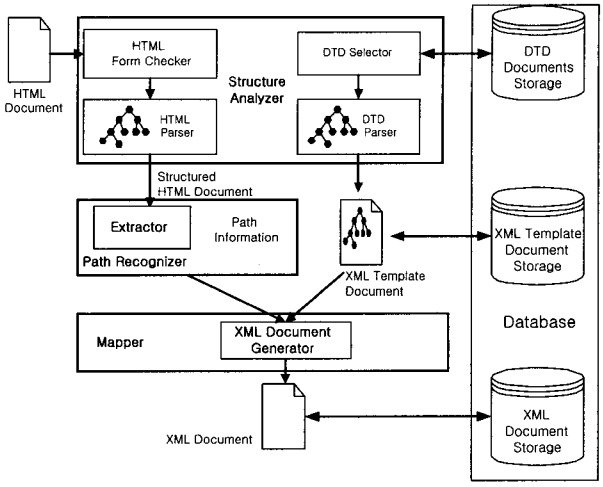
(정의 2) $T(d_i)=(V_i,E_i)$를 d_i의 트리 구조라 하고 $Sim(d_i,d_j)$를 d_i와 d_j의 유사도라 하자. 임의의 경계 $\phi>0$에 대하여 $D=\{d_i \mid \text{for } i=1,\cdots,n\}$는 다음을 만족해야 한다.

(i) d_i는 같은 DTD를 사용한다.

(ii) 임의의 정수 i,j에 관하여 $1\geq Max(Sim(d_i,d_j))\geq\phi$이다.

4. 변환기의 구조 및 변환 과정

일반적으로 비구조적 형상에서 구조적 형상으로 변환하는 데 있어 필수적인 과정은 분석과 추출이다. 본 변환기에서도 역시 이러한 과정을 통해 분석 과정에서 생성되는 정보를 최대한 활용하고 사용자의 개입을 최소화 하는데 중점을 두어 설계되었다. (그림 2)는 변환기의 전체적인 구조를 보여준다. 그림에서 구조 분석기는 HTML 문서의 구조 분석을 수행하며 구체적으로 다시 입력되는 HTML 문서에 대한 구조 분석을 하는 모듈과 문서가 사용할 DTD를 선택해서 엘리먼트들 간의 상호 구조 관계를 분석하는 모듈로 이루어져 있다. HTML 문서에 대한 구조 분석은 문서 내에 데이터에 대한 위치 파악을 수월하게 하며 데이터가 있는 경로를 중심으로 데이터에 의미를 부여한다[14, 15]. 또한 DTD 분석을 통해 데이터가 매핑되어 들어갈 실제적 영역을 지정할 수 있다. 여기서 말하는 실제적 영역이란 XML 문서를 이루는 기본 골격을 의미하며 DTD에서 정의한 엘리먼트들의 조합을 통해 구체화된다.



(그림 2) 변환기의 전체 구조

공통 패턴을 가진 문서들의 변환 과정은 크게 사용자 개입에 의한 초기 문서 변환 과정과 초기 문서 변환에서 나온 정보를 이용하여 유사한 다른 문서들을 자동으로 변환하는 과정으로 나뉜다.

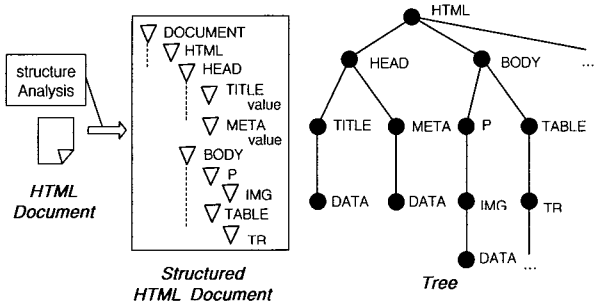
4.1 유사한 임의의 문서에 대한 초기 변환

공통 패턴을 가진 유사한 문서들의 집합 D가 있다고 가정하자. 집합내의 문서를 자동 변환하기 위해 첫 문서의 변환은 다음과 같은 네 단계를 통해서 이루어진다 : (i) 입력되는 HTML 문서의 분석과, (ii) 입력 문서와 관계가 있는 DTD 문서의 분석, (iii) 데이터 추출과 더불어 이루어지는 경로 정보 추출, 그리고 (iv) XML 문서의 생성 등이다. 여기서 경로 정보의 추출은 사용자의 개입을 통해서 이루어지며 집합내의 다른 문서들에 대한 자동적인 변환에 사용된다. 다음은 집합내의 임의의 문서 d_i에 대한 변환 과정을 설명한다.

HTML 문서의 분석 : HTML 문서의 분석 과정에서는 우선 문서 d_1을 파싱 과정을 거친 후 루트를 갖는 트리 $T(d_1)=(V_1,E_1)$로 나타낸다. $T(d_1)$의 구조에서 각각의 태그는 계층적으로 표현되며 트리 구조의 말단에는 추출될 실제적 데이터들이 위치하고 있다. 이때 위치하고 있는 데이터들과 데이터들에 관계되는 경로 $\overline{P_i}$는 분석 과정을 마친 후 경로 인식기로 전달이 된다.

(그림 3)은 이러한 트리 생성의 예를 보여준다. 트리에 정의되는 엘리먼트로는 루트 노드와 내부 노드, 그리고 말단 노드가 있고 각 노드는 해석된 HTML 태그로 표현된다. 루트 노드는 HTML 문서를 대표하며, 내부 노드는 HEAD, BODY를 나타내는 노드에서부터 HTML의 가장 작은 단위의 태그를 나타내는 노드들로 구성되며 각 노드들은 태그가 가지는 이름, 속성, 값의 정보를 포함하게 된다. 말단 노

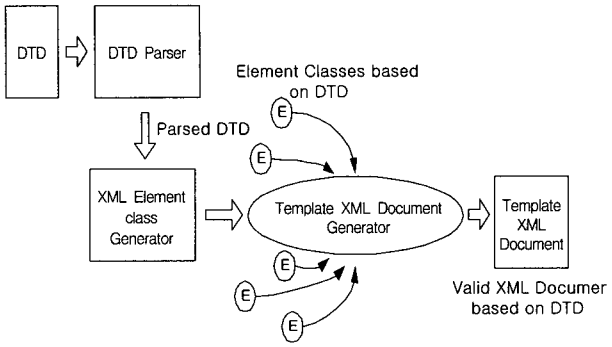
드에는 추출될 데이터가 저장된다. 태그에서 사용되는 URI (Uniform Resource Indicator)나 URL(Uniform Resource Locator)은 태그를 나타내는 노드의 자식 노드로 표현된다.



(그림 3) 구조화된 HTML 문서

DTD 문서의 분석 : 사용자는 $T(d_1)$에 관계되는 DTD를 선택할 수 있으며 변환기는 사용자의 편의성을 위해 그래픽 인터페이스를 제공한다. 선택된 목록을 바탕으로 구조 분석기는 저장소에서 해당 DTD를 선택하여 파싱하고, DTD에 정의되어 있는 엘리먼트들을 추출한다. 추출된 엘리먼트들은 DTD에 명시된 그들 사이의 관계를 고려하여 다시 조립된다. 단, 이때 데이터는 존재하지 않고 단지 DTD내에 정의된 엘리먼트들만 고려되며 이 과정을 통해 해당 XML 문서의 골격을 만들게 된다. 우리는 이것을 템플릿 XML 문서(template XML document)라 부른다. 템플릿 XML 문서는 집합 D에 존재하는 다른 문서의 변환에 재사용되며 이를 위해 변환기는 템플릿 XML 문서를 별도로 저장한다. (그림 4)는 템플릿 XML 문서를 생성하는 과정을 보여주고 있다.

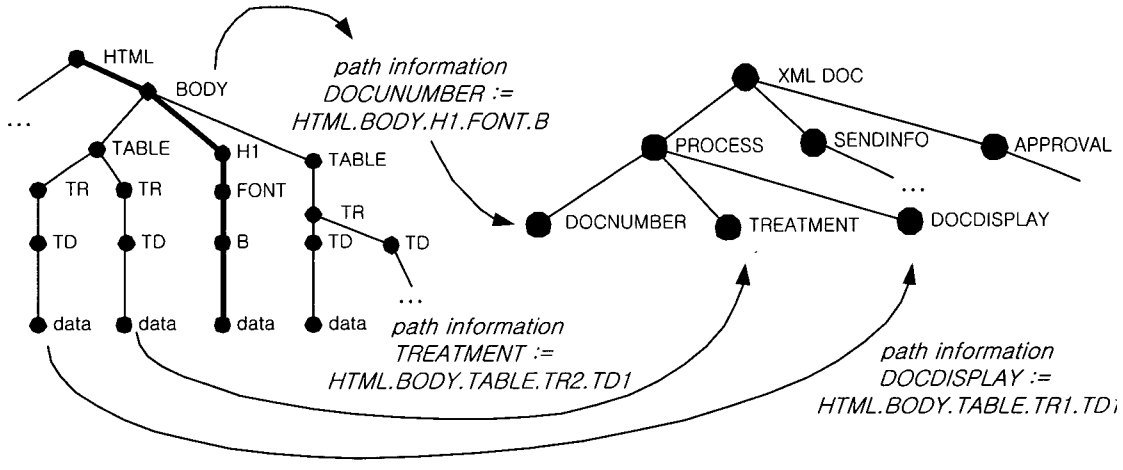
템플릿 XML 문서를 만드는 과정에서 엘리먼트들은 정확히 한번씩 사용된다. 그러나 많은 XML 문서에서 동일한 엘리먼트가 여러 번 나타날 수 있다. 예를 들어, 엘리먼트



(그림 4) DTD를 이용한 템플릿 XML 문서의 생성

를 정의할 때 출현 빈도를 나타내기 위해 "*"나 "+", "?" 등의 기호를 사용한다. "*"의 경우 엘리먼트가 문서 내에 없어도 되거나 여러번 나타나도 됨을 의미하며, "+"의 경우는 한번 혹은 여러 번의 출현을 의미한다. "?"기호는 한번 혹은 나타나지 않음을 의미한다. 만약 동일한 엘리먼트가 문서 내에 여러 번 나와야 할 경우, 템플릿 XML 문서에 반복되는 엘리먼트를 삽입하는 추가적인 과정이 필요하게 된다. 이 과정은 데이터 삽입 과정 전에 이미 사용된 동일 엘리먼트가 데이터를 가지고 있는 경우를 확인하여 데이터가 이미 존재할 시에 엘리먼트를 생성하고 데이터를 삽입함으로써 해결할 수 있다.

경로정보 저장과 데이터 추출 : 이 단계에서는 데이터를 추출하고 그에 관한 경로 정보를 저장하며, 데이터가 위치한 경로와 템플릿 XML 문서 안의 해당 엘리먼트와의 관계를 결정하여 기록한다. 이때 사용자는 정확한 관계 설정을 위해 제공되는 인터페이스를 따라 데이터의 내용과 해당 엘리먼트의 연결에 참여할 수 있다. 이러한 과정은 각각의 데이터들에 대해 반복적으로 수행된다. 모든 데이터에 대한 경로 정보의 저장이 끝나면, 다음 문서의 변환 과정에 재사



(그림 5) 데이터 경로와 템플릿 XML 문서사이의 관계

용되어 나타나는 경로는 저장된 경로 정보와의 비교를 통해 해당 엘리먼트로 치환된다. 이때 더 이상의 사용자 개입이 없이 자동적인 변환을 시도할 수 있다. (그림 5)는 데이터의 경로와 템플릿 XML 문서에 있는 엘리먼트와의 관계를 생성하는 과정을 보여주고 있다.

예를 들어, 공공기관에서 사용하는 기안문에 해당하는 HTML 문서를 구조화하였고, 각각의 기안문에서 발신자의 데이터가 다음과 같은 경로에 위치한다고 가정하자.

html.body.table[5].tr[1].td[1].p[1]

변환기는 사용자에게 데이터가 위치한 경로와 데이터 내용을 알려주게 되고, 사용자는 데이터 내용을 고려하여 인터페이스에서 관련된 엘리먼트인 <발신인>을 선택하면 템플릿 XML 문서에서 <발신인>의 실질적 노드를 찾아 데이터를 삽입하게 된다. 이때 경로 인식기는 위의 경로를 대신하여 엘리먼트 <발신인>을 기억하게 되며 이러한 관계는 앞으로 입력될 HTML 문서의 변환에 반복적으로 사용된다.

XML 문서의 생성 : 추출기는 트리의 말단에서 추출한 데이터를 매퍼에게 전달하며 매퍼는 템플릿 XML 문서에 각각의 데이터를 삽입하는 과정을 수행하게 된다. 모든 데이터의 삽입 과정이 끝나면 변환된 새로운 XML 문서를 만들게 된다.

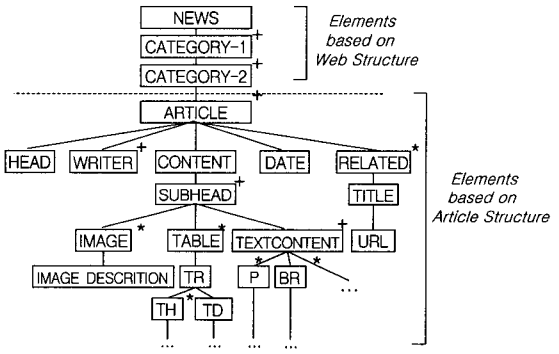
4.2 집합내의 다른 문서에 대한 반복적인 변환

위에서 문서의 초기 변환 과정을 설명하였다. 그 결과로 XML 문서가 만들어졌고, 더불어 사용할 경로 정보를 생성되었다. 생성된 경로 정보를 이용하면 사용자 개입을 최소화하거나 배제할 수 있고 DTD 분석 과정도 생략할 수 있다. 변환에 있어 집합내의 문서가 서로 동일한 구조를 가졌을 경우 변환에 문제가 없이 진행되지만, 동일하지 않고 유사한 구조를 지닌 문서라면 변환에 있어서 약간의 수정 과정이 필요하게 된다. 여기서 변환할 문서는 공통 패턴을 지니고 있는 문서들 중에서도 문서의 데이터 유실이 없는 경우를 전제로 설명하고자 한다. 전체적으로 반복적인 변환에서 진행되는 과정은 HTML 문서의 분석, 데이터 추출 그리고 마지막으로 XML 문서의 생성으로 구성되어 있다. HTML 문서의 분석 과정은 입력되는 문서는 메모리에 트리 형태로서 올려지며 시작된다. 이러한 작업은 경로 정보를 기반으로 데이터 추출을 위한 과정이다. 변환기는 초기 변환 후 저장되어 있는 경로 정보를 불러와 경로 정보에 저장된 규칙에 따라 데이터를 추출하게 된다. 추출된 데이터가 삽입될 해당 템플릿 XML 문서의 엘리먼트를 찾는 과정을 반복하며 모든 데이터의 삽입이 완료될 때까지 삽입 과정을 반복하게 된다. 삽입이 완료되면 XML 문서가 생성되게 된다.

5. 예제 : 뉴스 기사 문서의 변환

이 장에서는 웹상에서 일반적으로 접할 수 있는 뉴스 사이트에 대한 XML 변환 과정을 설명하고자 한다. 최근 들어 뉴스 기사를 위한 XML 어플리케이션들이 생겨나고 있으나 기존에 제작되었던 다수의 문서는 HTML을 이용하여 제작되었고, 이를 XML 문서로 변환하기 위해서 변환하고자 하는 문서의 분석 과정이 필요하다[16].

일반적으로 뉴스 정보 제공은 시간에 따라 분류가 되고, 고정된 날짜의 뉴스 페이지는 여러 개의 하위 목록으로 연결되어 있다. 하위 목록들은 또 다른 하위 페이지로 반복적으로 연결되어 있다. 뉴스 사이트를 보면 정치, 경제, 스포츠 등의 여러 가지 목록으로 나누어져 있으며, 스포츠는 다시 야구, 축구, 농구 등의 여러 목록으로 연결되어 있다. 이러한 중간 페이지들은 어떠한 기준에 적합하게 반복적으로 연결되어 있으며 그 끝은 특정한 사건 중심의 HTML 문서임을 알 수가 있다[17]. 관리자는 변환에 사용할 DTD를 디자인해야 하는 데, 위의 과정을 고려하여 설계할 수 있으며 (그림 6)은 그러한 예를 보이고 있다.

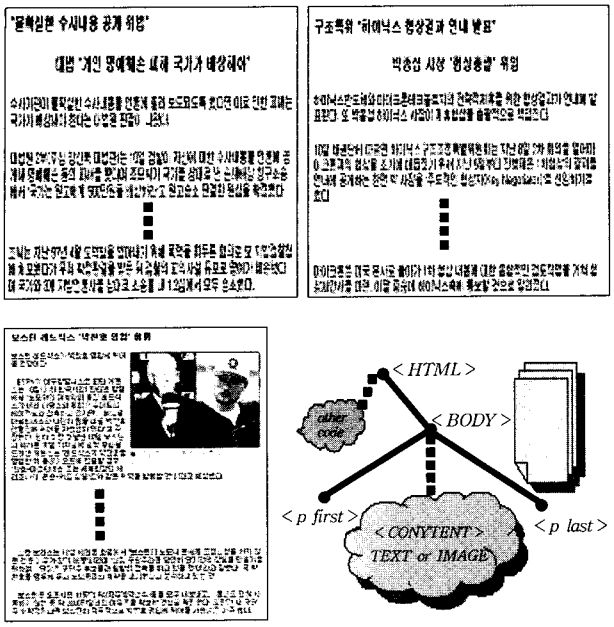


(그림 6) DTD의 예와 관계되는 템플릿 XML 문서의 예제

(그림 7)과 (그림 8)은 현재 서비스 중인 한 웹사이트의 문서의 모습들이다. 이곳에 있는 다량의 기사 문서들은 이와 같은 형태로 저장되어 있으며 기사는 제목과 기사 내용 그리고 리포터로 구성되는 공통 패턴을 가지고 있다.

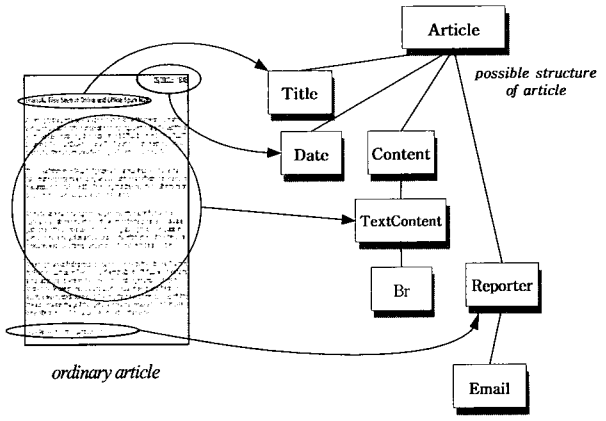
변환이 시작되면 구조 분석기는 입력된 문서가 잘 구성된 문서인지 점검한 후 파싱 과정을 통해 구조화된 문서로 변환을 한다. DTD 선택기는 사용자와의 상호작용을 통해 해당 문서에 관계되는 DTD를 DTD 저장소에서 찾아 파싱하고 분석하여 템플릿 XML 문서를 만들게 된다. (그림 8)의 우측 그림은 좌측 DTD와 관련된 템플릿 XML 문서의 한 예를 보여주고 있다.

추출기는 구조화된 HTML 문서의 말단 노드에서 실제 데이터를 추출하고 해당 엘리먼트에 삽입하고 경로 인식기는 변환 중 발생되는 경로정보를 저장하게 된다. 모든 데이터들에 대해 추출 및 매핑 과정이 끝나면, 해당 XML 문서가 생성되게 된다.



(그림 7) 일정 패턴의 뉴스 사이트 웹 문서들

초기 변환 후 반복적 변환은 각각의 문서가 지니고 있는 공통적인 구조를 기반으로 이루어진다. 이러한 문서들은 서로 동일하거나 혹은 유사한 구조를 지니는 데, 구현된 변환기에서는 파일로 저장된 경로 정보 문서를 기초로 하여 데이터의 위치를 기록하고 각 문서의 데이터 추출 시에 사용하게 된다. 반복적 변환 후에 완성된 문서들은 (그림 9)와 같은 형태의 XML 문서로서 생성이 된다.



(그림 8) 문서의 데이터와 해당 DTD의 매핑

```
< ?xml version = "1.0"? >
< article >
< title > Dramatic Rise Seen in Online and Offline Spam Mail
</title >
< content >
According to the postal service administration at the Ministry of
Information and Communication, total number of mass mailings
sent during the last three years has been increasing dramatically
from 1.17500 billion in 1998 ; 1.31562 billion in 1999 ;
< br/ >
...
< br/ >
</content >
< reporter >
Park Min-seon
< email > sunrise@chosun.com </email >
</reporter >
</article >
```

(그림 9) 생성된 XML 문서의 예

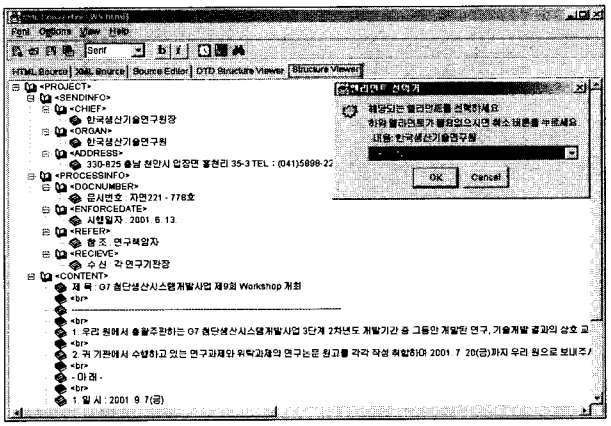
6. 변환기의 구현

6.1 구현 환경

본 변환기는 Windows 95,98/NT 환경에서 개발되었다. 자바(Java) 언어로 개발되었기 때문에 이식성과 플랫폼 독립성을 제공하며, 자바 기술을 위해서 썬 마이크로 시스템의 JDK(Java Development Kit)1.3을 기반으로 하였다. 주요 패키지로는 사용자 인터페이스 제작을 위해 썬 마이크로 시스템의 스윙(Swing)을 사용하였고, 파싱과 DOM 사용을 위해 jaxp1.1을 사용하였다. 또한 오라클(Oracle) 데이터베이스와의 연동은 JDBC 커넥션 풀을 이용하였다.

6.2 세부 기능 모듈

변환기는 문서 편집을 위한 모듈, HTML 문서와 DTD의 구조를 보기 위한 뷰어(viewer) 그리고 데이터베이스와의 연동을 위한 모듈로 구성되어 있다.



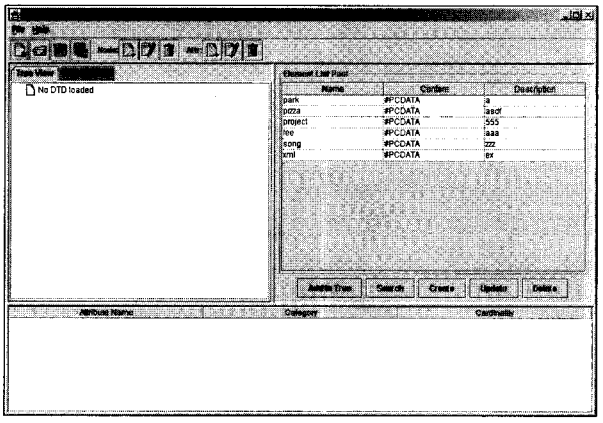
(그림 10) 자동 변환기 화면

- HTML 문서 편집기 : 입력되는 문서에 대하여 변환 전에 수정 작업이 필요하다면 문서에 대한 편집이 가능하다. XML 편집기와 유사한 구조를 지니고 있고 수정과 저장 그리고 새로운 문서의 생성 등이 가능하다.
- XML 문서 편집기 : 변환 결과로 생성된 XML 문서에 대해서 데이터 조작 등과 같은 수동적인 작업이 필요한 경우 사용자에게 인터페이스를 저장하고 HTML

문서 편집기와 유사한 기능을 제공한다.

- DTD 뷰어 : DTD내의 엘리먼트들간의 관계를 트리 구조를 통해 시각화하고 DTD 제작에 경험이 부족한 사용자라도 구조를 쉽게 알 수 있게 한다.
- HTML 문서 구조 뷰어 : 구조화된 HTML 문서의 데이터 위치를 쉽게 파악할 수 있도록 하며 변환될 데이터의 경로를 사용자에게 보여준다.
- 데이터베이스와의 연동 : DTD 선택이나 템플릿 XML 문서의 선택 그리고 XML 문서의 저장을 위해서는 데이터베이스와의 연계가 필요하며 이를 위해서 데이터베이스에 종속적인 모듈을 제공한다.

(그림 10)은 구현된 변환기의 화면을 보여주며 (그림 11)은 HTML 문서를 XML 문서로 변환하기 전에 XML 문서에서 사용될 엘리먼트, 에트리뷰트 등을 정의하는 DTD 문서를 수정, 삭제, 생성하는 DTD 에디터의 모습이다.



(그림 11) DTD 데이터베이스를 통한 DTD 선택 화면 모습

7. 결론 및 향후 연구 방향

본 논문에서는 HTML 문서를 구조 분석을 통해 XML 문서로 자동으로 변환하는 변환기의 개발에 대해 설명하였다. 구체적으로, 자동적인 변환을 수행하기 위해 (i) 유사한 구조를 지닌 문서들의 집합을 정의하고, (ii) XML 문서에 사용될 HTML 문서와 DTD를 분석하며, (iii) 구조화된 HTML 문서의 말단 노드에 위치하고 있는 실질적 데이터를 추출하고, (iv) 추출된 데이터에 관계되는 경로 정보를 작성하며, (v) 유사한 문서의 변환을 위해 경로 정보를 수정한다. 이러한 방법을 응용하면 다른 정의된 집합 내의 유사성을 지닌 문서들의 변환에 적용할 수 있다. 향후 연구 과제로는 저장되어 있는 XML 문서를 질의 시스템과 연계하고, XSLT(eXtensible Stylesheet Language Transformations) 등을 적용하여 사용자의 취향에 맞는 맞춤형 정보 제공을 자동적으로 이루어지게 하는 시스템을 개발하는 것이다.

참 고 문 헌

[1] A. Sahuget and F. Azavant, "WysiWyg Web Wrapper Factory (W4F)," Available at http : //db.cis.upenn.edu/~sahuguet/WAPI/wapi/.

[2] L. Liu, C. Pu and W. Han, "XWRAP : An XML-enabled Wrapper Construction System for Web Information Sources," *Proc. of International Conference on Data Engineering*, pp.611-621, 1998.

[3] L. Liu, W. Han, D. Buttler and C. Pu, "An XML-Enabled Wrapper Construction System for Web Information Sources," *SIGMOD Conference*, pp.540-543, 1999.

[4] F. Harary, "*Graph Theory*," Addison-Wesley, Reading, MA, 1969.

[5] J. R. Gruser, L. Raschid, M. E. Vidal and L. Bright, "Wrapper Generation for Web Accessible Data Sources," *Proc. of International Conference of Cooperative Information Systems*, pp.14-23, 1998.

[6] N. Kushmerick, D. Weil and R. Doorenbos, "Wrapper induction for information extraction," *Proc. of Conference on Artificial Intelligence*, pp.729-737, 1997.

[7] N. Ashish and C. A. Knoblock, "Wrapper Generation for Semi-structured Internet Sources," *Proceeding of the Workshop on Management of Semi-structured Data*, pp.8-15, Tucson, Arizona, 1997.

[8] Taniar, Y. Jiang, J. W. Rahayu and L. Bishay, "Structured Web Pages Management for Efficient Data Retrieval," *Proc. of International Conference on Web Information Systems Engineering*, 2000.

[9] G. Huck, P. Fankhauser, K. Aberer and E. Neuhold, "Jedi : Extracting and Synthesizing Information from the Web," *Proceeding of CoopIS*, pp.32-43, 1998.

[10] "Document Object Model (DOM) Level 2 HTML Specification Version 1.0, W3C Working Draft, WD-DOM-Level-2-HTML-20011210," Johnny Stenback, et al.

[11] "HTML 4.01 Specification, W3C Recommendation 24-December-1999, REC-html401-19991224," Dave Raggett, et al, December, 1999.

[12] M. Pelillo, K. Siddiqi and S. W. Zucker, "Matching Hierarchical Structures Using Association Graphs," *Proc. of European Conference on Computer Vision*, pp.3-16, Feb., 1998.

[13] J. Hofbauer and K. Sigmund, "*The Theory of Evolution and Dynamical Systems*," Cambridge University Press, Cambridge, UK, 1988.

[14] Vodnik, "*HTML complete*," Course Technology, Reading, 1999.

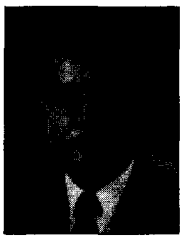
[15] Taniar, Y. Jiang, J. W. Rahayu and L. Bishay, "Structured Web Pages Management for Efficient Data Retrieval," *Proc. of International Conference on Web Information Systems Engineering*, 2000.

[16] S. Lee, E. Hwang and K. Byeon, "Template-based XML Data Integration System," *Proc. of International Conference on E-Commerce*, pp.1-7, Seoul, Korea, 2000.

[17] K. Oh, D. Park and E. Hwang, "Automatic Conversion of Web Pages with Common Pattern," *Proc. of International Conference on Internet Computing*, pp.213-218, Las Vergas, 2001.

오 금 용

e-mail : keumyong.oh@samsung.com
2000년 아주대학교 정보통신대학 (학사)
2002년 아주대학교 정보통신전문대학원 (석사)
2002년~현재 삼성전자 연구원
관심분야 : 멀티미디어 데이터베이스, 이질 데이터베이스 통합, XML



황 인 준

e-mail : ehwang@madang.ajou.ac.kr
1988년 서울대학교 컴퓨터공학과 (학사)
1990년 서울대학교 컴퓨터공학과 (석사)
1998년 Univ. of Maryland at College Park 전산학과 (박사)
1998년~1999년 Bowie State Univ., Assistant Professor
1999년~1999년 Hughes Research Lab. 연구교수
1999년~현재 아주대학교 정보통신전문대학원 조교수
관심분야 : 데이터베이스, 멀티미디어 시스템, 정보 통합, 전자상거래, XML