웹 정보의 추출 및 통합을 위한 래퍼 시스템
(A Wrapper System for Extraction and Integration of Web Information)

정 재 목 † 김 형 주 ††
(Jae Mok Jeong) (Hyoung-Joo Kim)

요 약 이 논문은 웹 정보를 추출하기 위한 래퍼 프로그램을 생성해내기 위한 XWS(XWEET Web-wrapper System)의 데이타 모델과 소프트웨어 개발방법에 대해 설명하고 있다. 다양한 정보 출처에 존재하는 정보에 접근하기 위해서는 원본 데이타를 공통된 데이타 모델로 변환하고 통합해야 된다. XWS 시스템은 XWEET 프로젝트의 부분으로 개발되었다. 우리는 효율적이고 사용하기 쉬운 Perl 프로그램 언어를 사용하여 XWS 시스템을 구현하였다. XWS은 다른 시스템과 구별되는 몇 가지 특징을 가지고 있다. 첫째, HTML 페이지로부터 정보를 추출하기 위해 사용되는 데이타모델과 연산자들은 HTML 문서의 다양한 뷰를 지원할 수 있는 통합된 모델을 사용한다. 둘째, XWS는 사용자가 래퍼 프로그램을 손쉽게 생성해 내기 위한 그래픽 인터페이스 프로그램을 제공한다. 셋째, 객체지향적으로 설계된 고수준의 스크립트 언어를 사용하였다. 또한 논문에서 DBLP 사이트로부터 검색된 논문 정보를 추출하기 위한 자세한 예제를 통해 XWS의 사용법을 보이고 있다.
키워드 : XWEET, Web, HTML, WWW, XML

Abstract This paper describes the data model and software development of XWS, an XWEET Web-wrapper System for generation wrapper program. To access information from various information sources, one has to convert and integrate source data into the same data model. XWS is developed as a part of XWEET project. We have implemented the XWS system using the Perl programming language stressing efficiency and ease-of-use. XWS has a few distinct features. First, data model and operator used for extracting information from HTML support a unified model of different views of HTML document. Second, it provides a user-friendly interface program to enable wrapper programmer to generate wrapper easily. Third, XWS use the high-level script language designed by object-oriented methodology. In this paper, we also present the detail demonstration where it is useful for extracting article information from DBLP site.
Key words : XWEET, Web, HTML, WWW, XML

1. 서 론

웹(WWW)은 근래에 들어 정보를 저장하고 공유하기 위한 수단으로 많이 사용되고 있다. 상품 카탈로그, 예약, 전자 상거래, 날씨 예보, 컨퍼런스 알림 등의 정보가 웹을 통해 전달되고 있다. HTML에 기반한 웹 정보들은 사람이 읽고 살펴보기 위한 목적으로 디자인되고 출판되었으므로 프로그램이 그 의미를 이해하기가 어렵다. 또한 웹 페이지의 내용은 중복되거나 그래픽 이미지 내에 정보가 숨겨져 있는 경우가 많다. 또한 웹 정보 소스들은 각각의 웹 페이지 관리자에 의해 운영되므로, 알맞은 포맷의 데이타를 요구하기 힘들게 된다. 최근 들어 XML[1]이 정보를 공유하기 위한 표준으로 이용되고 있다. XML은 태그 기반의 접근 방법을 사용하는 신축적이고 확장 가능한 언어이다. XML의 기본적인 아이디어는 데이타 엘리멘트의 태그가 데이타의 의미를 나타내도록 하는 것이다. 아직은 HTML과 XML이 웹 환경에서 혼재 되어 사용되고 있다. XML은 정보를 설명하기 위한 목적으로 사용되고, HTML은 화면에다가 정보를 표현하기 위한 목적으로 사용된다. XSL[2]은 XML을 HTML로 변환하는데 필요한 스타일을 정의하고, 이를

· 본 논문 결과는 서울대 Bk-21 정보기술 사업단과 정보통신부 ITRC (e-Biz 기술 연구센타)의 지원으로 이루어졌다.
† 비 회 원 : 서울대학교 컴퓨터공학부
jmjeong@oopsla.snu.ac.kr
†† 종신회원 : 서울대학교 전기컴퓨터공학부 교수
hjk@oopsla.snu.ac.kr
논문접수 : 2000년 3월 27일
심사완료 : 2003년 6월 4일

바탕으로 HTML을 만들어 내게 한다. 그러나 HTML 데이타를 XML로 변환하는 방법에 대해서는 아직 표준적인 방법이 존재하지 않는다.

많은 웹 응용 프로그램은 웹 데이타 소스로부터 좀 더 구조적인 형태로 변환된 결과를 입력으로 받아들이고 있다. 웹 데이타를 읽어 들일 필요가 있는 응용 프로그램은 웹으로부터 구조적인 데이타를 추출하기 위해 디자인된 래퍼(wrapper)라는 소프트웨어를 사용하고 있다. 이러한 래퍼 소프트웨어를 직접 만드는 작업은 시간이 많이 들고, 만들어진 래퍼가 잘못 동작할 우려가 있다. 이 논문에서, 우리는 웹 데이타 소스를 추출하기 위한 강력하고 신축적인 방법을 제공하는 XWS(XWEET Web-Wrapper System)을 설명한다. XWS 시스템의 목적은 HTML 페이지를 웹 응용 프로그램이 이용하기 쉬운 XML로 변환하는 일이다. XWS는 XWEET(**X**ML DBMS for **W**Eb Environmen**T**)[3,4] 시스템의 일부로 개발되었다. XWS는 Perl을 이용하여 구현되었다. 그리고 XWS는 다른 시스템과 구별되는 몇 가지 특징이 있다. 첫째, HTML로부터 정보를 추출하기 위한 데이타 모델과 연산자들은, HTML 문서의 다양한 뷰를 지원할 수 있는 통합적 모델을 지원한다. 둘째, 래퍼 디자이너가 래퍼를 손쉽게 생성해내기 위한 인터페이스 프로그램을 지원한다. 셋째, 객체지향적으로 설계된 고수준의 스크립트 언어를 사용하였다.

3 장에서는 XWS 시스템에 대한 구조에 대해 설명하고, 4 장에서는 HTML 페이지를 추출하고 통합하기 위해 사용된 연산자들의 데이타모델에 대해 언급한다. 5 장에서는 XWS 시스템의 자세한 예를 보여주고, 2 장에서는 관련 연구에 대한 설명과 XWS의 구별되는 특징에 대해서 언급한다. 그리고 6 장에서 결론을 내린다.

2. 관련 연구

웹 상에서 정보를 추출하거나 통합하는 일은 최근 들어 활발히 연구되고 있다. 웹 페이지에서 특정한 구조적 정보(튜플들의 집합, 또는 객체들의 집합)를 추출하여 다른 형태의 데이타모델로 변환하는 작업을 포함하고 있다. 웹 페이지에서 정보를 추출하는 일은 논문에서 언급한 래퍼라는 프로그램이 담당하고, 변환된 데이타의 통합은 미디에이터(mediator)라는 시스템에 의해 처리된다[5].

웹 래퍼 시스템을 구축하는 데 어려운 점은 HTML 페이지들이 프로그램에 의해 정보를 추출하는 목적보다는 사람에게 정보를 보이기 위한 목적으로 설계되었다는 점이다. 그러므로 정보가 자연언어로 기술된 문장 속이나 그래픽으로 표현된 그림 내에 숨겨져 있는 경우가 많다. 더욱이 정보의 소스가 되는 HTML 페이지가 자주 바뀌는 경우 래퍼도 변경하여야 하기 때문에 래퍼를 관리하는 게 힘들어지게 된다. 래퍼의 빠른 생성을 돕기 위해 여러 시스템들이 제안되었다. 래퍼 생성 도구는 데이타가 웹 페이지에 어떻게 포함되어 있고 어떤 식으로 정보를 추출해 내는지 기술하는 문법(Grammar)기반의 방법[6-9]과 자동으로 귀납적인 학습 추론을 통해서 정보를 추출하는 방법[10,11]으로 크게 분류할 수 있다. 후자는 시스템에 정보가 저장되어 있는 HTML 페이지에 특정 라벨을 붙여서 입력하거나 대화식으로 정보를 입력하면 시스템 엔진이 특정한 룰의 문법을 생성해 내는 방법이다.

W4F[9,12]는 웹 데이타 소스를 처리하는 래퍼를 생성시키기 위한 자바 툴 킷이다. HTML 페이지로부터 정보를 추출해 내는 구조적인 기술 언어를 가지고 있으며, 래퍼 생성을 도와줄 수 있는 비쥬얼 툴을 지원한다. 그리고 래퍼의 모든 기술이 선언적이다. Jedi[8]도 역시 자바 기반의 정보 추출 및 통합을 담당하는 시스템이다. 래퍼에 어튜리뷰트 문법을 사용하여 소스로 입력되는 데이타에 장애가 발생한 경우에도 처리될 수 있도록 장애-용인(fault-tolerant)적인 파싱 문법을 제공하고 있다. HTML로부터 정보를 추출하는 언어는 패턴 매칭 방법과 문법에 의한 파싱 두 가지 기법을 모두 사용하고 있다. 그리고 래퍼의 출력 결과를 처리하는 미디에이터는 간단한 객체 모델을 사용하고 있다. WebL[13]은 자바로 구현된 웹 문서 처리를 위한 독립적인 스크립트 언어이다. HTTP나 FTP와 같은 공통된 프로토콜에 대한 지원과 HTML이나 XML과 같은 데이타 타입을 지원하고 있다. WebL은 특정 웹 페이지가 접근 가능하지 않은 경우 여러가지 예외 사항을 처리할 수 있게 하여 웹 데이타를 좀 더 믿을 수 있게 하는 서비스 결합자(service combinator)와, 웹 페이지를 처리하여 특정 정보를 추출할 수 있게 하는 마크업 대수(markup algebra)를 가지고 있다. TSIMMIS[6] 프로젝트는 이질적인 정보 소스를 빠르게 통합하기 위한 툴을 개발하기 위한 것이다. TSIMMIS에서 쓰이는 래퍼는 질의어를 하위 소스에서 이해할 수 있는 하나 또는 그 이상의 명령어나 질의어로 변환시키고 결과를 응용 프로그램에서 이해할 수 있는 포맷으로 변환하는 일을 한다. 각 데이타베이스나 웹 페이지를 위한 래퍼가 직접 코딩(hard-coded)되었으며 툴 킷에서는 통신, 질의어 변환 등 공통적으로 사용하는 부분을 라이브러리화하여 제공하고 있다. XWRAP[14]은 반자동화된 래퍼 프로그램 생성 시스템이다. 사용자와의 대화식 작업을 통하여 자바로 이루어진 래퍼 프로그램을 생성해 내는 일을 하고 있다.

XWS에서는 HTML로부터 정보를 추출하기 위한 데이타모델과 연산자가 HTML 문서의 다양한 뷰를 지원

할 수 있는 통합적 모델을 지원한다. 그리고 래퍼 디자이너가 웹 래퍼를 손쉽게 생성해 내기 위한 인터페이스 프로그램을 지원하고, 객체지향적으로 설계된 고수준의 스크립트 언어를 사용한다는 점에서 관련 연구와 차별성을 가진다. 또한 타입에 상대적으로 자유로운 Perl 언어를 이용하여 구현되고 스크립트 파일에 기술된 처리 규칙들이 XWS 라이브러리와 함께 Perl 엔진에 의해 수행된다.

3. XWS 시스템 구조

XWS는 XWEET[3] 시스템의 **데이타 소스**에서 사용되는 웹 래퍼 프로그램이다. XWEET 시스템은 이질적인 데이타의 통합을 위해 설계된 미들-웨어 시스템이다. XWEET에서 XWS는 외부 웹 데이타 페이지를 처리하기 위한 입력소스 중의 하나로서 동작한다.

전체적인 XWEET 시스템의 구조가 그림 1에 나타나 있다. **데이타 소스**는 외부 데이타 소스로부터 정보를 추출하기 위한 래퍼와 XML 문서와 XWEET 시스템의 XML 저장소인 PDM으로 구성되어 있다. 데이타 소스는 XWEET의 입력 출처로 사용되고 이 모듈의 출력은 XDM(XWEET Data Model)이 된다. **XDM**은 XWEET 시스템의 데이타 모델이고, PDM[15](Persistent Data Manager)는 XML 데이타를 RDBMS로 저장하거나 인출하는 인터페이스를 제공하고 있다. 또한 PDM은 XWEET의 메타 데이타를 처리하는 인터페이스를 제공하고 있다. PDM에서는 데이타의 빠른 입출력을 위하여 인덱스를 제공하고 있다. 저장 엔진으로서 RDBMS를 사용함으로써 RDBMS 기술의 안정성을 보장받을 수 있다. **XSI**(XWEET *Semantic Integrator*)는 다른 데이타 인코딩을 통합시키는 일을 한다. **XQP**(XWEET Query Processor)는 데이타 소스로부터 온 XML 데이타에 대한 선언적인 질의를 처리한다.

XWS는 XWEET 시스템의 데이타 소스에서 웹 데이타 소스를 변환하기 위한 래퍼로 사용된다. XWS는 HTML 페이지로부터 정보를 추출하여 그것을 XML로

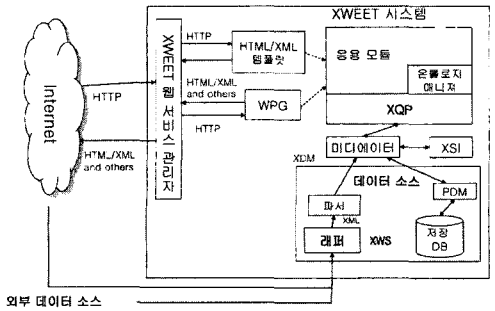

그림 1 XWEET 전체 구조

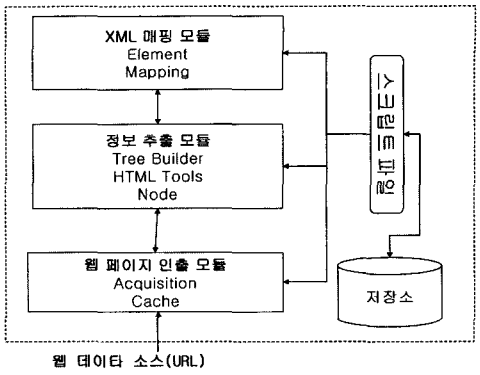

그림 2 XWS 시스템 구조

변환하는 일을 담당한다. XWS는 Perl[16]을 이용하여 구현되었으며, 래퍼를 생성해 내기 위해 유용한 라이브러리 함수와 그래픽 인터페이스 프로그램을 제공하고 있다. XWS는 웹 페이지 인출 모듈, 정보 추출 모듈, XML 매핑 모듈로 이루어져있다. 그림 2에 XWS의 전체 구조도가 나타나 있다. HTML 페이지는 인출되어 XWS에서 XDM으로 변환된다.

웹 페이지 인출(Web Page Retrieval) 모듈은 정보 추출 처리를 위한 환경을 설정하는 첫 번째 XWS의 부분이다. 먼저 사용자로부터 입력 받거나 선택된 URL을 받아들인다. URL, 동적 웹 페이지를 생성해 내기 위한 변수 인자들, 메쏘드 타입(예를 들어 GET, POST) 등이 또한 주어진다. 현재 버전의 XWS에서는 쿠키 정보를 처리하지 않고 있다. 이 정보를 바탕으로 웹 페이지 인출 모듈은 웹 페이지를 가져오기 위한 규칙을 만들고, 로컬 캐쉬에 유효한 사본이 존재하는지를 조사한다. 뉴스 신문이나, 일기 예보 같은 웹 페이지들은 특정한 시간에 갱신이 된다. 네트워크 트래픽과 추출 시간을 줄이기 위해 가능하다면 유효한 로컬 사본이 사용된다. 만약 사본이 이용 가능하지 않다면, 웹 페이지 인출 모듈은 HTTP 요청을 보내서 해당되는 웹 페이지를 가져온다. 로컬 디스크에 저장되는 캐쉬 공간의 크기는 XWS 시스템의 전체 설정파일 maxcachesize라는 설정인자를 통하여 조절할 수 있다. 캐쉬의 크기가 너무 커지는 경우에는 캐쉬 관리자가 오래된 캐쉬 파일을 지우고 새로운 내용으로 갱신하여 일정 한도 이상으로 커지지 않도록 관리한다. 마지막으로, 이 웹 페이지는 파싱 되어 불필요하거나 에러가 있는 태그들이 처리가 된다. <TR>, </TR>과 같이 쌍으로 쓰여야 할 태그들이나
과 같은 단일 태그들은 분류되어 처리된다. 여기에서 불필요한 태그란 내부적인 XWS의 데이타 모델에 부합되지 않는 태그를 일컫는다. 그리고 클라이언트측 웹 브라우저에서 화면 입출력이나 이미지 처리를 위한 목적으로

사용되는 자바스크립트 정보는 무시된다. XWS의 웹 페이지 인출 모듈의 일부인 파싱 모듈이 헤더에 나와야 되는 태그(예. title, meta 등), 바디에 나와야 하는 태그(예. h1, a, ol 등), 다른 엘리멘트 내에 직접적으로 포함이 되어야 하는 태그(예. samp, cite 등)들로 구별하여 파싱을 하게 된다. 오류가 있는 태그인 경우 휴리스틱 알고리즘에 의해 수정을 한다. 그 이후에 인출된 웹 페이지를 파싱 트리 형태로 변환한다.

정보 추출(Information Extraction) 모듈은 래퍼 디자이너가 관심이 있는 웹 페이지로부터 정보를 어떻게 뽑을지를 기술해 놓은 추출 규칙을 적용하는 두 번째 모듈이다. 하위 모듈에서 넘겨진 내부 파싱 트리는 텍스트 오프셋이나, 엘리멘트 어트리뷰트 등의 부가적인 정보를 추가한 HTML 노드 트리로 변환된다. 텍스트 검색에 있어서 힘든 일 중에 하나는 웹 페이지의 다양한 뷰에 대한 단일적인 모델을 제공하는 것이다[8]. XWS에서는 웹 페이지에 대한 두 가지 뷰를 제공한다. 하나는 태그를 가진 페이지를 텍스트 문자열의 순차파일로 보는 뷰이다. 다른 하나는 이 텍스트 문자열을 방향성 그래프 표현으로 보는 뷰이다. 영역 추출(Region Extraction)은 순차파일 뷰에서 웹 문서로부터 관심 있는 영역을 구분해 내는 모듈이다. 계층 & 정규식 추출(Hierarchy & Regular Expression Extraction)은 HTML 노드 트리의 논리적인 패스(path)와 노드 위치를 구분해 내는 모듈이다. 래퍼 디자이너는 웹 문서에 대한 이 두 가지 뷰를 상호 변화하여 처리할 수 있다. XWS 스크립트 언어에 사용되는 데이타 모델에 대한 규약은 4 장에 설명되어 있다.

XML 매핑(XML Mapping)모듈은 XML DTD 템플릿과 두 번째 단계에서 추출된 정보를 바탕으로 XML 문서를 생성해내는 세 번째 모듈이다. 두 번째 단계에서 추출된 정보는 스트링의 복합 리스트 구조 형태로 이 모듈로 전달된다.

스크립트 파일은 XWS를 사용하는 래퍼 디자이너가 XWS의 각 단계를 조절하는 처리규칙을 지정하기 위해 사용된다. 각 단계별로 모듈에서 어떤 일을 하는지 명시하여 URL로 접근 가능한 웹 페이지로부터 XML 문서를 만들어낸다. 이 처리 규칙은 강력하고 자유로운 Perl의 이점을 살릴 수 있는 인터페이스를 제공하는 스크립트 언어에 의해 처리가 된다. Perl이 준 대화형 언어이기 때문에 스크립트 언어는 컴파일 작업이 필요 없이 Perl과 XWS 라이브러리 모듈 하에서 수행이 된다. XWS을 이용하여 만들어진 프로그램 모듈은 XWEET 시스템이나 자동 웹 페이지 생성 시스템 등에서 입력 모듈로 사용되거나 독립된 프로그램 모듈로서 사용 가능하다. 또한 XWS에서 자주 사용되는 설정 파일은 지정된 웹 사이트로부터 자동으로 다운로드 받아서 수행될 수 있는 기능을 가지고 있다.

4. 데이타모델

이장에서는 HTML 정보의 XWS 웹 페이지 데이타 모델에 대해 설명한다. 웹 문서에 대한 개념적인 모델은 HTML 문서로부터 관심 있는 정보를 추출하는 간단하고도 확실한 방법을 제공하고 있다. 지금까지 제안된 대부분의 웹 래퍼 생성기들은 웹 페이지에 대한 하나의 뷰를 가지고 있다. 예를 들어 W4F[9]에서는 웹 문서를 순서화된 그래프 뷰로 보고 있고, [17]에서는 글자들의 일련적인 텍스트 흐름으로 보고 있다. 글자 패턴을 찾고, 웹 문서로부터 광고와 같은 특정부분을 효율적으로 분리해 내기 위해서는 웹 문서에 대한 다양한 뷰를 제공하고 그 뷰에 대한 연산을 제공하는 것이 효율적이라는 것이 관찰되었다. XWS에서는 웹 문서를 텍스트 스트링 뷰와 계층 구조의 뷰로 보기 위한 데이타모델과 연산자들을 제안하였다. 웹 문서를 텍스트 스트링 뷰로 보고 처리하는 것은, 웹 래퍼 디자이너로 하여금 불필요한 헤더와 유일한 구분자에 의해 구별되는 부분을 손쉽게 떼어내는데 유용하다. 이 유일한 구분자는 XWS에서 정규식으로 판별이 된다. 순서화된 그래프 뷰에서는 웹 문서에 대한 계층 구조로부터 정보를 추출해 내는데 유용하다.

웹 문서에 대한 두 가지 뷰에 대한 추상화는 다음과 같다.

정의 1 (텍스트 스트링 뷰) 텍스트 스트링 뷰 $T_1(a_1, a_2, \cdots, a_n)$를 $a_i \in N$ (N이 캐릭터들에 대한 집합이고 $1 \le i \le n$)인 웹 페이지에 대한 뷰라고 정의하자. n은 스트링의 길이라고 부른다.

정의 2 (순서화된 그래프 뷰) 순서화된 그래프 뷰는 HTML 문서를 적절한 도메인(어트리뷰트 이름, 어트리뷰트 값, 캐릭터 내용)에 대한 라벨 순서 그래프로 보는 뷰로 정의한다. 모든 순서 그래프의 집합 T_2는 독립적인 값을 갖는 $d[t_i, \cdots, t_n]$($d \in D$는 라벨, $t_i \in T_2$.$[t_1, \cdots, t_n]$은 순서화된 그래프 리스트)로 이루어진다.

웹 문서는 XWS 스크립트 연산자를 통해 접근되고 처리가 된다. 여기서는 간단히 XWS 스크립트 연산자에 대해 설명하겠다.

텍스트 스트링 뷰에 대한 기본적인 단위는 **영역**이다. 영역은 시작점과 끝 점으로 구별되는 웹 문서의 연속적인 텍스트 부분이다. **영역집합**은 영역들의 묶음으로 정의가 된다. 영역집합에서 영역들은 중첩되거나 겹칠 수 있다. 순서화된 그래프 뷰의 기본적인 단위는 노드이다. 노드는 HTML이나 웹 문서의 텍스트 엘리멘트와 1:1

대응이 된다. **노드집합**은 노드들의 묶음이다.

웹 페이지를 읽어 들이기 위해서 getpage나 post-page 연산자가 사용된다. 두 연산자는 URL과 폼 매개변수를 입력으로 받아서 해당되는 페이지를 각각 get 또는 post 방법을 통해 읽어 들이게 된다.

텍스트 스트링 뷰에서는 *ExtractText*(data, starting, ending)이라는 연산자가 정의되었다. ExtractText는 웹 문서로부터 starting과 ending으로 구별되는 모든 텍스트 영역을 추출해 낸다. Starting과 ending값은 정규식으로 처리가 되며, '^'과 '$'는 각각 문서의 맨 처음과 끝을 나타내기 위해 사용된다.

다음 절에서는 순서화된 그래프 뷰에서 정의된 연산자에 대해서 설명한다. *elem*(객체, 엘리멘트 이름, 위치)은 객체의 자식 노드들로부터 엘리멘트 이름에 일치하는 노드나 노드 집합을 추출한다. 위치가 지정되지 않거나 객체가 노드 집합인 경우, 이 연산자의 결과 값은 노드 집합이 된다. 객체가 노드거나 위치가 단일 값인 경우에는, 결과값은 노드가 된다. 예를 들어 elem($h, 'li')은 엘리멘트가 'li' 노드인 노드 집합을 반환한다. *elem_w* 연산자는 객체의 모든 후손 노드로부터 검색을 하게 된다. *pattern*(객체, 정규식, 위치) 연산자는 정규식에 일치하는 텍스트 데이타를 가진 노드나 노드 집합을 반환한다. *pattern_w*는 모든 후손 노드로부터 검색을 하게 된다. *aget*(객체, href) 연산자는 HTML 노드로부터 링크 정보를 반환하게 된다. *aget*(객체, htext) 연산자는 링크의 텍스트 정보를 반환하게 된다. 위치 연산자로 before, after 연산자가 있다. *before*(o1, o2)는 순서적으로 o2 앞에 오는 o1 노드 집합을 반환하고, *after*(o1, o2)는 o2 이후에 나타나는 o1 노드 집합을 반환한다. 계층 연산자로서 in(o1, o2)가 정의되어 있다. *in*(o1, o2)는 o2 포함되어 있는 o1 을 반환한다. o1의 시작 위치가 o2보다 크거나 같고, 끝 위치가 o2보다 작거나 같을 때 포함되어 있다고 정의한다.

HTML 문서에 대한 두 가지 뷰인 텍스트 스트링 뷰와 순서화된 그래프 뷰는 서로간에 손쉽게 변환이 된다. 두 가지 뷰를 변환하기 위해 *to_string*과 *to_node*가 정의 되어 있다. 이 외에도 노드 집합에 대한 합집합, 교집합, 차집합 등을 구하기 위한 집합 연산자들이 정의되어 있다.

추출된 정보는 내부적으로 스트링과 스트링 리스트로 관리가 된다. 추출된 정보는 두 가지 방법에 의해 XML로 변환된다. Perl에 익숙한 래퍼 디자이너는 내부적으로 관리되는 이 정보로부터 Perl 언어를 이용하여 XML 문서를 만들 수 있다. XWS에 제공되는 중첩된 스트링을 XML로 변환하는 함수와 연산자로 XML을 만들 수 있다. XML을 만드는 다른 방법으로는 사용자로부터 정의된 XML 생성 템플릿으로부터 만들어 내는 방법이다. XML 생성 템플릿과 내부 중첩 스트링을 이용하여 XWS는 XML 파일을 만들어 낸다. 이 방법은 W4F[13]에서 사용하는 방법과 유사하다. 그러나 W4F에서는 XML 파일을 만들어 내기 위한 Java 프로그램을 만들어 내는 반면, XWS에서는 이 처리 규칙들이 그 자체로 Perl 엔진에 의해 처리된다는 점에서 차이가 있다.

각 XWS 스크립트는 독립적으로 이름 붙어 있고, 잘 알려진 저장소에 저장이 된다. XWEET에서 웹 문서를 처리할 때, 존재하지 않는 XWS 입력 스크립트에 대해서는 이 알려진 저장소로부터 자동으로 다운로드 받아서 수행을 하게 된다. 이것은 XWS 스크립트가 Perl 스크립트로 이루어져 있고, 플랫폼에 독립적이기 때문에 가능하다.

```
template    :=  leaf | record | list
leaf        :=  '.' tag | '.' Tag '^'
list        :=  '.' tag '*' template
record      :=  '.' tag '('templist')'
templist    :=  template | template templist
tag         :=  STRING
```

그림 3 XML 생성 템플릿

그림 3에 XML 생성 템플릿의 BNF가 나와 있다. leaf 템플릿은 스트링 인자를 받아서 XML 엘리멘트를 생성한다. 예를 들어 .title 템플릿은 <!ELEMENT title (#PCDATA)>를 생성하고 .id^는 엘리멘트의 어트리뷰트 값을 생성한다. list 템플릿은 동일한 타입 리스트를 가진 XML 엘리멘트를 생성해내기 위해 사용된다. 예를 들어 논문의 저자가 한 명 이상인 경우 이를 위한 생성 템플릿은 .authorlist* .author 템플릿으로 정의될 수 있다. XML 매핑 모듈은 이 템플릿을 <!ELEMENT authorlist (author*)>와 <!ELEMENT author (#PCDATA)>로 변환한다. 중첩 스트링 값으로는 저자에 대한 리스트가 들어 있다. record 템플릿은 다른 타입리스트를 가진 엘리멘트를 만들어 내기 위해 사용된다. .item(.id^ .authorlist* .author .title)은 <!ELEMENT item (authorlist, title)>로 변환이 된다. item에 대한 자식 엘리멘트에 대해서는 재귀적으로 호출되어 처리된다. 자세한 예는 5 장에 나타나 있다. DTD와 XML 문서는 이 생성 룰에 의해 자동으로 만들어지게 된다.

5. XWS의 구현 및 사용 예

웹 래퍼는 자바나 C와 같은 타입을 가진 프로그램 언어[8,12]로 구현될 수 있다. 그리고 때로는 웹 문서를 처

리하기 위한 독자적인 추출 언어[13]로 만들어지기도 한다. 타입을 가진 언어는 컴파일러로 하여금 타입에 관련된 에러를 찾아내고 특정 코드를 생성해 냄으로써 바이너리 코드를 최적화할 수 있게 한다. 그리고 사용자로 하여금 큰 프로그램을 관리 가능하게 만들어준다. 그러나 이런 타입 제한 조건들이 코드의 재사용성을 떨어뜨린다[18]. 특히 타입 시스템은 래퍼 프로그램에서 데이타들 사이의 타입 변환을 위해 프로그래머가 형변환을 정의해 주어야 하는 문제가 있다. 그러므로 웹 래퍼를 디자인하는데 있어서 효율적이지 못하다. 그래서 XWS 시스템에서는 상대적으로 타입에 자유로운 Perl을 이용하여 구현하였다. XWS 스크립트 언어는 Perl로 구성되어 있으며, 래퍼로서 동작을 하는 각각의 스크립트들은 잘 알려진 저장소에 저장되었다가 실시간으로 다운로드 받아서 수행이 가능하다. 이 것은 XWS 스크립트가 제공 되는 라이브러리와 함께 Perl 인터프리터 언어에서 직접 수행이 되기 때문에 가능하다.

XWS 스크립트 인터페이스는 다음과 같은 특징을 가지도록 설계되었다. XWS의 데이타 모델로 사용되는 HTML 페이지에 대한 통합적 뷰가 가능하도록 하였으며, 순차적 그래프 뷰에서 HTML 파일의 각 부분은 객체 또는 텍스트 스트링으로 간주한다. 래퍼 디자이너는 XWS 스크립트를 이용하여 HTML 문서를 텍스트 스트링 뷰 또는 순서적 그래프 뷰로 볼 수 있다. HTML의 계층구조로부터 정보를 얻을 때는 순서적 그래프 뷰를 사용할 수 있다. 그리고 패턴 매칭을 위하여 정규식이 사용되었다. HTML 페이지를 세밀하게 제어하기 원하는 래퍼 디자이너는, XWS에서 제공되는 선언적 질의 언어뿐만 아니라 Perl의 순차적 처리 기능을 이용할 수 있도록 하였다.

여기에서는 Michael Ley의 DBLP 페이지[1]로 부터 래퍼 디자이너가 XML 문서를 생성해 내는 예를 통하

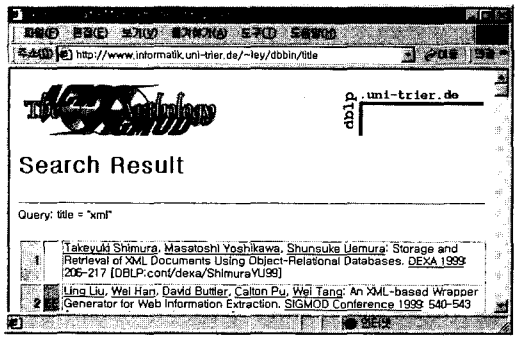

그림 4 Michael Ley의 DBLP 페이지

1) URL은 http//www.informatik.uni-trier.de/~ley/dbbin/title이다.

여 XWS의 동작 및 래퍼 설계 과정을 설명하기로 한다. XWEET 시스템에서 일부 데이타는 XML 파일로 저장되어 있고 일부 데이타는 XWS를 통하여 웹 데이타 소스로부터 실시간으로 생성이 된다. 해당 페이지가 자주 수정이 안 되는 경우에는 미리 변환된 XML 데이타를 XWEET의 PDM에 저장하여 성능을 향상시키기도 한다. DBLP 페이지를 웹 브라우저에서 접근했을 때의 스크린 뷰가 그림 4에 나타나 있고, 해당 페이지를 표시하는 HTML 소스가 그림 5에 나타나 있다.

엘리멘트 타입의 노드는 애트리뷰트 값을 가지고 있다. 이 노드에 붙여진 번호는 루트 노드로부터 자식 노드로 갈 때마다 레벨이 하나씩 증가하여 붙여진다. 그리고 같은 형제(sibling) 노드끼리는 뒷자리 번호가 증가하여 붙여지게 된다. 이 정보는 디버깅 목적과, 각 노드 사이의 순서관계를 부여하기 위하여 사용된다. 예에서 <body>와 </body> 사이의 자식 노드는 각각 1.2.3, 1.2.4와 같은 형태로 번호가 붙여졌다. 파싱된 엘리멘트에 대한 자세하고 세밀한 제어를 원하는 사용자는 이 값을 이용해서 각각의 정보에 접근할 수도 있고, 고 수준의 처리 언어를 이용하여 값들을 처리할 수도 있다. XWS 스크립트 언어에서 각 엘리멘트 간의 순서관계나 포함관계를 통해 데이타를 얻어내고자 할 때 사용되기도 한다. 해당 페이지를 표현하는 HTML 소스와 이 페이지를 내부적인 웹 페이지 데이타 모델로 표현한 그래프[2]가 각각 그림 5와 그림 6에 나타나있다. 그림 4의 페이지로부터 XML을 추출하기 위하여 XWS에 사용되는 설정 파일은 다음과 같다. XWS에서 사용되는 각각의 설정 파일은 XWEET 시스템에서 유일한 이름으로 명명되고 사용된다. DBLP exec:/home/httpd/xweet/xws/main.pl?title=xml 값이 XWEET 시스템의 질의어 처리기에 의해서 읽혀진다. 질의어 처리기에서는 XWS 데이타 입력 소스를 DBLP라는 유일한 이름으로 인식하고 처리하게 된다.

먼저 추출 모듈에서 getpage()함수를 이용하여 해당하는 페이지를 가져오게 된다. 추출 함수에서 XWS 시스템 내에 캐싱되었는지 여부와 페이지가 얼마나 자주 변경되는지에 대한 정보를 가지고 페이지를 읽어올지 판단을 하게 된다. 이 추출 모듈의 결과로 반환되는 것은 그림 5와 같은 HTML 페이지이다. 이렇게 추출된 HTML페이지는 가공 모듈의 입력 값으로 주어지게 된다. XWS::Node의 입력으로 처리된 페이지는 내부에서 그림 6과 같은 내부 HTML 모델로 표현되게 된다. 내부적으로 트리 형태로 표현된 HTML페이지에 4장에서

2) 여기서는 간략하게 나타나기 위해 텍스트 영역은 제외하고 엘리멘트 계층 구조만 표시하였다.

```
<HTML><HEAD><TITLE>Search Result</TITLE></HEAD> <BODY bgcolor="white" text="black" link
="black"> <table width="100%"><tr><td align="left"><a href="http://www.informatik. uni-trier.de/
ley/db/anthology.html"><img alt="ACM SIGMOD Anthology" src="http://www.informatik.uni-trier.de/
ley/db/AnLogo.gif" border=0 height=60 width=233></a></td><td align="right"><a href="http://www.
informatik.uni-trier.de/ ley/db/index.html"><IMG alt="dblp.uni-trier.de" src="http://www.informatik.uni-
trier.de/ ley/db/Logo.gif"border=0 height=60 width=170></a></td></tr></table> <h1>Search
Result</h1><hr>Query: title = "xml"<p> <table border=1><tr><td align="right"
bgcolor="#CCCCFF">1</td><td>  </td><td><a href="http://www.informatik.uni-trier.de/
ley/db/indices/a-tree /s/Shimura:Takeyuki.html">Takeyuki Shimura</A>, <a href="http://www.
informatik.uni-trier.de/ ley/db/indices/a-tree/y/Yoshikawa:Masatoshi.html"> Masatoshi Yoshikawa</A>, <A
href="http://www.informatik.uni-trier.de/ ley/ db/indices/a-tree/u/Uemura:Shunsuke.html">Shunsuke
Uemura</a>: Storage and Retrieval of XML Documents Using Object-Relational Databases. <a
href="http ://www.informatik.uni-trier.de/ ley/db/conf/dexa/dexa99.html#ShimuraYU99"> DEXA 1999</a>:
206-217 [DBLP:conf/dexa/ShimuraYU99]</td></tr> .....
```

그림 5 그림 4의 HTML 소스

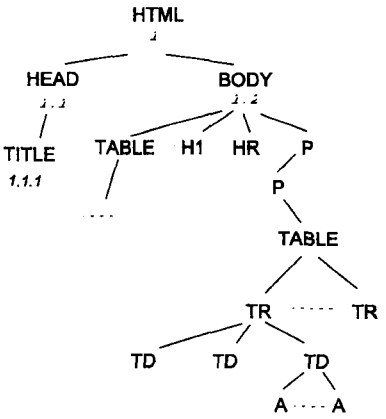

그림 6 내부 HTML 모델로 표현된 Ley 참고 문헌 페이지

```
$html = getpage('http://www.informatik.uni-trier.de/ ley/dbbin/title?title=xml');
$h = new XWS::Node $html;
$r = $h->elem_w('table',1)->elem_w('tr')->elem_w('td',2);

@string = $r->to_flat_string;
$result = convert_nl(\@string);
$xml = new XWS::Mapping ".thesis*.item(.id^ .authorlist*.author .title)\n", $result;
$xml->print_dtd();
$xml->print_xml();
```

그림 7 XWS에서 사용되는 스크립트 파일

설명한 스크립트 언어를 이용하여 원하는 부분만 추출하게 된다. 여기서 추출하고자 하는 데이타는 두 번째 테이블의 각 <TR> 태그 중에서 세 번째로 나오는 <TD> 값이므로 $r = $h->elem_w('table',1)->elem_w('tr')->elem_w('td',2); 로 표현이 된다. 앞에서 설명했듯이 elem_w은 트리 계층구조에서 자손 노드 중에 해당하는 엘리멘트를 찾아 리스트로 반환하는 함수이다. 추출된 엘리멘트 집합에다가 다시 elem_w 함수를 적용하는 경우 각각의 집합 원소에 대해서 처리를 한 다음 다시 집합으로 반환하게 된다. 이 문장의 수행 결과 $r에는 추출하기 원하는 <TD>의 값에 해당하는 정보가 리스트로 저장이 되어 있다. 추출된 정보는 대응 모듈을 통하여 XML로 출력한다. XWS::Mapping 객체에서 하는 일은 출력될 XML의 형태를 결정하는 템플릿 값과 가공 모듈에서 추출된 정보를 바탕으로 생성된 NL (*nested list*) 값을 바탕으로 XML의 DTD와 데이타 값을 만들어 낸다. to_flat_string은 추출된 엘리멘트 집합의 원소를 각각 스트링으로 변환하는 함수이다. 변환된 스트링은 convert_nl 루틴에 의해 NL로 바뀌게 된다. 이 루틴은 XWS::Mapping에 인자 값으로 들어가는 템플릿에 맞는 형태의 NL을 반환하도록 구성되었다.

추출된 엘리멘트로부터 NL을 만들기 위한 함수는 Perl 함수로 이루어져 있다. Perl은 정교한 패턴 매칭 연산자와, 다양한 문자열 처리 함수 등을 제공하고 있기 때문에 스트링으로 변환된 내부 값들은 이를 이용하여 NL로 변환하여 반환된다. 위의 예에서 title의 경우 스트링으로, author의 경우 스트링에 대한 리스트 값으로 변환하여 반환된다. XML로 변환된 출력 결과가 그림 9에 나와 있다. 이 결과물은 print_dtd, print_xml의 수행 결과이다.

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<!DOCTYPE XWS_DOC [
        <!ELEMENT thesis (item)*>
        <!ELEMENT item (authorlist,title)>
        <!ATTLIST item id CDATA #IMPLIED>
        <!ELEMENT authorlist (author)*>
        <!ELEMENT author (#PCDATA)>
        <!ELEMENT title (#PCDATA)>
]>
<XWS_DOC>
        <thesis>
        <item id="0">
        <authorlist>
                <author>Takeyuki Shimura</author>
                <author>Masatoshi Yoshikawa</author>
                <author>Shunsuke Uemura</author>
        </authorlist>
        <title>Storage and Retrieval of XML Documents Using Object-Relational Databases</title>
        </item>
        ...
        <item id="99">
        <authorlist>
                <author>Bert Bos</author>
        </authorlist>
        <title>XML: From Bytes to Characters</title>
        </item>
        </thesis>
</XWS_DOC>
```

그림 9 XML로 변환된 HTML 페이지

```
sub convert_nl
{
        my $args = shift;
        my @result;
        my $id;

        foreach (@{$args}) {
                $_ = ~ s/&/&amp;/g;
                my ($author, $title) =
                        ($_ =~ /[^:]+):\s+([^\.\?]*)(\.|\?)/ );
                $authorref = [split /,/, $author];
                push (@result, [$id++, $authorref, $title]);
        }
        \@result;
}
```

그림 8 XWS에서 XML 매핑

6. 결 론

이 논문에서는 XWEET 시스템에서 래퍼 프로그램의 생성을 위해 사용되는 XWS의 데이타 모델과 소프트웨어 개발에 관해 보였다. XWS는 여러 웹 사이트로부터 웹 페이지를 추출하여 그것들은 입력 DTD 템플릿에 기술된 XML 문서로 변환하는 일을 한다. XWS는 Perl 언어를 이용하여 만들어졌다. XWS은 XWS 라이브러리와 프로그램들로 이루어져 있고, Perl의 스트링 처리 능력을 효율적으로 이용하고 있다. XWS의 각 연산자들은 객체지향적으로 디자인되어 래퍼 디자이너가 손쉽게 원하는 데이타를 추출할 수 있도록 하였고, 복잡한 처리가 필요한 래퍼는 Perl의 저 수준 스크립트 언어를 이용하여 처리할 수 있도록 하였다.

XWS는 세 가지 구별되는 특징을 가지고 있다. 첫째, XWS에서 사용되는 데이타 모델과 연산자들은 HTML 문서의 여러 뷰들에 대한 통합적인 모델을 지원한다. 어떤 HTML 문서에서는, HTML 문서를 텍스트 스트링으로 처리하는 것이 HTML 문서를 순서 그래프 뷰로 보는 것보다 영역의 추출을 쉽게 만들기도 한다. 때로는 그 반대의 경우도 있다. XWS에서는 HTML 문서를 여러 뷰로 볼 수 있는 데이타 모델을 제공한다. 둘째, XWS는 사용자가 이용 편이한 XWS 생성 프로그램을 제공하고 있다. 셋째, 우리는 XWS 스크립트 언어를 객체지향적 방법론에 의해 설계하였다. 대부분의 래퍼 디자이너가 XWS 스크립트 언어를 이용하여 대부분의 웹 문서를 처리할 수 있지만, 복잡한 웹 문서를 처리하기 위해서 Perl의 내장 함수나 라이브러리를 이용할 수 있

도록 하였다.

XWS 스크립트는 Perl 인터프리터에 의해 처리가 되기 때문에, 스크립트 파일은 여러 개의 XWS 시스템들 간에 공유가 된다. 웹을 위해 디자인된 언어[13]는 기존의 프로그래밍 언어에 대해 연산자의 다양성이나 프로그램의 처리 능력면에서 뒤떨어진다. 그리고 자바로 이루어진 래퍼 모듈은 바이트 코드로 컴파일되어 처리되어야 한다.

현재 우리는 XWS 래퍼 생성 시스템을 UML에 기반한 그래픽 사용자 인터페이스를 가지도록 확장하고 있다. 그리고 사용자가의 웹 항해에서 얻어진 정보를 바탕으로 자동으로 스크립트 파일을 생성해 내는 연구를 진행 중이다. 향후 연구과제로는, HTML 페이지로부터 중요한 정보를 자동으로 추출해 낼 수 있는 인공지능 또는 발견적인(heuristic) 모듈을 추가하는 일이 있다. 기존에 사용되는 XWS 스크립트 파일을 바탕으로 유사한 구조의 HTML 파일로부터 XML DTD를 만들어내거나 의미 있는 영역을 찾아주는 일이다.

참 고 문 헌

[1] World Wide Web Consortium (W3C). Extensible Markup Language (XML) 1.0, 1998. http://www.w3.org/TR/1998/REC-xml-19980210.

[2] World Wide Web Consortium (W3C). Extensible Stylesheet Language(XSL), 1998. http://www.w3.org/Style/XSL.

[3] 정재목, 박상원, 정태선, 이병준, 민경섭, 이강우, 김형주. XWEET: 웹 환경을 위한 통합 데이타베이스 시스템. *정보과학회*, 28(2), 2001.

[4] XWEET Team. XWEET System. Technical report, Seoul National University, Feb 2000. http://oopsla.snu.ac.kr/xweet/xweet.ps.gz.

[5] D. Florescu, A. Levy, and A. Mendelzon. Database techniques for the World-Wide Web: A survey. *SIGMOD Record (ACM Special Interest Group on Management of Data)*, 27(3):59--74, 1998.

[6] S. Chawathe, H. Garcia-Molina, J. Hammer, K. Ireland, Y. Papakonstantinou, J. D. Ullman, and J. Widom. The TSIMMIS project: Integration of heterogeneous information sources. In *16th Meeting of the Information Processing Society of Japan*, pages 7--18, Tokyo, Japan, 1994.

[7] J. Hammer, H. Garcia-Molina, S. Nestorov, R. Yerneni, M. Breunig, and V. Vassalos. Template-based wrappers in the TSIMMIS system. In *Proceedings of the ACM SIGMOD International Conference on Management of Data*, volume 26,2 of SIGMOD Record, pages 532--535, New York, May 13--15 1997. ACM Press.

[8] G. Huck, P. Fankhauser, K. Aberer, and E. J. Neuhold. Jedi: Extracting and synthesizing information from the web. In *CoopIS 1998*, pages 32--43, 1998.

[9] A. Sahuguet and F. Azavant. Building light-weight wrappers for legacy web data-sources using w4f. In *VLDB*, 1999.

[10] B.Adelberg. NoDoSE - a tool for semi-automatically extracting semi-structured data from text documents. In L. M. Haas and A. Tiwary, editors, *SIGMOD 1998, Proceedings ACM SIGMOD International Conference on Management of Data, June 2-4, 1998*, Seattle, Washington, USA, pages 283--294. ACM Press, 1998.

[11] N. Kushmerick, R. Doorenbos, and D. Weld. Wrapper induction for information extraction. In *International Joint Conference on Artificial Intelligence*, 15, 1997.

[12] A. Sahuguet and F. Azavant. Web ecology: Recycling HTML pages as XML documents using W4F. In *WebDB'99*, 1999.

[13] T. Kistler and H. Marais. Automating the web: WebL, 1999. http://www.research.digital.com/SRC/WebL.

[14] L. Liu, C. Pu, and W. Han. Xwrap: An xml-enabled wrapper construction system for web information sources. In *ICDE*, 2000.

[15] W3C. *Document Object Level (DOM) Level 1 Specification*, oct 1998. http://www.w3.org/TR/.

[16] L. Wall, R. L. Schwartz, T. Christiansen, and S. Potter. *Programming Perl*. Nutshell Handbook. O'Reilly & Associates, 2nd edition, 1996.

[17] J. Hammer, H. Garcia-Molina, J. Cho, R. Aranha, and A. Crespo. Extracting semistructured information from the web. Technical report, Stanford University, 1998.

[18] J. K. Ousterhout. Scripting: Higher Level Programming for the 21st Century. *IEEE Computer*, 31(3):23--30, Mar. 1998.

정 재 목

서울대 계산통계학과 졸업. 서울대 전산과학과 석사 졸업. 서울대 컴퓨터 공학부 박사과정 재학

김 형 주

정보과학회논문지 : 컴퓨팅의 실제 제 9 권 제 3 호 참조