2000년 한국정보처리학회 추계 학술발표논문집 제7권 제2호

Libwww을 이용한 단순 웹브라우저의 설계 및 구현

도유환*, 오삼권*, 임기욱**
*호서대학교 컴퓨터공학부
**선문대학교 산업공학과
avenue1@netian.com, ohsk@office.hoseo.ac.kr
rim@omega.sunmoon.ac.kr

The Design and Implementation of A Simplified Web-Browser Using Libwww

Yu Hwan Do*, Sam Kweon Oh*, Kee Wook Rim**
*School of Computer Engineering, Hoseo University
**Dept of Industrial Engineering, Sunmoon University

요 약

많은 웹 브라우저들이 사용되고 있고 또 개발 중에 있다. W3C등의 인터넷 사이트에서는 Libwww를 비롯한 각종 패키지들의 소스를 공개하고 있어서 웹 브라우저를 포함한 인터넷 어플리케이션의 개발에 많은 도움이 되고 있다. 본 논문의 목적은 내장형(Embedded) 실시간 운영체제에서도 동작할 수 있는 단순화되고 가벼운 형태의 웹 브라우저를 설계하고 구현하는 것이다. 브라우저의 단순화를 위해 일단 웹 페이지의 텍스트만을 처리하는 브라우저를 설계하고 구현하였다. 브라우저 개발도구로서 Libwww를 사용하였다.

1. 서론

Netscape Navigator, Internet Explorer, Opera를 비롯하여 Amaya, Arena, Bang Space, Doczilla, Express, Fortify, FreeWRL, gzilla, Lynx, MMM Browser, Mozilla, Qweb Browser, Techexplorer Hypermedia Browser, W3M Browser등 많은 웹 브라우저들이 현재 사용되고 있다[1].

W3C(World Wide Web Consortium)등의 인터넷 사이트에서 Libwww(HTTP를 포함한 클라이언트 웹 API의 구현 패키지)를 비롯한 각종 인터넷 어플리케이션의 소스들을 공개하고 있다. 이런 공개 소스들은 웹 브라우저를 포함한 인터넷 어플리케이션의 개발에 좋은 참고 자료가 되고 있다[2].

본 논문은 브라우저의 개발에 사용된 Libwww의 구조를 먼저 설명하고, 내장형 실시간 운영체제에서 사용될 수 있도록 단순화되고 가벼운 형태로 설계하고 구현한 웹 브라우저를 설명한다.

2장에서는 브라우저 구현에 필요한 Libwww의 구조를 소개하고 3장에서는 설계한 단순 브라우저의 구조를 설명하며 4장에서는 실제 사용 예를 보이고 5장에서 결론을 맺는다.

2. Libwww

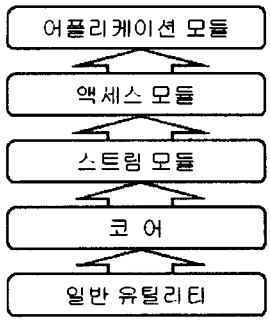
본 장에서는 브라우저 개발의 핵심이 되는 Libwww에 대해서 설명한다. Libwww는 W3C에서 제공하는 클라이언트 웹 API 패키지이며 2000년 8월 현재 5.2.8 버전이 배포 중이다.

Libwww는 HTTP와 URIs 뿐만 아니라 웹에 필요한 많은 코드를 포함하고 있으며, 주로 웹 로봇, 브라우저, GUI 어플리케이션, 자동화 도구(Automated Tool)등의 클라이언트 어플리케이션에 유용하다[3].

Libwww는 Tim Berners-Lee가 1992년 11월에 최초 버전을 구현했으며 그 후로 인터넷의 여러 분야에서 이용되었다. 텍스트 브라우저 Lynx, Mosaic 웹 브라우저 그리고 CERN 서버가 Libwww을 사용하였다.

2.1 기본 설계 모델

새로운 모듈의 필요 시, 라이브러리 내부의 변경 없이도 동적으로 모듈의 추가가 가능하도록 하는 것이 Libwww의 기본 설계 개념이며, 그 결과로서 라이브러리의 구조가 그림[1]과 같이 5개의 주요 부분으로 이루어져 있다[4].



그림[1] 라이브러리 계층도

그림에서 각 계층들의 하위계층은 상위계층에게 필요한 서비스를 제공한다.

다음은 Libwww의 주요 5개 부분의 설명이다.

(1) 일반 유틸리티(Generic Utilities)

컨테이너 클래스(Container Classes), 스트링 유틸리티(String Utilities), 네트웍 유틸리티(Network Utilities) 등의 일반적인 유틸리티 모듈들을 제공한다. 플랫폼에 따라 다르게 구현된 코드로부터 코아(Core)의 코드를 떼어내는 중요한 기능을 한다.

(2) 코어(Core)

라이브러리의 기초가 된다. 코아 외부에서 발생한 요구(Request)의 처리를 담당하고 서비스 요구에 대한 어플리케이션 프로그램의 인터페이스를 제공한다.

(3) 스트림 모듈(Stream Modules)

모든 데이터는 스트림(Stream)을 사용해서 어플리케이션과 네트웍 사이를 이동한다. 스트림은 입력 데이터의 변형과 출력을 담당한다.

(4) 액세스 모듈(Access Modules)

HTTP, FTP, Gopher, WAIS, NNTP, Telnet, rlogin, TN3270 등의 프로토콜(protocol) 액세스 모듈들을 포함하고 있다.

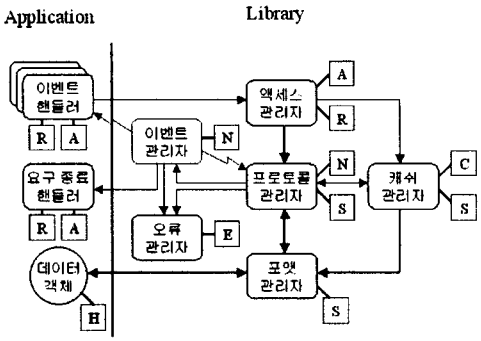
(5) 어플리케이션 모듈(Application Modules)

User Interaction, management of history lists, call back functions, logging 등의 기능을 포함하는 클라이언트 어플리케이션들이다.

2.2 코어(Core) 모듈과 객체

코어 모듈은 Libwww의 기초가 되는 중요한 부분이므로 자세히 설명한다.

그림[2]는 코어 모듈들과 코어 객체(Object)들과의 관계를 보여주고 있다.



그림[2] 코어 모듈들과 코어 객체들과의 관계

R: HTRequest Object
A: HTAnchor Object
N: HTNet Object
E: HTError Object
S: HTStream Object
C: HTCache Object
H: HTHyperDoc Object

- 액세스 관리자(Access Manager)
  데이터 객체 요구가 전달되는 곳이다. URI를 GET, POST하거나 찾는 등의 서비스를 어플리케이션에게 허가한다.
- 프로토콜 관리자(Protocol Manager)
  메모리나 파일 캐쉬에서 발견되지 않는 문서를 얻기 위해서 액세스 관리자에 의해서 호출된다. HTTP, FTP, NNTP, Gopher, WAIS, Telnet, local file system 등을 다루는 프로토콜 모듈들의 집합이다.
- 포맷 관리자(Format Manager)
  요청된 데이터 포맷의 변형을 담당한다.
- 캐쉬 관리자(Cache Manager)
  네트웍에서 다운로드된 데이터 객체들을 저장한다.
- 오류 관리자(Error Manager)
  서버와의 통신 중에 생긴 모든 오류들 또는 단순

히 현재의 상태를 저장하고 있는 메모리를 관리한다.

- 이벤트 관리자(Event Manager)
  내부 이벤트 관리자(Internal Event Manager)는 내부 libwww 스레드(Thread)를 관리하고, 외부 이벤트 관리자(External Event Manager)는 외부 스레드를 관리한다.
- 이벤트 핸들러(Event Handlers)
  새로운 요구, 요구 중지, 화면 스크롤 등의 사용자의 행동들을 다루기 위한 이벤트 핸들러들의 집합이다.
- 요구 종료 핸들러(Request Termination Handler)
  사용자 요구가 종료되었을 때, 히스토리 리스트(History List)등의 갱신을 위해 필요하다.
- 데이터 객체(Data Object)
  포맷 관리자에 의해 변형된 데이터

2.3 Libwww의 인터페이스 모듈

Libwww은 사용자의 편의와 모듈의 확장성을 위해 현재 인터페이스 모듈을 다음과 같이 나누고 있다. 각 모듈들의 기능은 모듈의 이름에 명시적으로 나타나 있으므로 추가로 설명하지 않는다.

- Basic Utilities
- Libwww core functions
- Initialization Utilities
- Application Utilities and Access functions
- Transport Package
- Mux Protocol
- MIME Package
- Local file access
- Gopher access
- FTP access
- HTTP access
- News access
- Telnet access
- WAIS access
- Persistent HTTP/1.1 cache
- Stream Package
- Directory Browsing Package
- Zlib deflate/inflate streams
- HTML parser
- XML parser
- SQL logging facilities

Libwww의 각 모듈들은 해당 모듈별 API함수가 매우 잘 구현되어있기 때문에 브라우저를 비롯한 웹 어플리케이션 개발에 매우 유용하다.

3. 브라우저 구성

Libwww을 이용하여 설계 및 구현한 가장 단순한 형태의 웹 브라우저는 그림[3]과 같이 코어와 유틸리티 모듈을 기본으로 HTTP, MIME, HTML, Transport, Application Interface 모듈을 사용하였다. 브라우저 각 모듈의 기능은 표[1]에 요약하였다.

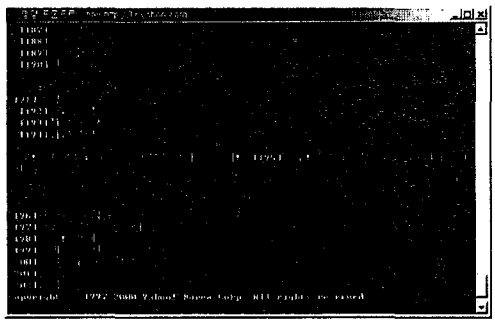
모 듈	기능
Utilities	기본 유틸리티
Core	기초 모듈, registration 모듈
Application Interface	어플리케이션 인터페이스 이벤트 메니저
HTTP	HTTP 프로토콜 사용
HTML	Hypertext Object 처리
MIME	MIME 사용
Transport	소켓(socket) 전송

표[1] 구현한 단순 브라우저를 구성하는 각 모듈의 기능

구현된 단순 웹브라우저는 단순히 입력 URL의 첫 페이지의 텍스트 만을 출력해 주는 기능을 수행하며, 이미지를 포함한 기타의 웹 페이지 구성요소들은 출력하지 않는다.

4. 실 험

그림[3]는 브라우저의 실행 모습이다.



그림[3] 브라우저 실행 모습

야후 웹 사이트(http://kr.yahoo.com)의 첫 페이지를 가져와서 화면에 출력해본 결과이다.

브라우저의 처리 대상에는 이미지 및 기타 요소들은 포함되지 않았기 때문에 오직 텍스트 화면만을

출력한다.

가장 단순하게 구현하기 위해서 텍스트 위주로 구현하였고, 사용자 인터페이스를 비롯한 기타 브라우저 구성요소들은 추후 구현할 예정이다.

5. 결론 및 향후과제

W3C에서 제공하는 Libwww 패키지에 대해서 알아보고 이를 이용하여 최소 기능의 웹 브라우저를 설계하고 구현하였다.

앞으로 브라우저의 추가 기능들과 사용자 인터페이스를 구현하고 이 브라우저를 내장형 실시간 운영체제에 포팅(Porting)하는 작업을 수행할 예정이다.

참고문헌

[1]http://www.linux.com/links/Software/Web_browsers/
[2]http://www.w3.org/
[3]http://www.w3.org/Library/Activity.html
[4]http://www.w3.org/Library/User/Architecture/DesignModel.html
[5]Libwww 5.2.8 User Manual