# C 및 Prolog 언어용 웹 입출력 라이브러리

신동하*

## WEBIO Libraries for C and Prolog Languages

Dongha Shin*

본 연구는 상명대학교 자연과학연구소가 연구비를 지원하여 수행되었습니다

요 약

인터넷이 등장하기 전에는 대부분의 응용 프로그램이 표준입출력 장치로 호스트 컴퓨터에 연결된 단말기를 사용하며 수행되었다. 오늘날은 인터넷의 사용이 보편화되었고 많은 서비스가 인터넷에 연결된 웹 브라우저 상에서 제공된다. 단말기를 통한 표준입출력 방식과 웹 브라우저를 통한 통신 입출력 방식은 기술적으로 다르기 때문에 과거에 작성된 단말기용 응용 프로그램을 쉽게 인터넷을 통하여 웹 브라우저 상에서는 수행시킬 수는 없다. 본 논문은 표준입출력 방식으로 작성된 프로그램의 소스를 수정하지 않고 다시 컴파일만 하여 인터넷에 연결된 웹 브라우저 상에서도 서비스할 수 있게 해주는 웹 입출력(WEBIO) 라이브러리의 개발에 대하여 기술한다. 현재 C 및 Prolog 언어 용 웹 입출력 라이브러리가 개발되어 시험 중이다.

ABSTRACT

Before the Internet was available, most application programs were executed using terminals, connected to host computers, as standard input output devices. The Internet is popular today and many services are provided on web browsers connected to the Internet. Since the standard I/O method used for terminals is different from the communication I/O methods used for web browsers, it is not possible to execute many application programs developed for the conventional terminals on web browsers. In this paper, we describe Web Input Output(WEBIO) library that enables application programs running on terminals to be executed on web browsers by recompilation without source modification. The WEBIO libraries for C and Prolog languages have been developed and they are under test now.

키워드
인터넷, 프로그래밍 도구, C 언어, Prolog 언어

## Ⅰ. 서 론

인터넷이 등장하기 전에는 대부분의 응용 프로그램이 호스트 컴퓨터에 연결된 터미널을 표준입출력 장치로 사용하며 수행되었으나 인터넷의 사용이 보편화된 요즘에는 많은 서비스가 웹 브라우저 상에서 제공된다. 단말기를 통한 표준입출력 방식과 웹 브라우저를 통한 통신 입출력 방식은 기술적으로 매우 다르기 때문에 예전에 작성된 많은 응용 프로그램을 쉽게 인터넷에 연결된 웹 브라우저 상에서는 서비스 할 수는 없다. 본 논문은 표준입출력 방식으로 작성된 프로그램을 수정하지 않고 다시 컴파일만 하여 웹 브라우저 상에서도 서비스할 수 있게 하는 웹 입출력(WEBIO: Web

*상명대학교 자연과학대학 소프트웨어학부

접수일자 : 2002. 5. 4

Input Output) 라이브러리의 개발에 대하여 기술한다. 본 라이브러리는 C[4][12] 및 Prolog[3][5] 언어의 표준입출력(STDIO: Standard Input Output) 라이브러리[10] 함수에 대응하는 웹 입출력 라이브러리 함수와 관련한 CGI[8] 프로그램으로 구성된다.

본 연구와 비슷한 최근의 연구로 C 언어용으로는 cgic[1] 라이브러리 및 cgi-util[6] 라이브러리 등이 있고 Prolog 언어용으로는 PiLLoW[2] 라이브러리, WebLS[10] 시스템 및 ProWeb[7] 서버 등이 있으나 이들 모두 응용 프로그램의 상태를 유지하면서 수행시킬 수 없기 때문에 단순 CGI 프로그램 개발 라이브러리에 지나지 않는다. 즉 이들은 웹 브라우저의 입력을 받아 한 페이지의 HTML 문서를 생성하고 종료하기 때문에 일반 응용 프로그램을 웹 브라우저 상에서 상태를 유지하며 수행시키는 기능은 가지고 있지 않다.

본 논문의 2장에서는 전체 시스템 구조를 정의하고 3장에서는 시스템의 동작 원리 및 개발한 라이브러리의 특징에 대하여 기술한다. 4장에서는 시스템 설계에 대하여 기술하고 5장에서는 C 및 Prolog 언어용 WEBIO 라이브러리 개발에 대하여 기술한다. 6장에서는 개발된 WEBIO 라이브러리의 장점을 기술하고 결론을 맺는다.

## II. 시스템 구조

본 장에서는 기존의 표준입출력 라이브러리를 사용한 응용 프로그램을 수정하지 않고 인터넷에 연결된 웹 브라우저 상에서 수행시키기 위한 전체적인 시스템 구조에 대하여 기술한다. 먼저 시스템을 구성하는 각 요소를 정의한 후 이들이 서로 도와서 어떻게 웹 브라우저 상에서 응용 프로그램을 수행시키는지를 기술한다. 그림 1은 전체 시스템 구조를 그림으로 나타낸 것이다.

전체 시스템은 4개의 요소인 웹 브라우저, HTTP[14] 서버, CGI[8] 프로그램 및 WEBIO라이브러리를 사용하는 응용 프로그램으로 구성된다. 여기서 WEBIO라이브러리는 기존 응용 프로그램이 사용한 STDIO 라이브러리를 대체한 것이다. 웹 브라우저는 사용자가 수행시키는 클라이언트 프로그램이다.

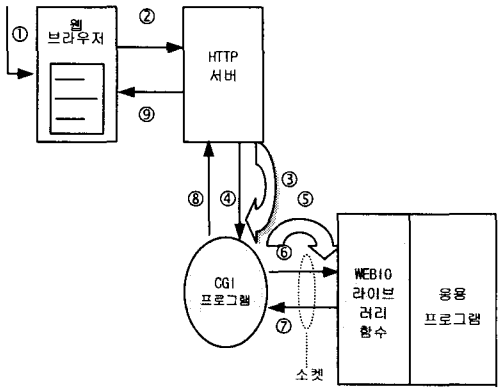


그림 1. 시스템 구조
Fig. 1 System Architecture

사용자는 웹 브라우저에서 URL을 입력하여 주어진 응용 프로그램을 시작시키며 응용 프로그램이 수행 중일 때는 웹 브라우저를 통하여 입출력을 수행한다. HTTP 서버는 응용 프로그램이 수행되는 호스트 컴퓨터에서 수행된다. HTTP 서버는 응용 프로그램을 수행시키기 위하여 사용자가 웹 브라우저에서 보낸 URL을 받아서 관련 CGI 프로그램을 수행시켜서 응용 프로그램을 시작시킨다. 또 HTTP 서버는 웹 브라우저와 CGI 프로그램 사이의 데이터 전달 기능을 수행한다. CGI 프로그램은 HTTP 서버와 응용 프로그램 사이에서 데이터 전달 기능을 수행한다. CGI 프로그램은 웹 브라우저에서 입력이 있을 때마다 수행되었다가 종료한다. 하나의 응용 프로그램이 시작하여 종료하는 동안 CGI 프로그램은 여러 번 수행과 종료를 반복한다. 응용 프로그램은 WEBIO 라이브러리와 링크 되어 있으며 수행을 시작하여 끝날 때까지 여러 번의 웹 입출력을 발생시킨다.

앞에서 정의한 4개의 요소가 서로 협조하여 웹 브라우저 상에서 수행되는 과정은 다음과 같다. 각 과정의 번호는 그림 1에 표시된 숫자와 같다.

① 사용자는 수행할 응용 프로그램을 지정하는 URL을 입력하거나 응용 프로그램이 수행되는 도중에는 원하는 데이터를 입력할 수 있다.

② 웹 브라우저는 사용자의 입력을 인터넷을 통하여 HTTP 서버에게 전달한다.

③ HTTP 서버는 사용자의 입력이 있을 때마다

CGI 프로그램을 수행시킨다. 이 때 CGI 프로그램은 응용 프로그램으로부터 받은 데이터를 HTTP 서버에게 전달한 후 곧 종료한다.

④ HTTP 서버는 웹 브라우저로부터 받은 데이터를 CGI 프로그램에게 전달한다.

⑤ 처음 수행된 CGI 프로그램은 응용 프로그램을 수행시킨다. 이 동작은 사용자가 웹 브라우저에서 응용 프로그램에 해당하는 URL을 입력할 때 한번만 일어난다.

⑥ CGI 프로그램은 소켓 IPC[11]를 사용하여 HTTP 서버로부터 받은 데이터를 응용 프로그램에게 전달한다.

⑦ 응용 프로그램은 수행 도중 출력할 데이터를 WEBIO 라이브러리를 통하여 CGI 프로그램에게 전달한다. 이때 소켓 IPC가 사용된다.

⑧ CGI 프로그램은 응용 프로그램으로부터 받은 데이터를 가지고 생성한 HTML[13] 문서를 HTTP 서버에게 전달한다.

⑨ HTTP 서버는 CGI 프로그램으로부터 받은 HTML 문서를 웹 브라우저에게 전달한다. 이 HTML 문서는 응용 프로그램이 출력한 데이터와 다음 입력을 받기 위한 입력 HTML 폼을 포함하고 있다.

## III. 동작원리

본 장에서는 웹 브라우저와 응용 프로그램 사이의 데이터 전달에 필요한 4가지 동작을 정의한다. 이 4가지 동작을 통하여 기존의 STDIO 라이브러리를 통하여 단말기에서 동작되던 응용 프로그램이 WEBIO 라이브러리를 통하여 웹 브라우저 상에서 동작된다. 본 장에서는 응용 프로그램이 WEBIO 라이브러리 함수 webio_begin(), webio_printf(), webio_scanf() 및 webio_end()를 순서적으로 사용한다고 가정한다. 여기서 함수 webio_printf() 및 webio_scanf()는 STDIO 라이브러리의 함수 printf() 및 scanf()에 해당된다. 그리고 함수 webio_begin() 및 webio_end()는 본 장에서 기본 동작을 정의하기 위하여 표시하였지만 실제로는 컴파일 시 자동으로 삽입되므로 응용 프로그램에서 넣을 필요는 없다. 그림 2는 4개의 기본 동작을 그림으로 표현한 것이다.

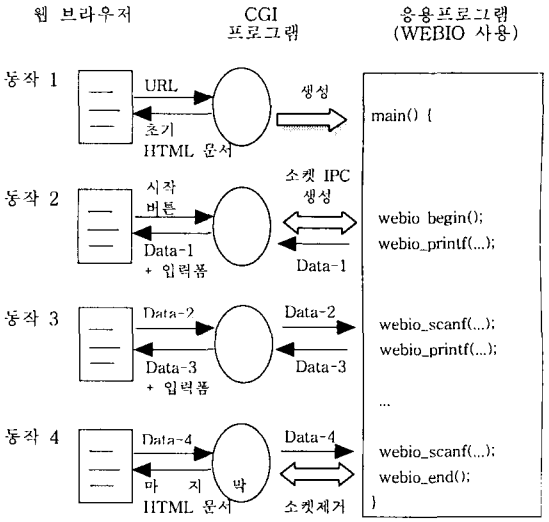


그림 2. 기본 동작
Fig. 2 Basic Operations

동작 1은 사용자가 웹 브라우저에서 응용 프로그램을 수행시키기 위하여 URL을 입력할 때 한번 수행된다. 이때 HTTP 서버는 환경 변수 QUERY_STRING에 NULL 값을 넣고 CGI 프로그램을 수행시킨다. CGI 프로그램은 QUERY_ STRING의 값이 NULL이기 때문에 응용 프로그램을 시작시키고 HTTP 서버에게 HTML 문서를 만들어 전달한다. 이때 만들어진 HTML 문서는 웹 브라우저 상에 응용 프로그램의 수행의 시작을 표시하는 START 버튼을 표시한다. 동작 1에서는 CGI 프로그램과 응용 프로그램 사이의 특별한 데이터 전달은 없다.

동작 2는 사용자가 웹 브라우저에 나타난 첫 페이지의 START 버튼을 누르면 수행된다. 이 동작의 처음에는 응용 프로그램이 함수 webio_ begin()을 수행하면서 IPC용 소켓을 생성하고 CGI 프로그램은 생성된 소켓에 연결한다. 응용 프로그램과 CGI 프로그램 사이의 소켓이 연결되면 응용 프로그램은 실행을 계속하여 함수 webio_printf()를 수행한다. 이때 Data-1이 소켓을 통하여 응용 프로그램에서 CGI 프로그램으로 전달되며 CGI 프로그램은 Data-1과 입력 폼이 포함된 HTML 문서를 생성한다. 생성된

HTML 문서는 HTTP 서버를 경유하여 웹 브라우저로 전달되고 이때 웹 브라우저 상에는 Data-1과 입력 폼이 출력된다.

동작 3은 사용자가 웹 브라우저 상의 입력 폼에 Data-2를 입력하면 시작된다. 이 데이터는 HTTP 서버에게 전달되며 HTTP 서버는 CGI 프로그램을 수행시켜 이를 전달한다. 생성된 CGI 프로그램은 Data-2를 소켓을 통하여 응용 프로그램에게 전달한다. 이때 응용 프로그램의 함수 webio_scanf()가 수행되면서 Data-2가 응용 프로그램의 주어진 변수에 할당된다. 응용 프로그램의 수행은 계속되고 함수 webio_printf()의 수행으로 Data-3이 CGI 프로그램을 거쳐 웹 브라우저로 전달된다. 동작 3은 응용 프로그램 수행 도중 여러 번 동작될 수 있다.

동작 4는 응용 프로그램에서 마지막 webio_scanf()를 수행하면 동작된다. 웹 브라우저에서 전달된 Data-4는 응용 프로그램으로 전달되고 응용 프로그램은 함수 webio_end()를 수행하면서 생성된 소켓을 제거한다. 이때 CGI 프로그램은 응용 프로그램의 종료를 알리는 HTML 문서를 웹 브라우저로 전송한다.

본 라이브러리의 주요 특징은 다음과 같다. 첫째, 기존의 터미널 상에서 수행되던 응용 프로그램을 수정하지 않고 인터넷에 연결된 웹 브라우저를 통하여 수행시킬 수 있다. 즉 기존의 비 통신 프로그램을 클라이언트 프로그램인 웹 브라우저와 통신하는 서버 프로그램으로 동작시킬 수 있다. 둘째, 웹 브라우저를 통하여 서비스되는 서버 프로그램을 터미널을 통하여 수행되는 응용 프로그램처럼 쉽게 프로그램 할 수 있게 한다. 즉 클라이언트 서버 통신 프로그래밍에 익숙하지 않은 프로그래머도 쉽게 인터넷에 연결된 웹 브라우저를 통하여 입출력이 가능한 서버 프로그램을 작성할 수 있게 한다. 셋째 본 라이브러리는 상태 유지 기능이 없는(stateless) 기존의 CGI 라이브러리와는 달리 응용 프로그램의 상태를 유지하면서 수행시킬 수 있다.

## IV. 시스템 설계

본 장에서는 WEBIO 라이브러리를 구성하는 프로그램인 CGI 프로그램과 주요 WEBIO 라이브러리 함수의 설계에 대하여 기술한다.

### 4.1 CGI 프로그램

CGI 프로그램은 HTTP 서버에 의하여 시작되어 HTTP 서버와 응용 프로그램 사이의 데이터 전달 역할을 한다. 또한 CGI 프로그램은 초기에 응용 프로그램을 시작시키기도 한다. 앞장에서 기술한 바와 같이 응용 프로그램 수행 도중 서로 다른 4 개의 동작이 있지만 이는 하나의 CGI 프로그램에서 처리한다. CGI 프로그램은 환경변수 QUERY_STRING의 값을 보고 수행할 동작을 알아낸다. 응용 프로그램을 작성하는 프로그래머는 이 CGI 프로그램을 프로그램할 필요는 없다. 아래 프로그램은 C 언어와 유사한 언어를 사용하여 CGI 프로그램의 설계를 기술한 것이다. 아래 프로그램에서 CGI 프로그램과 응용 프로그램 사이의 데이터 전달을 위하여 소켓 라이브러리 함수인 socket(), connect(), read(), write() 및 close()가 사용되었다.

```
                        /* 동작 1*/
if(QUERY_STRING==NULL){
   system("응용 프로그램 이름 &");
   printf(HTML 해드);
   printf(START 버튼);
                        /* 동작 2 */
} else if(QUERY_STRING=="start"){
   sd = socket(AF_UNIX, ...);
   connect(sd, ...);
   read(sd, read_buffer, ...);
   close(sd);
   printf(HTML 해드);
   printf("%s", read_buffer);
   printf(입력 폼을 위한 HTML);
                        /* 동작 3 및 4 */
} else {
   strcpy(QUERY_STRING, write_buffer);
   sd = socket(AF_UNIX, ...);
   connect(sd, ...);
   write(sd, write_buffer, ...);
```

```
    read(sd, read_buffer, ...);
    close(sd);
                        /* 동작 4 */
    if(read_buffer=="quit"){
      printf(HTML 헤드);
      printf(응용 프로그램의 끝을 알리는
        HTML);
                        /* 동작 3 */
    } else {
      printf(HTML 헤드);
      printf("%s", read_buffer);
      printf(입력 폼을 위한 HTML);
    }
  }
```

### 4.2 주요 WEBIO 함수

WEBIO 함수는 C 및 Prolog 언어의 STDIO 라이브러리 함수에 대응되는 함수이다. 본 절에서는 C 언어의 주요 WEBIO 함수인 webio_begin(), webio_printf(), webio_scanf() 및 webio_end()를 가상적인 C 언어를 사용하여 표현하였다. 함수 webio_begin()은 CGI 프로그램과 응용 프로그램 사이의 데이터 전달에 필요한 소켓을 생성하는 일을 하고 함수 webio_printf()는 인수에서 주어진 데이터를 쓰기 버퍼인 write_buffer[]에 추가하는 일을 하며 함수 webio_scanf()는 write_buffer[]에 저장된 데이터를 소켓을 통하여 CGI 프로그램으로 보낸 후 웹 브라우저에서 전달된 입력 데이터를 소켓에서 읽어서 넘긴다. 함수 webio_end()는 프로그램 종료를 알리는 메시지를 소켓을 통하여 보낸 후 소켓을 제거한다.

```
int webio_begin(void){
  s = socket(AF_UNIX, ...);
  bind(s, ...);
  listen(s, ...);
  sd = accept(s, ...);
}

int webio_printf(char *format, args){
  n = sprintf(buffer, format, args);
  strcat(write_buffer, buffer);
  return n;
}

int webio_scanf(char *format, args){
  write(sd, write_buffer, ...);
  close(sd);
  sd = accept(s, ...);
  read(sd, read_buffer, ...);
  n = sscanf(read_buffer, format, args);
  return n;
}

int webio_end(void){
  write(sd, write_buffer, ...);
  close(sd);
}
```

## V. C 및 Prolog 언어용 구현

앞장에서 기술된 설계는 거의 모든 언어에 적용 가능하다. 본 연구에서는 Solaris 2.6 운영체제를 사용하는 Sun Ultra5에서C 및 Prolog 언어용 WEBIO 라이브러리를 먼저 구현하였다. CGI 프로그램은 효율적인 수행을 위하여 두 언어 모두 C 언어로 구현하였다. WEBIO 라이브러리 함수는 C 언어용은 C 언어로 구현되었고 Prolog 언어용은 Prolog 언어로 구현되었다. C 언어용 WEBIO 함수는 대응되는 C 언어용 STDIO 함수와 프로토타입 및 사용법이 동일하며 Prolog 언어용 WEBIO 함수는 WEBIO 모듈에 구현되어 있다.

## VI. 라이브러리의 장점

본 연구에서 개발한 WEBIO 라이브러리는 다음과 같은 장점이 있다. 첫째 WEBIO 라이브러리는 기존의 CGI 프로그램 개발용 라이브러리와는 달리 응용 프로그램의 상태를 유지하면서 웹 브라우저 상에서 수행시키는 가능을 제공한다. 둘째 개발된 라이브러

리의 함수는 기존 C 및 Prolog 언어의 STDIO 라이브러리 함수와 사용 방법이 같기 때문에 특별히 사용 방법을 익히지 않고 사용할 수 있다. 셋째 기존의 단말기에서 수행되던 C 및 Prolog 프로그램의 소스를 수정하지 않고 다시 컴파일 하여 본 라이브러리와 링크 함으로서 인터넷에 연결된 웹 브라우저 상에서 수행시킬 수 있다. 넷째 본 라이브러리는 다중 클라이언트 기능을 가지고 있어 하나의 응용 프로그램을 여러 곳에서 다른 사람이 동시에 웹 브라우저 상에서 수행시킬 수 있다.

## Ⅶ. 결 론

본 논문은 STDIO 라이브러리를 사용하여 개발되어 단말기 상에서 수행되던 기존의 응용 프로그램을 수정하지 않고 인터넷에 연결된 웹 브라우저 상에서 수행시켜주는 WEBIO 라이브러리 프로그램에 대한 연구이다. 본 라이브러리는 기존의 CGI 프로그램 개발용 라이브러리와는 달리 HTTP 프로토콜을 통하여 응용 프로그램의 상태를 유지하며 수행시키는 기능을 가지는 최초의 라이브러리이다. 본 라이브러리는 STDIO 라이브러리 함수에 대응하는 WEBIO 라이브러리 함수와 응용 프로그램과 HTTP 서버와의 통신을 위한 CGI 프로그램으로 구성된다. 현재 C 언어 및 Prolog 언어용 라이브러리가 개발되어 시험 중이다.

## 참 고 문 헌


[1] T. Boutell, cgic 1.07, http://www.boutell.com/cgic/.

[2] D. Cabeza, M. Hermenegildo and S. Varma, The PiLLoW/CIAO Library for Internet/WWW Programming Using Computational Logic Systems, Procee-dings of the 1st Workshop on Logic Programming Tools for Internet Applications, JICSLP '96, 43-62, Bonn, September 1996.

[3] W. F. Clocksin and C. C. Mellish, Programming in Prolog, Fourth Edition, Springer-Verlag, Berlin, 1994.

[4] International Organization for Standardi-zation, Programming Languages - C, ISO/IEC 9899, 1990.

[5] ISO, ISO/IEC 13211-1 Programming Language Prolog, 1995.

[6] B. Kendrick, cgi-util 2.1.3, http://www.newbreedsoftware.com/cgi-util/.

[7] LPA Home Page, http://www.lpasystems.com/.

[8] NCSA, The WWW Common Gateway Interface 1.1, http://web.golux.com/coar/cgi/, 1997.

[9] P. J. Plauger, The Standard C Library, Prentice Hall, 1992.

[10] A. Sehmi and M. Kroening, WebLS: A Custom Prolog Rule Engine for Providing Web-based Tech Support, Proceedings of the 1st Workshop on Logic Programming Tools for Internet Applications, JICSLP '96, 107-123, Bonn, September 1996.

[11] W. R. Stevens, UNIX Network Programming, Networking APIs: Sockets and XTI, Vol. 1, Second Edition, Prentice-Hall Inc., 1998.

[12] B. W. Kernighan and D. M. Ritchie, The C Programming Language, Second Edition, Prentice Hall, 1988.

[13] World Wide Web Consortium, HTML 4.0.1 Specification, http://www.w3.org/TR/1999/REC-html401-19991224/, 1999.

[14] World Wide Web Consortium, Hypertext Transfer Protocol 1.1, http://www.w3.org/Protocols/HTTP/1.1/rfc2616.pdf, 1999.

## 저 자 소 개


신동하(Dongha Shin)

1980년 경북대학교 전자공학과 전자계산기모듈 졸업(학사)
1982년 서울대학교 대학원 전자계산기공학과 졸업(석사)
1994년 (미) University of South Carolina at Columbia 컴퓨터과학과 졸업(박사)
1982년~1996년 한국전자통신연구원(ETRI) 근무
1997년~현재 상명대학교 자연과학대학 소프트웨어학부 교수
※관심분야 : 프로그래밍 언어, 컴파일러, 인터넷 프로그래밍 도구, 내장형 시스템