2000년 한국정보처리학회 추계 학술발표논문집 제7권 제2호

호스트와 웹 환경의 통합을 위한 미들웨어의 설계

박나연, 정강용, 김원중
순천대학교 컴퓨터과학과
e-mail:{pny,jgy,kwj}@cs.sunchon.ac.kr

The Design of Middleware for Host&Web Computing Environment Integration

Na-Yeon Park, Gang-Yong Jung, Won-Jung Kim
Dept of Computer Science, Sunchon National University

요 약

현재의 정보처리시스템은 대형 컴퓨터 중심의 중앙 집중형 처리 환경, 클라이언트/서버의 분산처리시스템에서 웹 환경의 인트라넷 환경으로 빠르게 변화하고 있다. 인트라넷 환경은 웹 브라우저라는 사용자 중심의 통합된 환경, TCP/IP, HTTP, HTML, XML 등의 표준화 된 기술을 사용한 플랫폼 독립적인 사용환경, 비용의 효율성, 확장 및 유지보수의 용이성 등을 제공한다. 그러나 현재 사용중인 호스트 환경을 완전히 무시하고, 인트라넷 환경을 구축한다는 것은 매우 비현실적이다. 따라서 기존 환경은 그대로 유지하면서 인트라넷을 구축하기 위한 방법이 필요하다. 본 논문에서는 사용중인 기존 시스템은 전혀 변경하지 않고 호스트의 터미널 환경과 웹 브라우저 환경을 통합하는데 필요한 기능함수들을 정의하고, 이들을 이용하여 구현되는 호스트/웹 통합 미들웨어 시스템을 설계하였다.

1. 서론

정보처리시스템은 대형 컴퓨터 중심의 중앙 집중형 처리 환경, 클라이언트/서버 환경의 분산처리시스템으로, 그리고 지금의 인트라넷 환경으로 발전하여 왔다. 중앙집중형 시스템은 호스트 시스템의 과중한 부하, 느린 처리속도, 폐쇄적 아키텍처, 신뢰성, 업그레이드 비용 및 유지보수 비용, 환경의 변화에 대한 신속한 대응, 다양한 응용프로그램 개발 등에 있어서 많은 문제점을 가지고 있지만, 통합적인 업무 환경의 제공, 정보의 효율적 관리, 시스템 운영의 안정성, 보안성 등에서는 매우 뛰어나다.

중소형 컴퓨터와 네트워크를 기반으로 하는 클라이언트/서버의 분산처리 환경은 정보이용의 효율성, 자료의 무결성, 네트워크의 안정성 및 처리속도, 수 많은 클라이언트 프로그램 관리 등의 문제를 가지고 있으나, 화려한 그래픽 환경의 사용자 인터페이스(GUI), 사용의 편의성, 오픈 아키텍처, 다양한 툴들의 통합, 초기 투자비용의 절감과 같은 많은 장점을 가지고 있다.

인트라넷은 중앙집중형 시스템과 클라이언트/서버 환경의 분산시스템이 지니고 있는 장점들을 수용한 정보시스템 환경이다. 즉, 웹 브라우저라는 사용자 중심의 통합된 환경, TCP/IP, HTTP, HTML, XML 등의 표준화된 기술을 사용한 플랫폼 독립적인 사용환경, 비용의 효율성, 확장 및 유지보수의 용이성 등을 제공한다. 따라서 효율적인 인트라넷 구축, 즉, 호스트와 웹 환경의 통합에 대한 전략적인 연구와 접근이 필요하다[2,3,4].

2. 연구의 필요성

처음부터 인트라넷을 이용하여 정보시스템을 구축하면 별 문제가 없으나, 수년 동안 사용된 메인프레임 중심의 정보처리 환경을 인트라넷 환경으로 바꾸기 위해서는 기존에 개발된 모든 어플리케이션을 새로 개발하여야 하는 문제에 봉착하게 된다. 따라서, 중앙집중형 환경에서 개발된 모든 어플리케이션을 전혀 변경하지 않으면서, 기존의 호스트 터미널 사용자들에게 웹 브라우저와 같은 편리한 인터페이스를 제공하여야 한다.

또한, 기존 호스트와 터미널 사용자 사이에는 전용선으로 연결하여 각자 독자적인 통신 프로토콜을 사용하였기 때문에 많은 유지비용이 소요되었다. 그러나 호스트의 정보처리환경을 전혀 변경하지 않고, 호스트 터미널의 정보내용을 HTML 환경의 웹 페이지로 변환하면 전용선 대신 TCP/IP 프로토콜의 인터넷망을 통하여 소요비용을 대폭 절감 할 수 있다. [표 1]은 호스트의 터미널과 인터넷의 웹 브라우저 사용환경을

비교한 것이다[1,5,6].

	Terminal	Web Browser
사용자인터페이스	사용 호스트에 따라 각기 다름	웹 브라우저로 통일
정보검색	Host의 Application에 의존 (Function Key 등 사용)	하이퍼 링크로 편리함
프로토콜	Host에 따라 다름	TCP/IP, HTML로 통일
연결성	세션에 의해 연결성 유지	동적임 즉, 연결성 지원 못함
정보재검색	Host와 지속적인 상호 동기화	로컬캐싱, 정보검색기능 (앞/뒤 버튼 등 사용)

[표 1] Host Terminal 과 Web Browser의 차이점

3. 통합 미들웨어

호스트 환경과 웹 환경 통합을 위한 미들웨어인 웹 어플리케이션 서버(Web Application Server)는 호스트, 터미널, 웹 서버, 웹 브라우저의 환경을 상호 연동시키는 기능을 수행하여야 한다. 이것은 크게 터미널 관련 기능, 호스트/웹 환경의 상호 변경 기능, 그리고 분산된 상호 이질적인 데이터베이스들을 상호 연결시키는 기능들을 포함한다. 예를 들어, 호스트의 표준 터미널 데이터를 HTML, 자바 코드로 변환하고, 웹 브라우저로 정보를 전송할 수 있는 새로운 웹 어플리케이션을 생성하거나 웹 브라우저에서 사용자의 입력이나 어플리케이션 수행요구를 호스트에서 처리할 수 있도록 표준 터미널의 형태로 변환하는 작업을 수행한다. 본 논문에서는 이러한 환경의 변환에 필요한 기능함수들과 시스템의 구조에 대해 연구하였다.

3.1 터미널 관련 기능함수

터미널의 텍스트 화면을 HTML문서로 변환하는데 기본적인 기능은 접속, 로그인, 요청, 응답 접수, 로그아웃, 접속 종료의 여섯 단계를 가진다

접속 기능은 터미널에 접속하여 사용자에게 해당 어플리케이션의 사용자 인증 화면을 보여주는 단계까지이며, 접속 종료 기능은 사용중인 어플리케이션을 종료한 후 터미널의 물리적 및 논리적인 접속상태를 단절시키는 기능이다. 이를 위해서는 Connect, Disconnect, Checkfor, WaitFor, GetSessioninfo, GetScreenSize, KeyLocked, PressKey, SetCursor, GetCursor, Type, ReadScreen, GetAttribute 등이 필요하며, 각각의 기능은 다음과 같다.

Connect는 해당 호스트로의 물리적인 연결 및 논리적인 접속을 위한 것으로 매개변수는 접속 호스트의 IP Address, 접속 프로토콜, 접속 시도 후 응답 대기 시간, 최대 접속 회수이다. 연결이 성공하는 "0"을 리턴하고, 실패하면 해당 에러코드를 리턴 한다

Disconnect는 해당 호스트의 물리적인 연결 및 논리적인 연결을 위한 것으로 매개변수는 접속 호스트의 IP Address와 접속 프로토콜, 최대 접속 종료 시도 회수 등이다. 정상적인 접속 종료가 이루어진 경우 "0"을 리턴하고, 접속 종료가 실패한 경우 "1"을 리턴 한다

Checkfor는 호스트로의 접근이 가능한지를 체크하며, 매개변수로는 접속 호스트의 IP Address와 접속 프로토콜의 종류 등이다 정상적인 접근이 가능한 경우 "0"을 리턴하고, 그 밖의 경우는 해당 에러 코드를 리턴 한다

GetSessionInfo는 현재 접속되어 있는 상태를 알아보기 위해 사용되는 것으로 매개변수는 없고, 반환하는 값은 배열로서 접속된 호스트의 IP Address, 접속된 호스트의 도메인 이름, 프로토콜의 종류, 호스트 접근시 필요한 아이디, 호스트 접근시 필요한 패스워드, 세션 아이디, Sequence 번호이다.

GetReadScreen은 터미널에 접속한 후 해당 터미널의 기본 사이즈를 산출하는 경우로서 IBM3270 터미널인 경우 80×25가 기본 크기이므로 반환되는 값은 2000이다. 기본 사이즈를 산출하지 못하는 경우 "0"을 리턴 한다

KeyLocked는 접속중인 터미널 환경에서 키보드 입력을 가능하게 하거나 불가능하게 하는 기능을 수행한다. 매개변수는 "0"과 "1"이 사용되면 "0"인 경우 키보드 입력을 불가능하게 하며, "1"인 경우 키보드 입력을 가능하게 한다. 정상적인 경우 "0"을 반환하고, 비정상적인 수행 결과인 경우 "1"을 반환한다.

PressKey는 접속되어 있는 터미널 환경에 특수한 기능의 키 값에 해당하는 정보를 입력할 수 있도록 하는 기능을 제공한다. 매개변수는 터미널의 특수한 기능인 "Enter" 등의 기능을 지원하기 위해 예약된 키 값을 사용한다

SetCursor는 프롬프트의 위치를 지정하는 기능을 가진다. 매개변수로는 행과 열의 값을 가진다. 터미널의 기본 사이즈를 초과하는 경우 "1"을 반환한다.

GetCursor는 현재 프롬프트의 위치를 반환 받는다. 매개변수는 없고, 반환되는 값은 배열로서 첫 번째가 행, 두 번째가 열의 값이다. 커서의 위치를 찾지 못한 경우는 행과 열의 값에 "-1"을 반환한다.

Type는 사용자가 직접 값을 입력하는 것처럼 하는 기능을 제공한다. 매개변수는 ""로 이루어지는 문자열이며, 해당 문자열을 터미널의 프롬프트가 위치하는 곳에 타이핑하는 기능을 제공한다. 정상적인 수행 결과일 때는 "0", 비정상적인 수행결과일 때는 "1"을 반환한다.

WaitFor는 터미널에서 어떤 특정한 어플리케이션을 실행한 후 기다리는 대기 시간을 지정하는 것으로서 매개변수로는 일반 숫자를 사용하며, 단위는 초 단위이며 최소 값은 "0", 최대 값은 "3600"이다

ReadScreen은 시작위치의 좌표로부터 종료위치의 좌표까지 터미널의 화면 데이터를 가져온다. 매개변수로는 시작 행 위치, 시작 열 위치, 종료 행 위치, 종료 열 위치이다. 정상적인 결과인 경우 읽어온 화면을 반환하며, 비정상적인 경우 해당되는 에러 코드를 리턴 한다.

GetAttribute는 ReadScreen에서 읽어 온 화면 데이터에서 행, 열의 위치와 특성을 찾아서 Data Buffer Pool에 저장하는 기능을 수행한다. 매개변수는 ReadScreen에서 읽어온 화면 데이터이며 정상적으로 수행된 경우 "0"을 반환하며, 비정상적으로 수행된 경우 "1"을 반환한다.

3.2 호스트/웹 상호 변경 기능함수

터미널에서 읽어온 데이터를 웹 형태의 데이터를 변환하거나, 웹 형태의 CGI 데이터를 터미널 환경으로 변환하기 위한 함수로는 ConvertCGIEscapeChars, CreateCGIChars, SetCGIData GetCGIData CreateTerminalStream, createHTMLKeyPad 등이 있다.

ConvertCGIEscapeChars는 CGI 데이터 중 제어문자로 사용되는 &,%,+,=,?,;,:/ 등과 한글을 별도의 16진수로 되어 있는 문자로 디코딩하는 기능을 제공한다. 매개변수는 CGI 데이터의 입력 값이다. 반환하는 값은 16진수를 디코딩한 제어문자 및 한글이 CGI 데이터에 포함되어 반환되며, 에러가 발생한 경우 해당 에러코드를 반환한다.

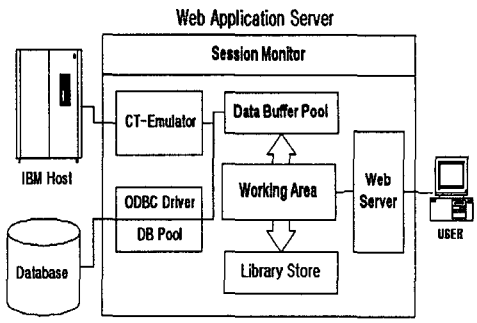
CreateTerminalStream은 Data Buffer Pool에 저장되어 있는 CGI 데이터 및 터미널 데이터 정보를 조합하여 터미널에서 사용되는 형태의 화면 데이터로 변경시키는 기능을 제공한다. 매개변수로는 터미널 타입, Data Buffer Pool 내의 화면 데이터를 가진 변수 이름, GetCGIData를 통해

서 넘어온 CGI Data값이다. 정상적인 처리가 수행된 경우 "0"을 반환하고, 실패한 경우 해당된 에러코드 값을 반환한다.
CreateCGIChars는 CreateHTMLForm이 Data Buffer Pool에 저장되어 있는 터미널 정보를 이용하여 생성한 HTML문에 SetCGIData가 생성한 Form문을 추가하는 기능을 제공한다. 사용되는 입력변수로는 CreateHTMLForm의 결과 값이며 반환되는 값은 브라우저가 해석할 수 있는 HTML문이다. 비정상적인 경우 "-1"을 반환한다.
CreateHTMLForm은 Data Buffer Pool에 저장되어 있는 터미널 화면을 Attribute의 특성을 살려 HTML문서로 생성시키는 기능을 제공한다. 입력 변수는 터미널의 화면 데이터이며, 반환되는 값은 생성된 HTML문서이다
CreateHTMLKeyPad는 웹 상에서 사용하지 않는 터미널의 특수한 키들을 위한 것으로서 HTML의 버튼형태로 키패드를 제공한다. 매개변수로는 터미널의 타입이다. 반환되는 값은 폼 형태의 HTML문서이며, 비정상적인 경우 "1"을 반환한다.
GetCGIData는 브라우저에서 전송한 CGI 데이터를 접수하여 Data Buffer Pool에 저장하는 기능을 제공한다. 매개변수는 Session ID와 Sequence ID 이다. 정상적으로 데이터를 접수한 경우 "0"을 반환하며, 비정상적인 경우 해당되는 에러코드를 반환한다.
SetCGIData는 생성된 HTML 문서를 Session ID와 Sequence ID를 포함시켜 브라우저에게 전송하는 기능을 한다. 정상적으로 데이터를 전송한 경우 "0"을 반환하고 비정상적인 경우 해당되는 에러코드를 반환한다.

3.3 데이터베이스 관련 기능함수
ODBC를 이용하여 관계형 데이터베이스에 접속하기 위해서는 접속, 질의문 수행, 결과 값 접수, 접속 종료 등의 단계를 거친다. 이러한 기능을 제공하기 위하여 사용되는 함수로는 BPDBDisconnect, BPDBConnect, BPDBExecuteSQL 등이 있다.
BPDBConnnect는 데이터베이스에 접속하기 위한 기능을 제공한다. 매개변수는 데이터베이스의 접속 IP Address나 도메인 이름, 사용자 ID, 패스워드가 사용된다. 정상적으로 접속이 된 경우 "0"을 반환하며, 비정상적인 경우 "1"을 반환한다.
BPDBDisconnect는 데이터베이스와 연결을 끊기 위한 기능을 제공한다. 매개변수는 없으며 정상적으로 종료가 된 경우 "0"을 반환한다.
BPDBExecuteSQL은 데이터베이스에 질의문을 전송하여 결과 값을 리턴 받는 기능을 제공한다. 매개변수는 ""로 정의된 SQL 질의문이며 정상적인 결과가 반환된 경우 "0"과 해당되는 값들이 배열 형태로 반환되며, 비정상적으로 결과가 수행된 경우 "1"을 반환한다.

3.4 기타
관계형 데이터베이스에서 추출한 데이터를 기존의 터미널 기반의 HTML 문서와 결합하거나 Data Buffer Pool에 저장되어 있는 화면 데이터를 변경하여 새로운 HTML 문서를 생성하기 위한 함수는 BPCompare, BPGetRow, BPInsertRow, BPJoin, BPGetColumn, BPDeleteColumn, BPInsertColumn, BPCopyTable 등이다. Data Buffer Pool에 저장되어 있는 화면 데이터는 테이블 구조 형태로 저장되어 있다.
BPCompare는 테이블의 행과 열을 비교하여 전자의 값이 큰 경우 양수, 같은 경우 "0", 후자의 값이 큰 경우 음수를 반환한다. 매개변수로는 Sequence ID, 행, 열, 사이즈이다
BPGetRow는 저장되어 있는 화면 데이터나 데이터베이스의 추출 결과에서 어떤 특정한 행을 추출할 때 사용된다. 매개변수로는 행이 사용되며, 값이 없는 경우 "-1"을 반환한다.
BPInsertRow는 한 행을 화면 데이터에 추가하는 기능이다. 사용되는 매개변수는 기준 행이며, 정상적인 결과가 수행된 경우 "0", 비정상적인 경우 "1"을 반환한다.
BPJoin은 두 개의 화면 데이터를 합칠 때 사용한다. 기준은 행이며 행 단위로 화면데이터를 연결한다. 매개변수로는 SequenceID가 사용된다. 정상적으로 수행된 경우 "0", 비정상적으로 수행된 경우 "1"을 반환한다.
BPGetColumn은 화면데이터에서 기준 열을 중심으로 크기만큼의 값을 가져오거나, 데이터베이스에서 추출한 결과 중에서 지정한 필드 값을 가져오는데 사용된다. 매개변수로는 기준 열 혹은 필드명과 사이즈가 사용된다
BPDeleteColumn은 화면 데이터에서 기준 열을 중심으로 크기만큼의 값을 삭제하거나 데이터베이스에서 추출한 결과 중에서 지정한 필드값 을 삭제하는데 사용한다. 매개변수로는 기준 열 또는 필드명과 사이즈가 사용된다. 정상적으로 수행된 경우 "0", 비정상적인 경우 "1"을 반환한다.
BPInsertColumn은 화면 데이터에서 기준 열을 중심으로 크기만큼의 값을 삭제하는데 사용한다. 매개변수로는 기준 열과 사이즈가 사용된다. 정상적으로 수행된 경우 "0", 비정상적으로 수행된 경우 "1"을 반환한다.
BPCopyTable은 화면 데이터를 복사하여 동일한 복사본을 생성하는 기능을 제공한다. 매개변수는 SequenceID가 사용되며, 정상적으로 수행이 된 경우 "0", 비정상적인 수행이면 "1"을 반환한다.

4. 시스템 구조
호스트와 웹 환경의 통합을 위한 미들웨어인 웹 어플리케이션 서버의 구조는 [그림1]과 같으며, 각 모듈들은 3절에서 정의된 기능함수들을 사용하여 각자의 기능을 수행한다.



[그림 1] 시스템 구조

4.1 Session Monitor
Session Monitor는 HTTP Protocol의 단점을 보완하기 위한 것으로서 사용자의 세션 정보를 유지함으로써 호스트 환경과 웹 환경을 연동시키는 기능을 수행한다. 호스트와 터미널 환경의 연결은 상태성, 연결성을 유지하는 특성을 가지고 있으나. 웹 환경의 HTTP는 비연결성, 비상태성을 갖는 프로토콜이다. 따라서 호스트와 웹 브라우저를 동기화 시키는것이 필요하다. 본 연구에서는 이를 위하여 각 화면에 대한 Session ID와 Sequence ID를 생성하여 호스

트 화면과 웹 브라우저를 화면을 서로 연결하도록 하였다.

4.2 CT-Emulator
호스트의 터미널은 지정된 프로토콜을 사용하여 호스트와 직접 데이터를 주고 받는다. CT-Emulator는 호스트의 터미널 화면에서의 각종 속성(attribute)들을 분석하여 속성 테이블과 더미(dummy)화면을 생성한다. 호스트의 터미널 화면에서 사용되는 속성들은 Alpha-Numeric, Auto-Skip, Hidden, Protected, Blink, Hidden, Intensified, Reverse, Underscore, Extended Color 등이 있다.

4.3 Data Buffer Pool
Data Buffer Pool은 사용자가 호스트의 터미널에 접속하는 경우나 데이터베이스를 접속하여 원하는 데이터를 가져오기 위하여 수행되는 일종의 배치(Batch)형 전처리기(preprocessor)들과 CT-Emulator와 DB Pools에서 가져온 데이터를 저장 관리하는 곳이다. 크게 Data Buffer Pool에 사용되는 데이터 유형은 ODBC 기반, 터미널 기반, 그리고 시스템 운영 데이터등 세 가지로 나눌 수 있다. 현재 본 연구에서는 관계형 데이터베이스만을 고려하고 있다.

4.4 Library Store
Library Store는 터미널 화면을 웹으로 바로 변경하거나, 변경한 화면을 사용자가 필요에 의해 수정할 수 있도록 하는 터미널 기반의 라이브러리와 관계형 데이터베이스의 데이터를 인출하여 처리할 수 있도록 한 ODBC 기반의 라이브러리, 또한 시스템 자체적인 인증 처리 및 시스템 운영을 위하여 필요한 자체 라이브러리와 Data Buffer Pool에 저장된 데이터를 조합하여 작업할 수 있도록 하는 라이브러리들로 규성되어 있다.

4.5 Working Area
Working Area는 시스템에서 제공하는 라이브러리들을 이용하여 사용자가 웹 브라우저를 통해 호스트의 터미널 화면의 데이터를 조작하거나, ODBC에 연결된 데이터베이스의 데이터를 처리할 수 있도록 정의된 어플리케이션들의 집합 장소이다. 이 정의된 어플리케이션들을 Information Rule이라고 한다. Information Rule들은 크게 Information Rule, Information Builder로 분류된다. Information Rule은 가장 기본적인 Application으로서 라이브러리를 이용해서 하나의 데이터에만 접근 가능하다. 하나의 데이터에만 접근이 가능하다는 것은 Information Rule은 동시에 호스트의 터미널 화면과 데이터베이스 양쪽에서 데이터를 가져오지 못하고, 한쪽에서만 데이터를 가져올 수 있다는 뜻이다. Information Rule에서 한 곳에서만 데이터를 가져올 수 있도록 한 것은 사용자의 세션을 관리하기 쉽게 하기 위해서다. Information Builder는 Information Rule에서 생성된 결과를 조합하여 가공할 수 있다. Information Builder 자체가 ODBC나 터미널 기반의 데이터 소스에 접근할 수는 없지만, Information Rule들의 결합으로 ODBC 기반의 데이터나 터미널 기반의 데이터에 접근이 가능하다. Working Area는 Information Rule이나 Information Rule들의 집합체인 Information Builder들의 Application들이 카트리지처럼 대기하고 있다가 사용자가 해당 정보를 요구하게 되면 결과를 전송하도록 한다.

4.6 DB Pools
ODBC 드라이버를 통하여 데이터베이스에 접속한 후 시스템에서 필요한 데이터를 읽어올 때 일차적으로 저장되는 곳은 DB Pools이다. DB Pools는 SQL Buffer Pool과 Query Analyzer 두 가지로 구성된다. ODBC를 이용하여 데이터를 가져오기 위해서는 접속 시도 → 접속 → 질의문 전송 → 결과 값 반환 → 접속 종료의 과정을 거친다. 이러한 과정을 질의가 있을 때마다 반복하면 동일한 질의에 대한 동일한 결과문을 여러 번 생성하게 되어 비효율적이다. Buffer Pool은 넘어온 질의문의 결과를 보관하고 있다가 동일한 질의에 대해서는 데이터베이스에 재접속하지 않고 바로 데이터를 넘겨줌으로써 불필요한 데이터베이스 접속을 줄여 전체적인 시스템의 성능을 시킬 수 있다. SQL Buffer Pool은 LRU(Least Recently Used) 알고리즘을 이용하여 관리한다.

5. 결론
본 연구에서는 호스트와 웹 환경의 통합을 위한 미들웨어인 웹 어플리케이션 서버를 구축하기 위한 기능함수들을 정의하고, 이들 함수들을 이용하여 각각의 기능을 수행하는 시스템 구성요소들에 대해 설명하였다. 시스템의 구축에는 기존의 CGI 방식이 아닌 웹 어플리케이션 전용의 API를 사용하고 있다. 이것은 CGI는 클라이언트의 호출 때마다 프로세스가 생성되어 시스템의 성능을 저하시키기 때문이다. 시스템은 호스트로는 IBM OS/390, 웹 어플리케이션 서버의 구축을 위해서는 PentiumII 450Mhz의 Windows NT 환경, 그리고 Visual C++ 5.x를 사용하고 있다.

참고문헌
[1] A. Luotonen and T. Berners-Lee, CERN httpd Refenence Manual - A Guide to a World-Wide Web Hypertext Daemon, CERN, May, 1994.
[2] D.M Kristol, Proposed HTTP State Managememt Mechanism, Internet Draft, Jun, 1996.
[3] D. Robinson, The WWW Common Gateway Interface Version1.1, Internet Draft, Jan, 1996.
[4] Douglas Comer, The Internet Book, Prentice-Hall, 1995.
[5] I.S. GRAHAM, The HTTP Protocol and Common Gateway Interface, HTML SOURCE BOOK, pp. 181-230, 1995.
[6] P.C Kim, "A Taxonomy on the Architecture of Database Gateways for the Web," Proc. of the 13th ICAST97, Schaumburg, IL, pp. 226-232, April. 1997.