웹 데이터베이스 연결도구의 비교 분석

박 성 헌* · 박 지 헌**

Comparative Analysis of Web Database Connectivity Tools

Sung-Hun Park* · Ji-Hun Park**

Abstract

Since web has been used as the front-end of database, many web database connectivity tools have been developed and being developed now. For web developers and educators, it has been a difficult problem to select one tool out of so many alternatives. This paper compares web database connectivity tools available on PC Windows environment in the viewpoint of developers and educators. This comparative analysis focuses on the functions and programming techniques provided by these tools through implementing a simple case study using these software solutions. The performance analysis of these tools was not done in this reason.

※ 본 연구는 2000년 정보통신부 우수대학원 지원사업에 의하여 이루어졌습니다.
* 명지대 지식정보학부
** 홍익대 컴퓨터공학과

1. 서 론

웹(Web) 기술은 현재 거의 모든 정보시스템을 변화시키고 있다. 인터넷 신문과 홈페이지에서 시작한 웹은 전자상거래[8] 등의 인터넷 분야뿐만이 아니라 전통적인 Client/Server[11] 환경에서만 구축되던 주식정보시스템, 학사정보시스템, 인사정보시스템, EIS(Executive Information System) 등과 같은 응용정보시스템 구축에도 활발하게 이용되고 있다. 이와 같은 웹 기반의 응용정보시스템 구축에서 제일 중요한 것이 데이터 베이스에 저장되어 있는 정보를 웹 브라우저를 통해 검색하고 추가/변경/삭제할 수 있도록 해 주는 웹 데이터 베이스 연결 도구들이다. 이들 웹 DB 연결 도구는 업체들에 의해 다양한 솔루션들이 개발되었고 지금도 새로운 도구들이 개발되고 있는 중이다.

기존의 연구는 연결 도구들의 구조의 차이에 대한 연구[4], 연결 도구들의 성능[14]에 대한 연구, 몇 개의 제한된 도구들만 비교한 연구들[9]이 진행되었다. 본 논문은 단편적으로 비교분석된 웹 데이터 베이스 연결 도구들을 간단한 주소록 관리 시스템의 구축 사례를 통하여 포괄적으로 비교해 보고자 한다. 비교 분석에 포함할 도구들에 어떤 도구들을 포함시키고 어떤 도구들을 제외할 것인가와 이들의 도구들의 비교분석은 어떤 관점에서 해야 하겠는가 어려운 문제이었다. 본 논문의 비교분석은 경영정보학을 전공하는 학부생을 대상으로 한 웹 데이터베이스 교육에 초점을 맞추었다. 그러므로 학생, 교수들이 보편적으로 많이 사용하는 PC Windows 환경에서 널리 사용되는 도구들을 선택하여 비교하였다. 개발 및 교육 관점에서의 비교 분석이므로 도구들의 처리 성능(performance)에 대한 비교는 의미가 없어서 제외하였고 주로 이들 도구들이 지원하는 기능과 프로그래밍 방법에 초점을 맞추어 분석하였다. 같은 이유로 비교분석에 포함한 연결 도구는 지적재산권이 문제가 될 수 있는 상용 소프트웨어를 배제하고 공개 소프트웨어나 거의 공짜로 구할 수 있는 소프트웨어들만 포함하였다.

사례 연구에서 사용하는 데이터 베이스 시스템은 PC Windows에서 쉽게 구할 수 있는 MS Access를 선택하였고 웹 서버는 Apache 서버와 Apache Tomcat, MS사의 PWS(Personal Web Server)와 IIS(Internet Information Server) 그리고 웹 프로그래밍 환경으로 Perl, PHP, ASP(Active Server Page), Java Servlet, JSP (Java Server Page)를 선택하였다. 다만 상용이지만 독특한 스크립트 기능을 최초로 제공한 Allaire사의 Cold Fusion은 다른 도구들 설계에 큰 영향을 미쳤으므로 이를 비교에 포함하였다. 그렇지만 특정 웹 서버에만 작동하고 상용인 연결도구인 Oracle Application Server와 Netscape Enterprise Server는 본 논문의 비교분석에 포함하지 않았다.

본 논문의 결과는 웹 데이터 베이스 연결 도구를 교육해야 하는 전산학과나 경영정보학 관련 분야에 종사하는 교육자들에게 길라잡이로 활용될 수 있으며 자택의 PC를 이용하여 개발한 후 작업 결과를 회사 서버 환경에 이식하는 개발자들에게도 의미 있는 자료로 활용될 수 있겠다.

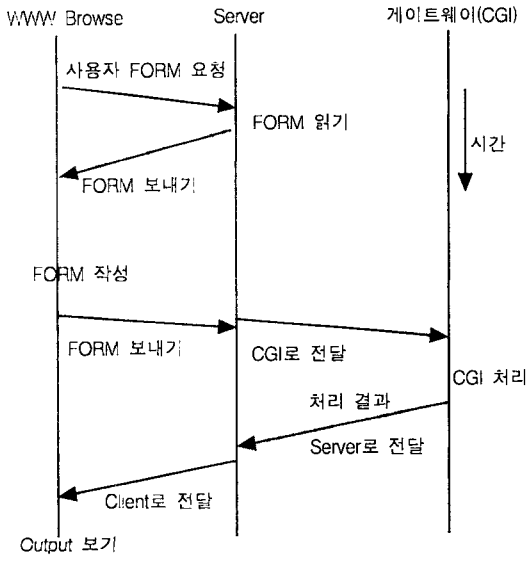
2. 웹과 데이터베이스의 연결

웹 브라우저는 인터넷의 여러 가지 정보 서비스를 처리할 수 있지만 모든 종류의 서비스를 처리하지는 못 한다. 그리고 모든 정보들을 HTML 파일로 작성하여 디렉토리에 저장해 둘 수 없다. Access와 Oracle등의 데이터 베이스 엔진이 저장하고 있는 정보는 필요한 경우 HTML형태

로 실시간 생성하여야 한다.

CGI(Common Gateway Interface)[10]는 이와 같은 문제들을 해결하기 위해 웹 서버의 기능 확장을 목적으로 정의되었다. CGI방식을 통해 수행되는 게이트웨이 프로그램은 웹의 기본 틀에 맞지 않는 정보를 웹 서버의 파일처럼 인식되도록 해준다. 게이트웨이는 스크립트(script)나 프로그램으로 웹 서버를 통해 사용자의 입력을 받고 결과는 HTML, URL, 혹은 다른 형태로 출력한다. 웹 서버와 게이트웨이 프로그램은 CGI가 정한 방식대로 정보를 전달해 주고 처리한 결과를 받게 된다. 게이트웨이 프로그램은 C/C++, Perl, PHP, Java, tcl, C Shell, Bourne Shell등의 거의 모든 언어로 작성할 수 있다.

게이트웨이와 함께 많이 사용되는 것은 HTML 폼(Form)으로 사용자들로부터 데이터 수집을 담당한다. 웹 서버는 폼 입력을 스크립트나 프로그램에 전달하고, 게이트웨이는 이들 입력을 처리한 후, 처리 결과를 웹 서버로 돌려준다. 이와 같은 과정은 (그림 1)에서 보여 주는 단계로 진행된다.



(그림 1) CGI 게이트웨이 프로그램의 흐름

CGI는 환경 변수(Environment Variable), 표준 입력(Standard Input), 표준 출력(Standard Output)을 이용하여 간단하게 웹 서버와 게이트웨이간의 통신을 제공하는 장점이 있지만 이와 같은 간단한 방식 때문에 성능에선 약점을 가진다. 여러 가지 이유들 때문에 CGI는 그다지 효율적으로 실행되지 않는다. 우선 CGI 스크립트가 실행될 때마다 서버는 프로세스를 생성하고, 코드를 프로세스의 메모리 세그먼트로 로드하여 실행한 다음 프로세스를 종료시키고 관련된 메모리를 반환한다. 데스크탑 컴퓨터에서 hwp와 같은 응용 프로그램을 수행할 때 hwp를 메모리로 로드하는데 시간이 오래 걸리는 것과 같다. CGI 스크립트가 응용프로그램에 비해 훨씬 더 작기는 하지만, 로드/언로드에는 시간이 상당히 걸린다. 보다 중요한 사실은 CGI 스크립트가 겨우 1~2초 동안 실행되도록 개발되었기 때문에 로드/언로드 단계는 그들의 전체 실행 시간의 큰 부분을 차지한다는 것이다. 특히 게이트웨이 프로그램에서 데이터 베이스 엔진을 연결하여 검색을 하는 경우 속도가 더 느려지게 된다. 즉 게이트웨이가 수행될 때마다 데이터 베이스 엔진과의 연결을 다시 맺어야 하고 수행이 끝나면 연결을 삭제함으로써 게이트웨이 프로그램의 수행 속도가 더 느려지게 된다. 이 문제를 해결하기 위하여 Server API나 JDBC를 이용하여 데이터 베이스로의 연결을 계속 저장해 두는 캐싱(Caching) 방법[4, 14]들이 제안되었다.

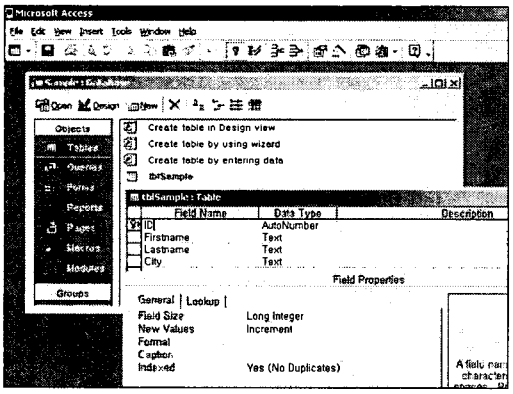
데이터 베이스 연결의 캐싱은 연결도구에서 지원하는 것으로 개발자는 이 기능을 직접 코딩하는 경우는 거의 없으며 웹 프로그램은 기본적으로 CGI 모델에 따라 개발하도록 되어 있다. 즉 사용자로부터 폼 입력을 환경 변수나 표준 입력을 통해서 받아 드리고 데이터 베이스 엔진에 연결을 만들고 적당한 SQL 명령어를 작성

하여 수행을 요청한 후 그 결과를 받아 HTML 형태로 표준 출력으로 프린트하는 그림 1의 구조를 따른다.

3. 비교분석

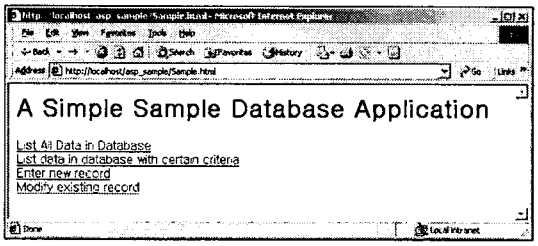
3.1 사례 연구

여러 연결 도구들을 비교 분석하기 위해서 간단한 주소록 관리 기능을 하는 웹 응용 사례를 설계하였다. 이 주소록 관리 프로그램은 주소 데이터베이스를 이용하여 주소의 추가, 변경, 검색 등의 기능을 제공한다. 주소록 관리 프로그램은 Microsoft Access로 만들어진 하나의 테이블 tblSample을 이용한다. tblSamle은 (그림 2)와 같이 ID, Firstname, Lastname, City의 4개의 필드를 갖는다. 이들 필드는 ID만 Access에서 자동으로 만들어지는 숫자이고 나머지 필드들은 문자열인 구조를 갖는다. 이 Access 데이터 베이스의 이름은 Sample.mdb로 정의하였고 ODBC(Open Data Base Connectivity) 드라이브에 소스 이름 Sample로 등록하였고 모든 웹 게이트웨이 프로그램들에서 DSN(Data Source Name) Sample을 이용하여 연결한다.



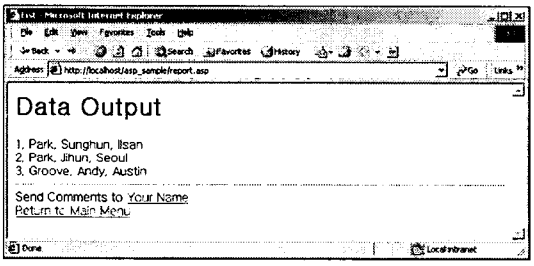
(그림 2) 데이터 소스 Sample.mdb의 구조

웹 프로그램은 (그림 3)과 같은 시작 페이지를 중심으로 구성되어 있다. 그림 3과 같은 시작 홈페이지(Sample.html)를 통해 주소록 전체 리스트(report 스크립트), 도시 이름으로 주소록 찾기(search.html과 query 스크립트), 새로운 주소 추가(add.html과 add 스크립트), 그리고 기존 주소 변경(modify, modify2, modify3 스크립트)을 지원하는 HTML 페이지나 관련 스크립트 프로그램으로 연결된다.



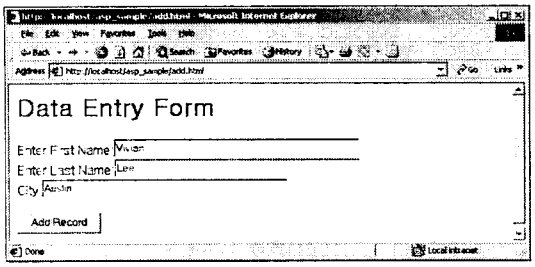
(그림 3) 주소록 사례의 시작 홈페이지

첫째 하이퍼링크(Hyperlink)를 선택한 경우는 tblSample 테이블의 내용을 모두 출력하는 스크립트가 수행되고 그 결과는 (그림 4)와 같다. 다만 수행되는 스크립트는 어떤 연결 도구로 구현되었는가에 따라 3.2절에서와 같이 여러 가지 방법으로 구현된다.



(그림 4) report 스크립트를 수행한 결과

(그림 5)는 주소록 추가를 선택하는 경우 입력 폼(Form)을 보여 준다. 필요한 데이터를 입력한 후 버튼을 선택하면 add 스크립트가 수행되며 이 스크립트는 입력된 데이터를 확인하고 새로운 주소를 tblSample 테이블에 추가한다.



(그림 5) add.html 주소록 입력 폼

다음절에서 여러 연결도구들의 비교 분석을 위해 report, add, query, modify, modify2, modify3 스크립트들 중에서 대표적인 기능들을 비교해 볼 수 있는 report와 add 스크립트만을 다룬다.

웹 데이터베이스 연결 도구 사용법에 대한 교육은 웹과 데이터베이스의 기본적인 교육에서 시작한다. 웹의 기본인 HTML 태그들에 대한 기본적인 교육이 선행되어야 한다. 입력 처리를 위해서는 HTML 폼(FORM)에 관련된 태그들과 테이블 형태의 검색 결과의 출력을 위해서는 TABLE 태그들에 대한 이해가 필수적이다. 데이터베이스 과목에서 소개된 데이터베이스의 보편언어인 SQL(Structured Query Language)에 대한 이해는 테이블들에 저장된 자료를 조작하는 데 필수적으로 필요하다.

모든 연결 도구의 교육에는 이와 같은 HTML과 SQL에 대한 교육이 선행되어야 한다. 여러 도구들의 사용법은 HTML FORM으로의 입력 처리, 데이터베이스와의 연결 방법, SQL 명령어 전달 방법, 데이터베이스 엔진으로부터의 수행 결과 처리 등에서 주된 차이가 난다. 본 비교 분석에서는 연결 도구의 사용 용이성과 기능성의 관점에서 다음과 같은 비교 척도를 사용하였다.

연결 도구가 지원하고 있는 프로그래밍 언어의 특징이다. 즉 C와 같은 프로시듀얼 언어(Procedural Language)인가? Visual Basic과 같은 객체 기반 언어(Object-based Language)인가? Java나 C++와 같은 객체 지향 언어(Object-oriented Language)인가? 연결 도구가 기초한 언어의 특징에 따라 새로운 기능을 추가하거나 기존 기능을 수정 보완할 수 있는 확장성에서 차이가 난다.

프로시듀얼 언어인 경우 새로운 기능은 모두 라이브러리로 구현하여야 하는데 반하여 객체 지향 언어는 제공되는 기존 객체 클래스를 일부 수정하여 새로운 객체를 구현할 수 있다. 다만 객체 지향 언어와 제공되는 기존 객체 클래스들에 대한 이해가 필수적이므로 객체 지향 언어의 학습이 프로시듀얼 언어에 비해 더 많은 노력과 시간을 필요로 한다. 객체 기반 언어는 이 두 언어의 중간 정도의 특징을 가지고 있다. 여러 객체를 제공하되 이들 객체를 이용하여 새로운 기능을 하는 객체를 정의하는 것이 객체 지향 언어에 비해 불편하다.

객체 언어는 확장성외 사용자 인터페이스와 데이터베이스를 분리 개발할 수 있는 분리성을 제공한다. 데이터베이스를 이용하는 비즈니스 로직을 객체에 감출 수 있고 정의된 getProperty혹은 setProperty 메소드를 통해 작동할 수 있으면 사용자 인터페이스와 비즈니스 로직은 서로 분리되어 개발될 수 있다. 즉 웹 디자인너는 HTML만을 이용하여 사용자 인터페이스를 설계하고 비즈니스 로직은 웹 프로그래머가 개발한 객체들의 getProperty와 setProperty 메소드만 이용하여 구현한다. 웹 프로그래머는 필요한 데이터베이스를 조작하고 복잡한 비즈니스 로직을 구현함으로써 웹 디자인너가 필요로 하는 비즈니스 로직 객체를 제공한다. 이와 같은 분리성은 대규모 웹 프로젝트 개발에 효과적으로 사용될 수 있는 기능이다.

또 다른 비교 척도는 Windows에 개발한 결과물을 Solaris, Linux등과 같은 다른 컴퓨터 환경에 얼마나 손쉽게 옮겨 갈 수 있는가 하는 이

식성이다. 연결 도구에 따라 전혀 수정 없이 바로 이식할 수 있는 도구들도 있는 반면에 Window 환경에서만 제한적으로 제공되는 도구도 있다.

연결 도구의 비교 분석에 사용하는 척도에 본 논문에서 사용한 기반 언어, 분리성, 이식성 이외에도 연결 도구의 데이터베이스 엔진과의 처리 능력 등을 포함할 수 있을 것이나 도구의 사용 용이성에 초점을 맞추기 위해 이를 제외하였다.

3.2 도구 비교 분석

3.2.1 ASP(Active Server Page)

ASP는 마이크로소프트사의 PWS(personal Web Server)나 IIS(Internet Information Server)에 내장되어 있는 베이직 인터프리터로 게이트웨이 프로그램이 필요로 하는 기능들을 ReQuest, Response, Server 등의 몇가지 객체로 제공한다. ASP 인터프리터는 데이터 베이스 연결을 캐싱함으로써 매번 연결을 새로 만들어야 하는 CGI 방법의 성능 문제를 개선하였다.

주소록 테이블의 내용을 모두 출력하는 report.asp 게이트웨이 스크립트의 코드는 다음과 같다.

```
<HTML><HEAD>
<TITLE>List</TITLE></HEAD><BODY>
<H1>Data Output</H1>
<%
'Create and Open Connection Object
Set OBJdbConnection = Server.CreateObject("ADODB.Connection")
strProvider="DSN=Sample;UID=;Passwd="
OBJdbConnection.Open strProvider
Set RsList = Server.CreateObject("ADODB.Recordset")
RsList.ActiveConnection = OBJdbConnection
RsList.Source = "select * from tblSample"
RsList.Open
%>
<%
   Do Until RsList.EOF
   NO = RSList("ID")
   A = RSList("Firstname")
   B = RSList("Lastname")
    C = RSList("City")
    Response.Write( No & ", " & B & ", " & A & ", " & C & "<br>")
   RsList.MoveNext
   Loop
%>
<HR>
Send Comments to <a href=mailto:yourmail@mail.utexas.edu>Your
Name</a><br>
<a href="Sample.html">Return to Main Menu</a>
</body></html>
```

report.asp 코드는 기본적으로 HTML로 만들어져 있으며 데이터 베이스와의 연결, 검색 결과를 처리 등을 위하여 Server 객체를 통해 Connection, Command, Recordset 등의 ADO (Active Data Object) 객체들을 이용할 수 있도록 설계되어 있다. 그리고 이들 객체 조작을 지원하기 위하여 <% 태그와 %> 태그 안에 베이직 명령어를 포함 할 수 있도록 설계되어 있다. Response 객체는 출력을 지원하고 있다.

ASP을 이용한 개발에는 HTML에 대한 기본적인 이해와 ADO 객체들을 Server.CreateObject 메소드를 이용해서 생성하고 검색에 필요한 SQL(Structured Query Language)를 작성할 줄 알며 적당한 베이직의 제어문을 이용하여 검색된 결과를 HTML로 변환하여 출력할 줄 아는 지식이 필요하다.

다음 add.asp에서는 폼으로부터의 입력을 처리하는 ReQuest 객체의 사용법을 보여준다. 폼은 두 가지 형태로 데이터를 서버에 전달할 수 있다. GET 방법과 POST방법이다. 두 가지 방법을 모두 지원하기 위해서 ASP에서는 GET 방법의 폼인 경우 ReQuest.QueryString 컬렉션(Collection)에 정보를 전달해 주고 POST 방법인 경우는 ReQuest.Form 컬렉션(Collection)에

정보를 담아 준다. 폼의 각각의 필드 이름이나 위치를 이용하여 해당되는 폼 필드의 데이터를 사용하면 된다. add.html에서는 POST 방법을 이용했으므로 add.asp에서는 ReQuest.Form("Firstname")를 이용하여 입력된 "Firstname" 텍스트 필드의 값을 가져왔다.

```
<HTML><BODY>
<%
'Create and Open Connection Object
Set OBJdbConnection = Server.CreateObject("ADODB. Connection")
strProvider="DSN=Sample;UID=;Passwd="
OBJdbConnection.Open strProvider
OBJdbConnection.Execute "Insert into tblSample(Firstname, Lastname,
                        City)" & _
    "values('"&Request.Form("FirstName")&"','"&Request.Form
                ("Lastname")&"','"& _
    Request.Form("City")&"');"
%>
Record has been added.<br>
Return to <a href='sample.html">Main Page</a>
</BODY></HTML>
```

ASP는 마이크로소프트사의 비주얼 베이직(Visual Basic) 프로그래머들이 웹 개발자로 손쉽게 적응할 수 있도록 설계되었다. 비주얼 베이직 프로그래머들은 프레임 버튼 텍스트 필드 등의 그래픽 객체 대신에 HTML/Form/Table에 대한 지식만 추가로 습득하면 손쉽게 ASP 개발이 가능하다. 이에 반해 Unix나 Linux 환경으로의 이식은 따로 마이크로소프트사가 아닌 ChiliSoft사와 같은 다른 업체로부터 상용 ASP를 구입하여야 한다. 마이크로소프트사는 Unix 운영체제와 경쟁하는 NT 운영체제를 지원하고 있어 Unix용 ASP를 제공할 계획이 현재는 없다. 그리고 ASP에서 제공하는 객체가 아닌 새로운 객체를 추가하여 기능을 확장하기 위해서는 ActiveX 프로그래밍 환경을 이해하는 많은 노력이 필요하다.

3.2.2 Cold Fusion

Allaire사에서 개발한 Cold Fusion[1]은 상용 소프트웨어이지만 독특한 프로그래밍 개념을 지원하고 있기 때문에 본 논문의 비교 분석에 포함하였다. Cold Fusion은 데이터 베이스 연결 도구로 거의 모든 웹 서버와 데이터 베이스 엔진과의 연결을 지원한다. 본 논문에서는 MS PWS에 Cold Fusion을 설치하여 테스트하였다.

Cold Fusion을 이용한 사례연구도 3.1절과 같은 구조를 갖는다. 즉 시작 홈페이지인 Sample.html, 새로운 주소록 입력을 제공하는 폼인 add.html, 검색을 지원하는 폼인 search.html 그리고 Cold Fusion으로 구현된 게이트웨이 프로그램들인 report.cfm, add.cfm, query.cfm, modify.cfm, modify2.cfm, modify3.cfm으로 이루어진다. 이들 중에서 report.cfm과 add.cfm의 코드에 대해 검토해 본다.

```
<CFQUERY NAME=listing DATASOURCE=Sample>
        select * from tblSample
</CFQUERY>
<HTML><BODY>
<H1>Data Output</H1>
<CFOUTPUT QUERY=listing>
            #ID#, #Lastname#, #Firstname#, #city# <BR>
</CFOUTPUT>
<HR>
Send Comments to <a href=mailto:yourmail@mail.utexas.edu>Your
                    Name</a>
<a href="Sample.html"><br>
Return to Main Menu</a>
</body></html>
```

위의 report.cfm에서는 HTML 태그들과 함께 몇 가지 Cold Fusion 태그가 사용되고 있다. 데이터 베이스의 검색을 하는 SQL 명령어를 지정하는 <CFQUERY> 태그와 그 결과를 지정하는 <CFOUTPUT>태그이다. 특히 <CFOUTPUT> 태그는 검색한 결과의 출력 형태를 각각

의 필드에 저장된 값을 #필드명# 형태로 지정해 주기만 하면 반복되는 레코드를 하나씩 Cold Fusion이 출력해 주는 기능을 제공한다. ASP와 같이 Recordset의 Cursor의 위치를 확인하여 Recordset의 끝을 만날 때까지 반복하여 출력해 주는 제어 문장이 필요가 없다.

다음 add.cfm은 한 걸음 더 나아가서 입력 폼의 필드 명과 테이블의 필드 명을 서로 같이 사용하면 SQL Insert 명령어가 <CFINSERT> 태그 하나로 대치될 수 있는 기능을 보여 준다.

```
<HTML><BODY>
<CFINSERT DATASOURCE=Sample TABLENAME=tblSample>
Record has been added.<br>
Return to <a href=sample.html>Main Page</a>
</BODY></HTML>
```

위의 add.cfm은 다음 코드를 간단하게 표현한 것이다. 즉 모든 폼 입력에 대해 Cold Fusion은 폼의 각각의 필드에 대해 #필드명# Cold Fusion 변수를 생성한다. 이들을 Cold Fusion 태그 내에서 #필드명#으로 불러서 사용할 수 있다. 다음 코드는 <CFQUERY>태그에서 SQL insert 문장을 이용하여 레코드의 추가를 하는 것을 보여 준다. '#필드명#' 의 ' '는 Firstname, Lastname, City 필드가 모두 문자열 타입이기 때문에 꼭 필요하다.

```
<HTML><BODY>
<CFQUERY DATASOURCE=Sample>
        insert into tblSample(Firstname, Lastname, City)
        values('#Firstname#', '#Lastname#', '#City#')
</CFQUERY>
Record has been added.<br>
Return to <a href=sample.html>Main Page</a>
</BODY></HTML>
```

이와 같이 Cold Fusion은 SQL과 HTML에 대한 지식만 있으면 데이터 베이스를 이용한 웹 개발을 할 수 있도록 설계되었다. 물론 이와 같은 Cold Fusion을 이용하여 복잡한 응용개발을 지원하기 위해서 <CFSET>, <CFLOOP>, <CFIF> 등의 많은 추가된 태그들을 이용해야 한다. 이와 같이 새로운 태그들을 사용하여 프로그래밍을 하면 베이직을 이용한 ASP 프로그래밍과의 차이가 없어지게 된다.

Allaire사에서 Cold Fusion의 이식성을 지원하기 위해서 거의 모든 웹 서버와 작동할 수 있는 Windows 버전 및 Unix 및 Linux 버전의 Cold Fusion을 공급하고 있어 우수한 이식성을 제공하고 있다. 그러나 개인 개발자가 새로운 기능을 추가 확장하기는 거의 불가능하고 모든 기능 향상은 Allaire사의 개발 팀에 달려 있다는 단점이 있다.

3.2.3 Perl

Perl[13]은 1993년 웹 CGI가 개발되면서부터 CGI 게이트웨이 프로그램의 구현 도구로 늘리 사용되어 왔다. 주로 Unix환경에서 이용되었어나 Win32용 Perl로 이식이 되면서 Windows 환경에서의 웹 개발에도 활발히 사용되고 있다. 본 논문에서는 Apache 웹서버와 ActivePerl[3]을 이용하여 3.1절의 사례를 구현하였다. 똑 같은 구조를 가지나 게이트웨이 프로그램만 add.cgi, query.cgi, modify.cgi, modify2.cgi, modify3.cgi로 Perl로 구현하였다. 다음은 report.cgi의 내용이다.

```
#!D:\Perl\bin\perl.exe # Perl Interpreter의 위치 지정
use OLE; # use Win32::OLE if using the Std Distribution
$Conn = CreateObject OLE "ADODB.Connection";
$Conn->Open("DSN=Sample;UID=;PWD=");
print "Content-type: text/html\n\n";
print '<HTML>';
print "<HEAD><TITLE>List</TITLE></HEAD>\n";
print "<BODY><H1>Data Output</H1>\n";
$RS = $Conn->Execute("SELECT * FROM tblSample");
while ( !$RS->EOF ) {
```

```
    my($NO, $Last, $First, $City) = (
            $RS->Fields('ID')->value,
            $RS->Fields('LastName')->value,
            $RS->Fields('FirstName')->value,
            $RS->Fields('City')->value );
    print $NO, ", ", $Last, ", ", $First, ", ", $City, "<br>\n";
    $RS->MoveNext;
}
$RS->Close;
$Conn->Close;
print '</BODY>';
print '</HTML>';
```

Perl은 문자열 검색과 가공 기능이 우수한 일반 언어로 CGI와 데이터 베이스 연결 기능을 라이브러리를 이용해서 지원하고 있다. 데이터 베이스 연결은 Win32::OLE 모듈을 이용하여 Windows ADO 기능을 Perl에서 사용할 수 있다. 이 코드의 구조는 3.2.1절에서 구현한 report.asp와 같은 구조를 갖는다. 다만 Perl의 모든 변수가 $로 시작하는 것이 특이하다. 그리고 Perl이 일반 언어이기 때문에 모든 HTML 출력은 print문을 통해서 이루어져야 한다. Connection 객체를 만들고 SQL 명령어를 보내고 그 결과인 Recordset을 받아서 레코드 하나 하나씩 필드 값을 확인하고 HTML로 출력하는 것은 ASP와 똑 같다.

다만 Perl에서 지원하는 데이터 베이스 연결 라이브러리는 다양하다. 특정 데이터 베이스의 엔진의 API(Application Programming Interface)를 직접 지원하는 AccessPerl, OraPerl, CISAMPerl 등이 초기에 개발되었다. 이 방식은 데이터 베이스 엔진이 변경되면 변경된 엔진에 맞추어 코드를 다시 작성해야 한다는 단점이 있다. 이를 보안하기 위하여 DBI(Data Base Independent) API를 정하고 특정 엔진으로의 접속은 DBD(Data Base Dependent)로 구현하는 방식이 제안되었다. 본 논문에서는 ODBC를 지원하는 ADO를 사용하므로 DBI를 이용하지 않았다.

폼 입력의 처리는 다음 add.cgi가 cgi-lib.pl 라이브러리를 이용하여 구현한 코드를 보여 준다. 최근 Perl 5에서 지원하는 CGI.pm이 더 다양한 기능을 제공하나 사용이 간단하기 때문에 Perl 4에서 사용되던 cgi-lib.pl을 이용하였다. 이 라이브러리의 &ReadParse 함수는 환경 변수와 표준 입력으로부터 폼 데이터를 받아 이를 %in 의 첨자형 변수에 넘겨주고 이들 첨자형 변수는 폼의 필드 이름과 환경 변수 이름의 키로 하여 각각의 값을 $in{'Firstname'}과 같이 확인할 수 있도록 지원한다.

```
#!D:\Perl\bin\perl.exe # Perl Interpreter의 위치 지정
require "cgi-lib.pl";
use OLE; # use Win32::OLE if using the Std Distribution
$Conn = CreateObject OLE "ADODB.Connection";
$Conn->Open("DSN=Sample;UID=;PWD=");
print "Content-type: text/html\n\n";
print '<HTML>';
&ReadParse(*in);
$Conn->Execute("Insert into tblSample(Firstname, Lastname, City)
values('$in{'Firstname'}','$in{'Lastname'}','$in{'City'}');" );
print 'Record has been added.<br>';
print 'Return to <a href="/~shpark/sample.html">Main Page</a>';
print '</BODY></HTML>';
```

Perl은 새로운 프로그래밍 언어를 습득해야 한다는 부담이 있어나 웹 초창기로부터 개발된 여러 가지 Perl 예제 게이트웨이 프로그램들의 소스가 많이 공개되어 있기 때문에 초급 개발자에게는 좋은 학습도구가 될 수 있다.

Perl은 우수한 이식성를 지원한다. 거의 모든 웹 서버가 Perl CGI를 지원하고 Perl은 모든 운영체계에서 수행이 가능하다. 그리고 기능을 추가하기 위해서는 Perl 5에서 지원하는 오브젝트 모듈과 Perl 4의 라이브러리를 이용하여 개인 개발자의 요구 사항에 맞추어 새로운 모듈을 개발하면 되므로 노력만 하면 용이하게 확장할 수 있다. Perl의 큰 단점은 CGI에 기반을 두고 개

발이 되었으므로 매번 게이트웨이 프로그램이 수행될 때마다 새로운 데이터 베이스 연결을 만들어야 하는 성능상의 문제점을 가지고 있다. 최근에는 이와 같은 단점을 보안해서 modPerl과 같이 Perl 스크립트를 서버에 로드해서 반복하여 사용하고 만든 데이터 베이스 연결을 계속 사용할 수 있도록 하는 새로운 시도가 이루어지고 있다.

3.2.4 PHP

PHP[12]는 최근 공개 소프트웨어 분야에서 관심을 끌고 있는 웹 데이터 베이스 연결 도구이다. PHP는 Perl에 기초하여 그 단점을 보안하여 설계되었다. 본 논문에서는 마이크로소프트사의 PWS 4.0과 PHP 4.01을 이용하여 사례를 구현하였다. 구조는 같으며 게이트웨이 프로그램만 php로 변환하였다. 이들 중 report.php와 add.php를 통해 PHP의 특징을 살펴본다.

```
<HTML><HEAD>
<TITLE>List</TITLE></HEAD>
<BODY>
<H1>Data Output</H1>
<?
//Create and Open Connection Object
$cnx = odbc_connect('sample', '', '');
$cur = odbc_exec( $cnx, 'select * from tblSample' );
while ( odbc_fetch_row($cur) ) {
        $NO = odbc_result( $cur, "ID");
        $A = odbc_result( $cur, "Firstname");
        $B = odbc_result( $cur, "Lastname");
        $C = odbc_result( $cur, "City");
        echo( $NO . ", " . $B . ", " . $A . ", " . $C . "<br>\n");
}
odbc_close( $cnx );
?>
Send Comments to <a href=mailto:yourmail@mail.utexas.edu>Your
Name</a><br>
<a href="Sample.html">Return to Main Menu</a>
</body></html>
```

PHP는 ASP와 같이 인터프리터를 이용하여 <? 태그와 ?> 태그 사이에 포함된 PHP 명령어들을 처리하고 있다. 데이터 베이스 처리는 PHP에 내장된 ODBC 지원 함수를 이용하여 Connection 객체를 만들고 SQL 명령어를 보내며 그 결과를 받아서 하나 하나 처리하는 과정이 ASP와 같다. 다만 출력을 위해서 ASP의 Response 객체 대신 echo 명령어를 이용하는 점이 다르다.

add.php는 PHP에서의 폼 입력 처리 방법을 보여 준다. ASP에서의 ReQuest 객체를 이용하는 방식이 아니라 PHP 인터프리터는 Cold Fusion처럼 각각의 폼 필드에 대해 $필드명 형태의 변수를 생성한다. 환경 변수들로 이와 같은 방법인 $환경변수명으로 환경 변수들의 값을 제공한다.

```
<HTML><BODY>
<?
'Create and Open Connection Object
$cnx = odbc_connect('sample', '', '');
odbc_exec($cnx, "Insert into tblSample(Firstname,
            Lastname, City)" .
            "values('$FirstName','$Lastname',
                    '$City');");
?>
Record has been added.<br>
Return to <a href="sample.html">Main Page</a>
</BODY></HTML>
```

PHP는 공개 소프트웨어인 Apache 서버와 MySQL 데이터 베이스 엔진을 함께 패키지로 제공하고 있어 특히 Linux상에서 널리 사용되고 있다. 본 사례 연구는 Window상에서 이루어져서 MySQL 대신 MS Access를 이용했으므로 ODBC 함수를 이용하였다. 모든 운영 체계에서 지원되는 이식성과 공개 프로그램이라는 특징으로 PHP는 개발용과 교육용으로 많이 사용되고 있다. 복잡한 게이트웨이 프로그램의 개발을 위해서는 Perl과 비슷한 복잡도를 요구하는 PHP 언어에 대한 완전한 이해가 필요하다. 기능 확장을 위해서도 PHP 모듈을 개발할 수

있는 능력이 필요하다.

3.2.5 자바(Java)

객체 지향 언어인 자바가 몇 년 전부터 CGI의 단점을 해결할 수 있는 대안으로 급부상하고 있다. 즉 Servlet[7]이라는 클래스를 정의하여 모든 게이트웨이 프로그램은 이 클래스의 서브 클래스(Subclass)로 정의되도록 하여 Servlet에서 정의한 모든 기능들이 상속되도록 하였다. Servlet 객체는 웹 서버가 구동되면서 동시에 컴퓨터 메모리로 로드가 되고 Servlet에 대한 사용자 요구가 있으면 그때마다 Servlet의 쓰레드(thread)를 생성하여 처리하는 방식이다. 이 때문에 Servlet에서 정의한 데이터베이스로의 연결은 여러 쓰레드에서 공유함으로써 CGI 성능 문제를 해결할 수 있도록 해준다.

이와 함께 지원되는 JSP(Java Server Page)[7]는 ASP와 마찬가지로 HTML과 함께 자바 명령어들을 <% 태그와 %>태그 사이에 포함하여 손쉽게 사용할 수 있도록 해 준다. JSP 컴파일러는 이들 JSP 파일을 읽어 들여 필요한 Servlet 파일로 변환하고 이 변환된 파일을 자바 컴파일러로 컴파일하여 만들어진 바이너리 코드를 웹 서버가 수행하도록 함으로써 개발의 용이함과 함께 성능을 향상시킬 수 있도록 설계되었다.

본 사례 연구에서는 Windows 상에서 수행되는 Apache Tomcat 3.1[2]을 이용하여 3.2.1절에서 ASP로 구현한 코드를 Java의 Servlet 서브 클래스와 JSP코드로 변환하였다. HTML 파일들은 3.1절과 같은 구조를 가지며 게이트웨이 프로그램들만 Servlet의 경우 Report.java, Add.java, Query.java, Modify.java, Modify2.java, Modify3.java로 변환하였고 JSP의 경우는 각각 report.jsp, add.jsp, query.jsp, modify.jsp, modify2.jsp, modify3.jsp로 변환하였다.

3.2.5.1 Servlet

Servlet 게이트웨이 프로그램들 중에서 다음 Report.java를 통해 Servlet을 이용한 프로그래밍을 특징을 살펴보면 먼저 모든 게이트웨이 프로그램은 HttpServlet의 서브 클래스로 정의된다. 이들 Servlet 클래스는 init, doGet, doPost, destory와 같은 메소드 함수들을 정의해 주어야 한다. init과 destroy 메소드는 각 Servlet의 쓰레드가 만들어지고 파괴되는 경우에 처리해야 할 작업을 정의해 주고 doGet과 doPost는 각각 GET 방식과 POST 방식으로 수행되는 경우의 처리를 지정해 준다. Java에서의 데이터 베이스 연결은 JDBC(Java Data Base Connectivity)를 이용하여 ODBC 드라이버를 사용하는 방식을 채택하였다.

```
import java.io.*;
import javax.servlet.*;
import javax.servlet.http.*;
import java.util.*;
import java.sql.*;

public class Report extends HttpServlet {

private Statement stmt = null;
private Connection con = null;
private String URL = "jdbc:odbc:Sample";

public void init( ServletConfig config )
        throws ServletException
{
        super.init ( config );
        try {
            Class.forName( "sun.jdbc.odbc.JdbcOdbcDriver" );
            con = DriverManager.getConnection( URL, "", "" );
            }
        catch ( Exception e ) {
            e.printStackTrace( );
            con = null;
            }
}
public void doPost( HttpServletRequest req, HttpServletResponse res )
        throws ServletException, IOException
{
        doGet(req, res);
}
```

```
public void doGet( HttpServletRequest req, HttpServletResponse res )
        throws ServletException, IOException
{
      int ID;
        String FirstName, LastName, City;
      String query = "select * from tblSample;";
      PrintWriter output = res.getWriter();
      res.setContentType( "text/html;" );
        output.println("<H1>Data Output</H1>");
        try {
        stmt = con.createStatement();
        ResultSet rs = stmt.executeQuery(query);
        while (rs.next()) {
                output.print(rs.getInt("ID")- ", ");
                output.print(rs.getString("LastName") - ", ");
                output.print(rs.getString("FirstName") - ", ");
                output.println(rs.getString("City") - "<br>");
                }
        stmt.close();
        }
        catch ( Exception e ) {
          e.printStackTrace( );
          }
}

public void destroy( )
{
     try {
         con.close( );
     }
     catch ( Exception e ) {
         System.err.println( "Error in closing servlet!" );
          }
}

}
```

Servlet에서의 폼 입력 처리는 다음 Add.java를 보면 알 수 있다. doGet과 doPost 메소드는 HttpServletReQuest와 HttpServletResponse 객체를 매개 변수로 받는다. 폼의 입력은 Http ServletReQuest 타입의 매개변수인 req에 get Parameter 메소드를 이용하여 req.getParamter ("Firstname")와 같이 데이터 값을 가져온다. JDBC 프로그래밍도 ODBC와 같이 Connection 객체를 만들고 여기에 필요한 SQL 명령어를 전달하는 방식으로 처리된다. 여러 가지 에러를 처리하기 위한 Catch 절 때문에 프로그램이 필요 이상 길어졌다. 기본적인 코드는 add.asp와 큰 차이가 나지 않는다.

```
import java.io.*;
import javax.servlet.*;
import javax.servlet.http.*;
import java.util.*;
import java.sql.*;

public class Add extends HttpServlet {

private PreparedStatement stmt = null;
private Connection con = null;
private String URL = "jdbc:odbc:Sample";

public void init( ServletConfig config )
        throws ServletException
{
        super.init ( config );
        try {
            Class.forName( "sun.jdbc.odbc.JdbcOdbcDriver" );
            con = DriverManager.getConnection( URL, "", "" );
            }
        catch ( Exception e ) {
            e.printStackTrace( );
            con = null;
            }
}
public void doGet( HttpServletRequest req, HttpServletResponse res )
        throws ServletException, IOException
{
         doPost(req, res);
}

public void doPost( HttpServletRequest req, HttpServletResponse res )
        throws ServletException, IOException
{
      int ID;
        String FirstName, LastName, City;
        FirstName = req.getParameter("Firstname");
        LastName = req.getParameter("Lastname");
        City = req.getParameter("City");

    String query = "insert into tblSample(LastName, FirstName, City)
values(?,?,?);";
    PrintWriter output = res.getWriter();
    res.setContentType( "text/html" );
        try {
        stmt = con.prepareStatement(query);
        stmt.setString(1, LastName); stmt.setString(2,FirstName);
stmt.setString(3, City);
        stmt.executeUpdate();
        output.println("data inserted");
        stmt.close();
```

```
        }
        catch ( Exception e ) {
          e.printStackTrace( );
          }
}
          .
public void destroy( )
{
    try {
        con.close( );
    }
    catch ( Exception e ) {
        System.err.println( "Error in closing servlet!" );
         }
}
}
```

Servlet 프로그래밍을 위해서는 자바 언어에 대한 이해가 꼭 필요하다. Perl이나 C언어와는 달리 자바는 객체 지향 언어이므로 객체 지향 언어에 대한 이해 없이는 Servlet 프로그래밍이 거의 불가능 할 것으로 보인다. 높은 이식성과 확장성을 지원하는 Servlet이지만 HTML과 SQL에 대한 이해와 함께 자바 객체 언어에 대한 지식이 꼭 필요하다는 것이 초보 웹 개발자에게는 큰 부담이 된다.

3.2.5.2 JSP(Java Server Page)

Servlet의 객체 지향 언어에 대한 부담을 줄이기 위해서 SUN사의 개발자들은 ASP와 같이 HTML 태그와 함께 자바 명령어를 사용할 수 있도록 JSP를 설계하였다. 3.1 절의 사례를 구현한 JSP 코드들 중에서 report.jsp 코드를 보면 report.asp와 거의 같은 구조를 갖는다는 것을 알 수 있다. ASP와 달리 JSP는 변수의 타입의 선언해 주어야 한다. 이에 반해 ASP는 Variant 타입을 지원하고 있다. JSP는 몇 가지 미리 정의된 객체들을 가지고 있다. 폼 입력 처리를 위한 request 객체, 출력을 위한 out 객체들이 이에 속한다.

```
<%@ page import="java.sql.*,java.util.*,
java.io.*,javax.servlet.*,javax.servlet.http.*" %>
<html>
<head><title>List</title></head>
<body><h1>Data Output</h1>
<%
Class.forName("sun.jdbc.odbc.JdbcOdbcDriver").newInstance();
Connection con= DriverManager.getConnection("jdbc:odbc:Sample");
Statement stmt = con.createStatement();
String query = "select * from tblSample;";
ResultSet rs = stmt.executeQuery(query);
while (rs.next()) {
        out.print(rs.getInt("ID")-", ");
        out.print(rs.getString("LastName")-", ");
        out.print(rs.getString("FirstName")- ", ");
        out.println(rs.getString("City") - "<br>");
        } %>
</body>
</html>
```

다음 add.jsp는 폼 입력으로부터 새로운 레코드를 첨가하는 방법을 보여 준다. 모든 입력은 제공된 request 객체를 이용하여 request.getParameter("Firstname")로 확인할 수 있다.

```
<%@ page import="java.sql.*,java.util.*,
java.io.*,javax.servlet.*,javax.servlet.http.*" %>
<html>
<head><title>Insert</title></head>
<body>
<%
Class.forName("sun.jdbc.odbc.JdbcOdbcDriver").newInstance();
Connection con= DriverManager.getConnection("jdbc:odbc:Sample");
String FirstName, LastName, City;
FirstName = request.getParameter("Firstname");
LastName = request.getParameter("Lastname");
City = request.getParameter("City");
String query = "insert into tblSample(LastName, FirstName, City)
values(?,?,?);";
PreparedStatement stmt = con.prepareStatement(query);
stmt.setString(1, LastName);
stmt.setString(2, FirstName);
stmt.setString(3, City);
stmt.executeUpdate();
%>
data inserted
</body>
</html>
```

JSP는 여러 가지 Servlet에서 정의해 주어야 하는 기본적인 메소드와 Exception 처리 등을 지원해 주어 Servlet보다는 훨씬 간단한 프로그

〈표 1〉 웹 데이터 베이스 연결 도구의 비교

도구	사용 언어	추가된 객체	이식성/DB 처리 성능	확장성/분리성
ASP	객체 기반 언어 베이직	ReQuest, Response, Server 의 ADO 객체 등	Unix상에서는 상용 소프트웨어 필요/캐싱을 통한 우수한 성능	ActiveX 객체를 정의해야 함(새로운 환경에 대한 이해가 필요)
Perl	C 언어와 비슷한 Perl	cgi-lib.pl이나 CGI.pm, Win32::OLE 등과 같은 DB 모듈	우수함/ CGI로 인한 제한된 성능	Perl 라이브러리나 모듈을 통한 확장
Cold Fusion	간단한 자체 언어	자체 인터프리터가 모든 기능을 제공	상용으로 우수함/ 상용으로 우수함	없음
PHP	Perl과 비슷한 PHP	PHP 인터프리터 ODBC 인터페이스 폼 처리	대부분 서버를 지원함으로 우수함/캐싱을 지원함	PHP 모듈을 이용한 확장
Servlet	객체 지향 언어	Servlet, JDBC 객체 추가	우수함/우수함	Java를 이용한 높은 확장성
JSP	객체 지향 언어	JSP컴파일러, Servlet, JDBC등이 추가	우수함/우수함	Java Bean을 이용한 분리성 추가

래밍이 가능하다. 그러나 복잡한 게이트웨이 프로그램의 개발을 위해서는 자바 언어에 대한 완전한 이해가 없이는 JSP의 기능을 충분히 활용할 수 없다고 본다.

JSP는 비즈니스 로직을 객체로 구현할 수 있는 틀인 Java Bean과 EJB(Enterprise Java Bean)[7]을 제공한다. 이와 같은 틀을 따르고 원하는 비즈니스 로직을 필요한 데이터베이스를 이용하여 구현하는 비즈니스 객체가 웹 프로그래머들에 의해 제공되면 웹 디자인너는 이들 비즈니스 객체들과 HTML만을 이용하여 웹 프로그래머들의 직접적인 도움이 없이도 사용자 인터페이스를 독립적으로 설계할 수 있게 된다.

3.3 비교 분석 결과

본 논문에서 주소록 사례 연구를 통해 검토한 PC Windows 상에서 제공되는 웹 데이터베이스 연결 도구들의 분석결과는 <표 1>과 같이 정리할 수 있다.

4. 결 론

데이터베이스 게이트웨이 프로그래밍에 대한 교육은 HTML과 SQL에 대한 지식과 몇 개의 Cold Fusion 태그만 소개하면 간단하게 구현해 볼 수 있도록 하는 Cold Fusion이 제일 간단하다. 웹 프로그래밍 초기 교육은 Cold Fusion으로 시작하고 다음 단계에서 ASP, Perl, PHP중에서 하나를 선택하여 객체 기반 언어에서의 웹 프로그래밍 개념을 소개할 수 있겠다. 마지막 단계의 웹 개발 교육에 오래 동안 늘리 사용할 수 있는 개발 도구인 Java Servlet과 JSP를 소개할 수 있겠다.

Servlet 프로그래밍에는 객체 언어 Java에 대한 이해가 먼저 요구되기 때문에 학습 속도는 상당히 떨어질 수 있다. 그러나 적당한 JSP Template을 이용한 JDBC 객체의 사용법에 대한 소개와 request와 out 객체의 사용법에 대한 소개가 함께 이루어지면 Servlet 프로그래밍 기법을 빠르게 교육할 수 있을 것이다.

Perl과 PHP는 C언어에 기초한 프로그래밍 언어들로써 경영정보학을 전공하고 있는 학생들에게는 많이 사용되고 있지 않는 도구들이다. 그러나 이들 공개된 도구들로 구현된 공개 코드가 인터넷에 많이 제공되기 때문에 C언어에 대한 기초 지식이 있는 경우 사용을 강력 추천하

는 바이다. 자바 언어도 기본적인 명령어는 모두 C 언어에서 빌려 온 것으로 경영정보학에서 많이 사용하는 마이크로소프트사의 비주얼 베이직이외에도 C 언어에 대한 학습도 꼭 필요하다고 본다. 마지막으로 비주얼 베이직에 익숙한 개발자의 경우는 ASP가 제일 손쉬운 도구가 될 것이다. 허나 확장성, 이식성, 분리성에서 제한이 있으므로 다른 연결도구에 대한 교육도 필요하다고 본다.

본 논문은 경영정보학을 전공하는 학부생들에게 웹 데이터베이스 연결도구를 교육하는 관점에서 PC Windows상에서 제공되는 여러 연결 도구들을 사용 용이성 측면에서만 비교해 보았다. 교육 목적으로 분석을 한정하다 보니 손쉽게 구할 수 있는 MS Access 데이터베이스만 사용하였고 연결 도구들 중에서 상용인 Oracle Application Server와 Netscape Enterprise Server등에서 지원하는 PL/SQL과 Server Side Java Script을 지원하는 연결 도구들은 포함하지 않았다.

참 고 문 헌

[1] Allaire, Cold Fusion, http://www.allaire.com/, 2000.

[2] Apache Software Foundation, Apache Tomcat 3.1, http://www.apache.org/, 2000.

[3] Active State, ActivePerl 5.6, http://www.activestate.com/, 2000.

[4] Feng L., H. Lu, Integrating database and Web Technologies, International Journal of World Wide Web, Vol.1, No.2, 1998.

[5] Gunther Birznieks G., S. Sol, CGI for Commerce, M&T Books, 1997.

[6] Fedorov A., et al, Professional Active Server Page 2.0, WROX Press, 1998.

[7] Hall M., Servlets and Java Server Pages, Prentice Hall, 2000.

[8] Kalakota R., A.B. Whinston, Electronic Commerce, Addison Wesley, 1997.

[9] Lazar P., P. Holfelder, Web Database Connectivity with Scripting Languages, Web Journal, Vol.2, No.2, 1997.

[10] Liu, J. Peek, R. Jones, B. Buus & A. Nye, Managing Internet Information Services, OReilly & Associates Inc., 1994.

[11] Orfali R., D. Harkey, J. Edwards, The Essential Client/Server Survival Guide, 2nd Ed., John Wiley & Sons, 1996.

[12] PHP Group, PHP Manual, http://www.php.net/, 2000.

[13] Wall L., T. Christiansen, R. Schwartz, Programming Perl 2nd Ed., OReilly & Associates Inc., 1996.

[14] Zhao W., A Study of Web-Based Application Architecture and Performance Measurements, 5th Australian World Wide Web Conference Proceedings, 1999.

저자소개

박 성 헌

서울대학교 금속공학과를 졸업하고, KAIST에서 경영과학 석사, 그리고 퍼듀대학교에서 경영정보학 박사학위를 취득하였다. 현재 명지대대학교 경상대학 지식정보학부 부교수로 재직하고 있으며 주요관심분야는 정보통신기술의 응용분야이다.

박 지 헌

서울대학교 기계설계학과를 졸업하고, KAIST에서 전산학 석사, 그리고 텍사스 주립대학교에서 전산학 박사학위를 취득하였다. 현재 홍익대학교 공과대학 컴퓨터공학과 조교수로 재직하고 있으며 주요관심분야는 컴퓨터 그래픽 분야이다.