웹 인터페이스 개발을 위한 JDBC-ODBC Bridge 기법과 ISAPI 확장 기법에 관한 비교 연구

하창승*

A Comparing Study on the JDBC-ODBC Bridge Method and the ISAPI Extension Method for a Web Interface Development

Chang-seung Ha

본 연구는 2000년도 동명대학 학술 연구조성비로 수행되었습니다.

요 약

웹 기반의 컴퓨터 기술이 보편화되면서, 정보통신 환경은 클라이언트/서버 시스템에서 e-비지니스 시스템으로 점차 바뀌어 가고 있다. 이러한 e-비지니스 시스템은 데이터베이스와 서버사이드 프로그래밍을 통합하기 위한 웹 인터페이스 기술을 요구하고 있다. 전통적으로 CGI 기술이 웹 인터페이스의 표준으로 사용되어 왔으나 연결의 어려움과 처리에 많은 지연 시간을 발생시켰다. 따라서 이러한 문제점을 개선하기 위해 컴퓨터 자원의 사용량을 줄이면서도 빠른 처리가 가능한 ISAPI 방법과 JDBC방법 등이 개발되었다.

본 연구에서는 보다 효과적인 웹 인터페이스 방법을 제안하기 위해 JDBC-ODBC 기법과 ISAPI 확장 기법을 이용한 시스템을 구현하고 구현 방법과 처리시간을 상호 비교한다.

ABSTRACT

As a web technology is generalized, the IT environment has been changed into an e-business system. An e business system requires web interface technology that can be integrated in a database and be applied to a server-side programming. Traditionally, a CGI technology has been used a web interface. But the CGI technology has many difficulties in connection and delay in processing. So, it is necessary to introduce a new methods as ISAPI Extension and JDBC. These methods provide advantages as reduce of memory and fast process.

This paper proposes a comparing study on the JDBC-ODBC Bridge and the ISAPI Extension method. It will suggest more efficient technology with these methods for a web interface development.

* 동명대학 정보통신계열 조교수

Ⅰ. 서 론

최근 정보통신 분야의 시스템 개발 환경은 웹 기반의 프로그래밍 기술이 보편화되면 소프트웨어 개발이 호스트 중심 혹은 다운사이징된 클라이언트/서버 중심에서 인터넷/인트라넷으로 대표되는 웹 기반의 e-비지니스 시스템으로 바뀌어 가고 있다. 또한 전통적인 비즈니스 정보의 액세스가 기존의 표준화된 웹 기술을 통하여 접근할 필요성이 나타나고 있기 때문에 웹 프로그래밍에 대한 요구는 계속 증가될 것이다.[2]

일반적으로 개인이 제공하는 소량의 정보는 웹 서버상에 정적인 HTML 문서형식으로 쉽게 구현할 수 있지만 기업에서 사용하는 정보들의 대부분은 데이터베이스에 저장되고 저장된 데이터를 이용한 동적인 대용량 웹 시스템은 데이터베이스의 연동을 통하여 구현되어야 한다. 그리고 이러한 데이터베이스 연동 기술은 이제까지 서버 사이드 프로그래밍에 필수적인 기술로 인식되어 왔다. 따라서 웹 연동을 실현하기 위한 웹 인터페이스 기술은 데이터베이스에 대한 의존성이 높아질수록 계속 증가될 것이며 웹의 대중성이 증가할수록 더욱 복잡한 양상으로 나타날 것이다. 특히, 웹은 애플리케이션의 전위 모듈로서 많은 장점을 제공할 수 있다. 즉 웹 브라우저는 거의 모든 플랫폼에서 공통적으로 적용될 수 있기 때문에 다중 플랫폼 환경에서 발생할 수 있는 이식성의 문제와 변경된 클라이언트 모듈의 설치 문제를 해결할 수 있다. 더욱이 웹의 하이퍼미디어 기반 모델은 대부분의 사용자에게 친숙하다. 그러므로 웹 데이터베이스 연동 솔루션이 주어지면 웹은 데이터베이스 전위 응용을 개발하기 위한 개방형 플랫폼으로 사용될 수 있다.[1][3]

지금까지 웹 환경에서 보다 다양한 기능과 향상된 분산 환경을 지원하는 많은 웹 연동 기술이 제공되어 왔다. 그 가운데 몇몇 예로 GSQL, Decoux, UMass Information Navigator, DB2WWW, UniWeb 등을 들 수 있는데 이러한 데이터베이스 게이트웨이 연동 기술은 크게 클라이언트 확장 방식과 서버 확장 방식으로 나누어 진다. 클라이언트 확장 방식은 외부 뷰어를 사용하는 방식과 브라우즈 확장 방식으로 다시 나누어지며 서버 확장방식은 CGI 이용방식, API 확장방식, 전용 서버방식으로 나눌 수 있다. 현재까지 개발된 대부분의 데이터베이스 게이트웨이는 CGI 실행 방식을 이용하고 있다.[4]

CGI 기술은 웹 서버와 클라이언트를 연결할 때 사용되던 가장 전통적이며 일반적인 웹 인터페이스 기술이었다. 웹 서버/클라이언트 환경에 CGI 기술의 사용은 플랫폼과 개발 툴에 대한 독립성을 보장하고, 여러 웹 서버에 대한 범용성을 제공한다. CGI 기술은 이제까지 유닉스 운영체제의 웹 프로그램을 작성하기 위한 표준 인터페이스로써 사용되어 왔다. 이것은 대부분의 초기 웹 프로그램들이 유닉스 운영체제에서 개발되었고, 유닉스 운영체제와 마찬가지로 CGI 기술도 클라이언트의 요구에 응답하기 위해 별도의 독립된 프로세스를 사용함으로써 운영체제의 일관성을 유지할 수 있으며 다양한 유닉스 컨텐츠 문서 형식을 위한 공통된 솔루션을 처리할 수 있기 때문이었다. 그러나 CGI 기술은 사용자의 질의를 처리할 때마다 새로운 프로세스를 생성해야하며 생성된 각 프로세스는 데이터베이스에 대한 연결과 로그인 과정을 매번 거쳐야 하기 때문에 처리 성능에 많은 문제가 발생했다.[6][7][12]

본 연구에서는 이러한 문제점을 해결하기 위해 사용의 용이성과 빠른 웹 인터페이스를 제공하는 ISAPI 확장 기법과 플랫폼에 독립적이며 확장성이 뛰어난 JDBC-ODBC Bridge 기법을 제안하고 내부 메커니즘 및 성능 평가를 위해 상호 비교 연구를 수행한다. 이제까지 몇몇 시스템에서 개별적으로 ISAPI 확장 기법과 JDBC 기법을 통한 시스템의 구현이 이루어져 왔으나 두 응용 기술에 대한 실험적이며 체계적인 비교 연구는 없었다. 따라서 본 연구는 웹기반의 스케줄관리 시스템의 구현에서 이 두 기법을 함께 적용하고 보다 효율적인 웹 인터페이스로서 어떠한 환경에서 웹 응용프로그램 구현이 적용 가능한지에 대한 실효성 및 타당성을 검토하여 웹 인터페이스 구현의 선정 기준으로 제시하고자 한다.

Ⅱ. JDBC-ODBC Bridge 기법의 연동 기술

1. JDBC-ODBC Bridge 기법의 기술적 연구

JDBC란 자바를 이용한 데이터베이스 접속과 SQL 문장의 실행 그리고 그 실행결과로부터 얻어진 데이터의 처리 방법과 절차에 관한 프로토콜이다. 이 프로토콜은 하부 데이터베이스와 애플리케이션 사이의 연동

을 위해 표준 API를 이용하기 때문에 특정 데이터베이스와 독립된 프로그램의 구현을 가능하게 한다.[9]

JDBC 기법을 통한 연동은 통합형 SQL 데이터베이스를 처리하는 프레임워크로서 모든 데이터베이스와 독립적으로 자바 애플리케이션을 구현할 수 있는 기반을 제공한다. 즉 JDBC와 연동이 가능한 드라이버와 모듈을 동적으로 적재해 하부 드라이버의 종류에 상관없이 데이터베이스 소스에 액세스할 수 있는데 이 기술은 클래스로 구현된 드라이버 관리자를 통해 직접 제어할 수 있다. 그리고 JDBC는 기존의 ODBC 메커니즘과 비슷한 개념인 SQL CLI(Call Level Interface)를 지원하는 X/Open정의에 기반을 두고 있어 여러 데이터소스를 접근할 수 있는 단일화된 인터페이스를 제공하고 있다. 또한 JDBC는 SQL을 사용해 데이터를 엑세스할 수 있는 함수 라이브러리와 SQL 스트링을 DBMS에게 전달할 수 있는 수단도 제공하고 있다.[11]

JDBC는 크게 JDBC API와 데이터베이스 드라이버로 이루어져 있다. 개발자를 위한 표준 인터페이스의 역할을 수행하는 JDBC API는 ODBC나 기존의 데이터베이스 인터페이스 기술보다 상위 레벨을 구현할 목적으로 고안되었기 때문에 능동적인 데이터베이스의 접근이 가능하다. JDBC에서 사용하는 드라이버는 크게 JDBC-ODBC Bridge 드라이버와 순수 JDBC 드라이버로 나누어진다. 이 드라이버들은 종류에 따라 다른 내부적 구조를 갖고 있기 때문에 어떤 드라이버를 선택하느냐에 따라 성능과 시스템의 구성에 영향을 미친다. 본 연구에서 제안한 JDBC-ODBC Bridge 기법은 JDBC 기법에서 JDBC-ODBC Bridge 드라이버를 채택한 경우이다. JDBC-ODBC Bridge 드라이버는 데이터베이스에 접근하기 위해 기존에 사용하던 방법인 네이티브 메서드에 대신하여 윈도우에서 제공하는 ODBC를 통하여 일관된 방법으로 데이터베이스에 접근할 수 있도록 API를 제공하는 클래스이다. 이것은 기존의 ODBC 기능과 드라이버를 그대로 이용할 수 있고 시스템의 구현과 확장이 용이하며 자바 프로그램의 이식성을 높이기 때문에 일반적으로 많이 활용되고 있는 방법이다. 반면 순수 JDBC 드라이버의 경우는 해당 데이터베이스 공급회사들이 제공하는 JDBC 드라이버를 서버 시스템에 설치해야하는 번거로움은 있지만 기존 DBMS 미들웨어를 그대로 활용할 수 있고 여러 DBMS에 동시에 접근할 수 있는 유연성을 제공한다.[10] 그림 1은 JDBC-ODBC Bridge를 통해 자바 애플리케이션과 여러 데이터베이스들과의 연동 과정을 보여주고 있다.

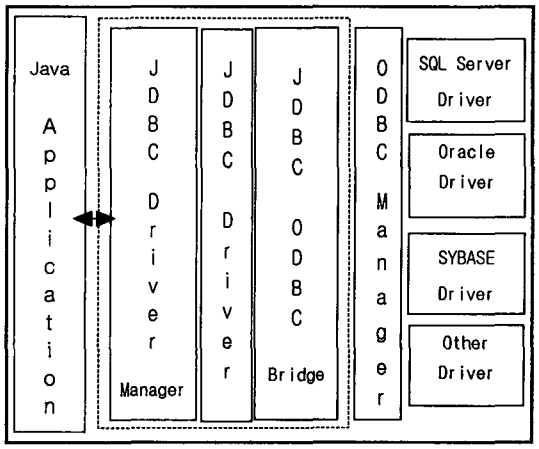

그림 2. JDBC-ODBC Bridge 기법을 이용한 데이터베이스 모델
Fig 1. Database model using a JDBC-ODBC Bridge method

클라이언트가 서버의 데이터베이스에 대해 JDBC 함수를 호출하면 JDBC 드라이버 매니저가 JDBC 드라이버와 ODBC DLL을 이용하여 ODBC 함수호출로 바꾼다. 일단 ODBC 형태로 변환된 함수 호출에 대한 처리는 기존의 ODBC 처리 방법과 같은 과정을 거치는데, 먼저 JDBC 드라이버 매니저에 의해 JDBC 드라이버가 메모리에 적재되면 JDBC-ODBC Bridge와 ODBC 매니저에 의해 데이터베이스와의 연결이 이루어지고 SQL 쿼리에 대한 실행 결과는 ODBC 매니저와 JDBC-ODBC Bridge를 거쳐 사용자에게 다시 보내지게 된다.

2. JDBC-ODBC Bridge 기법의 문제점

JDBC 연동기술에 JDBC-ODBC Bridge 드라이버를 활용하는 방법은 JDBC 프로그램의 구현 및 연동 방법이 단순하고 확장이 용이하기 때문에 자바 애플리케이션 프로그램 개발에 많이 활용되어 왔다. 그러나 이 방법은 기존의 ODBC를 이용하여 모든 상용 데이터베이스에 대한 접근을 보장하기 위해 드라이버 내부 구조를 여러 개의 독립된 계층으로 분리하여 구현하였다. 따라서 데이터 전송시 계층간에 많은 데이터 복사가

진행되고 이것은 데이터 처리에 불필요한 오버헤드를 발생시켜 대용량 데이터베이스와의 인터페이스로는 부적합한 면이 있다. 또한 ODBC와 데이터베이스간의 통신은 소켓(Socket)을 이용하기 때문에 방화벽을 통과할 수 없는 단점도 지니고 있다. 그리고 가장 큰 문제점은 JDBC-ODBC Bridge 드라이버를 JDBC 프로그램의 드라이버로 사용하는 경우에는 특정 웹 서버의 도움이 없으면 단위 애플릿 솔루션으로는 웹을 통한 접근에 많은 어려움을 가진다. 즉 원격의 클라이언트가 데이터베이스와 연동하기 위해서는 클라이언트 측에 ODBC와 관련된 동적 라이브러리가 설치되어 있어야 한다. 클라이언트에 정적인 설치를 대신하여 네트워크를 통한 동적인 설치도 드라이버 자체가 C나 C++과 같은 네이티브(Native) 코드로 되어 있기 때문에 다운로드 자체를 어렵게 한다. 따라서 웹 인터페이스에 JDBC를 이용하기 위해서는 자바코드로 구현된 순수 JDBC 드라이버를 사용하는 것이 유리하다.[5][11]

그러나 JDBC를 이용하는 웹 인터페이스의 구현은 JDBC 클래스에서 제공하는 JDBC API 메소드의 호출 방법과 드라이버 기술방법이 아주 비슷하기 때문에 JDBC-ODBC Bridge 드라이버를 사용하는 애플리케이션 프로그램을 순수 JDBC 드라이버를 사용하는 애플릿으로의 변경은 매우 용이한 편이다.

III. ISAPI 확장 기법의 연동기술

1. ISAPI 확장 기법의 기술적 연구

ISAPI 확장이란 IIS 웹 서버에서 IDC(Internet Database Connector)를 사용하여 웹 문서가 원격의 인터넷 데이터베이스와 상호작용을 할 수 있는 기반 기술을 의미한다. 여기서 확장이란 오라클, SQL 서버와 같은 ODBC 호환 데이터 소스에 접근할 수 있는 ODBC 기술을 의미하며 IIS는 데이터베이스와 웹 문서와의 인터페이스 통로 역할을 수행한다. 데이터베이스의 접근은 IDC라고 불리는 IIS의 컴포넌트를 통하여 이루어지는데 IDC란 ISAPI의 HTTPODBC.DLL 파일을 의미한다. 이것은 웹 문서가 필요로 하는 정보의 종류나 저장된 위치에 상관없이 클라이언트의 요청에 응답할 수 있는 동적인 대화 능력을 갖는 수단을 제공한다. 또한 이 파일은 DLL 파일 형식으로 구성되어 크기가 작고 응용 프로그램들의 요청에 빨리 응답할 수 있다. ISAPI는 데이터베이스에 쉽게 접근하기 위해 내부적으로 ODBC를 사용한다. ODBC를 사용함으로서 클라이언트의 요청이 데이터베이스와 표준화된 방법으로 연동할 수 있으며 SQL 명령의 실행 및 데이터 레코드가 포함된 웹 페이지의 생성을 용이하게 한다. 다음 그림 2는 사용자의 요청을 데이터베이스에 연결하기 위해 필요한 요소들과 요소들 간의 관계를 보여주고 있다.[15]

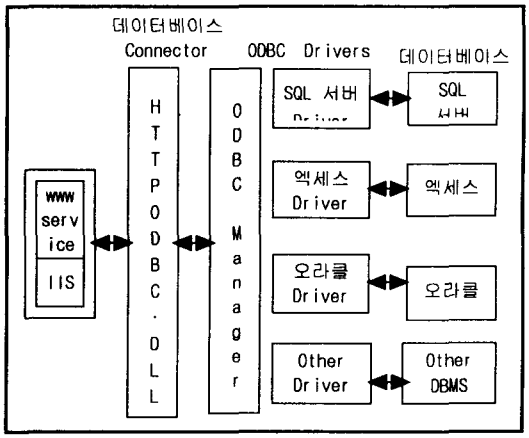

그림 2. ISAPI 확장 기법을 이용한 웹 데이터베이스 모델
Fig 2. Web database model using a ISAPI Extension method

ISAPI 확장 기법의 처리 과정은 그림 3과 같이 여섯 단계로 나누어진다. 먼저 Web 브라우저가 URL을 웹 서버로 보내면 IIS가 URL을 수신하고 IDC를 읽어들인다. IDC는 URL에서 IDC 파일 이름과 그 밖의 항목 정보들을 읽어들이고 IDC 파일에서 ODBC 데이터 소스 정보, HTML 확장 파일명과 SQL 명령문을 읽어들여 DLL내에 포함시킨다. IDC가 ODBC 데이터 원본에 연결되고 IDC 파일에 포함된 SQL 명령문을 DBMS가 실행하면 그 결과를 ODBC를 통하여 IDC에게 전달한다. IDC는 IDC 파일에서 지정된 HTX 파일을 읽어들여 그 파일에 데이터를 병합한 다음 병합된 문서를 다시 IIS로 돌려보내고 IIS는 해당 문서를 다시 클라이언트에 반환하여 브라우즈가 그 처리 결과를 출력한다.[13][14]

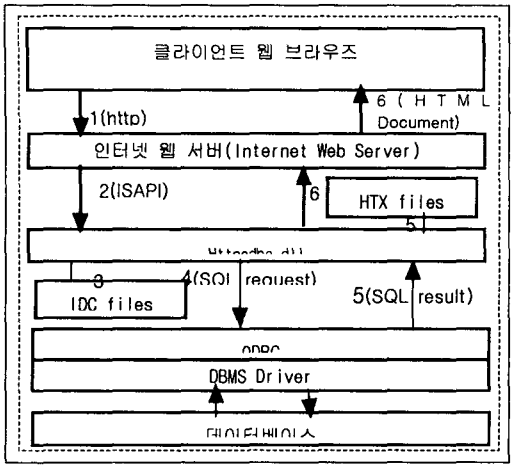

그림 3. ISAPI를 이용한 웹 데이터베이스 처리 과정
Fig 3. Web database processing using a ISAPI

ISAPI 확장은 CGI의 기본 기능을 제공하면서도 클라이언트의 요청을 처리하는데 필요한 시스템 자원을 줄이고 처리 효율을 높이기 위해 고안되었다. ISAPI 확장 기법은 개별적으로 독립된 실행 프로세스 코드를 이용하지 않고 여러 프로세스에서 공유되는 동적 연결 코드를 사용한다. 이것은 한 프로세스가 자신의 실행 코드에 포함되지 않는 함수를 동적 연결코드에서 호출하는 것을 의미한다. 즉 DLL을 사용하여 실행 코드와 동적 연결 코드가 별도로 컴파일 되고 링크 되며 처리 코드가 필요한 경우에는 DLL이 호출되어 그 처리 코드를 필요로 하는 모든 프로세스들에게 DLL 라이브러리 코드를 공유하도록 하여 처리시간을 줄이고 필요한 자원의 요구를 최소한으로 줄일 수 있는 이점을 제공한다.

IV. 웹 인터페이스의 구현 및 비교연구

1. JDBC 기법과 ISAPI 확장기법의 비교

1.1 JDBC 기법과 ISAPI 확장 기법의 구현방법 비교

웹 인터페이스의 구현을 위해 JDBC 기법은 사용자가 자바 애플릿 내에 JDBC 드라이버를 포함시키고 JDBC의 여러 클래스에서 제공하는 JDBC API 메소드를 호출하는 프로그램을 구현해야 하며, ISAPI 확장 기법은 사용자가 ODBC 소스를 생성한 후 IDC파일과 HTML확장 파일을 작성해야 한다. 먼저 웹 서버에 접근하기 위해 두 기법에서 기술하는 HTML 태그의 표현 방법은 표 1과 같다.

표 1. 웹 서버의 접근을 위한 HTML 태그의 표현 방법
Table 1. The HTML presentation for the web server access

JDBC 기법	<applet codebase= "." archive="driver_file.zip" code="class_file" > </applet>
ISAPI 확장 기법	<A HREF = "http://domain-name/scripts/idc_file_name.i dc> hyper link text </A>

JDBC기법의 "driver_file.zip"은 특정 데이터베이스 공급업체에서 제공하는 JDBC 드라이버 파일을 의미한다. 이 파일과 "java_class_file" 파일은 IIS 웹 서버를 사용하는 경우 c:\InterPub\wwwroot 폴더안에 함께 존재해야만 한다. ISAPI 확장 기법에서 "scipts"는 c:\InterPub\scipts 폴더를 의미하며 "idc_file_name.idc"는 사용자가 작성한 IDC 파일이다.

```
Datasource: member_source
Template: member.htx
SQLStatement:
+SELECT id, no from s_cust,
+s_cust1 order by id;
```

그림 4. 데이터 검색을 위한 IDC 파일
Fig 4. The IDC file for data searching

그림 4는 ISAPI 확장 기법에서 데이터를 검색하는 IDC 파일의 내용을 보여주고 있다. "Datasource : member_source" 필드는 오라클 데이터베이스와 연동하기 위해 ODBC 데이터소스의 이름이 member_source임을 나타내고 있으며, "Template : member.htx" 필드는 HTML 확장 파일로서 member.htx 파일을 사용하고 있다는 것을 의미한다. 또한 "SQLStatement : SELECT id, no from s_cust, s_cust1 order by id; " 필드는 데이터베이스 엔진에서 실행될 쿼리이다. 이때 플러스 기호(+)는 SQL문을 여러 라인에 나누어 표시하기 위해 사용한 연결 문자이다.

```
<HTML> <HEAD> </HEAD>
<BODY BGCOLOR='F3F3F3'>
<H2> Tour Schedule </H2>
<TABLE BORDER>
<TR><TH> 관광지명 </TH><TH> 경과시간 </TH></TR>
<%begindetail%>
<TR><TD> <%p_name%> </TD><TD>
<%p_time%> </TD></TR> <%enddetail%>
</TABLE> </BODY> </HTML>
```

그림 5. 데이터 검색 결과를 반환하는 HTML 확장 파일
Fig 5. The HTML extension file returning search result

그림 5는 ISAPI 확장 기법에서 이용되는 HTML 확장 파일로서 IDC 파일과 Scripts 폴더 안에 함께 존재해야 한다. <%p_name%>과 같이 "%%" 기호로 둘러싸인 변수들은 SQL명령의 실행 결과인 데이터베이스의 칼럼 값을 저장하기 위한 매개변수를 의미한다. <%begindetail%>과 <%enddetail%> 같은 요소들은 매개변수들의 존재 위치를 결정하는 범위 지정자(placeholder)이다.

```
static final String connect_string =
"jdbc:oracle:thin:scott/tiger@203.241.121.59:152
1:ORCL";
static final String query = "select p_name,
p_time from s_cust";
  TextArea cutput;
  Connection conn;
  public void run ()
  {
   if (conn == null)
    {
      DriverManager.registerDriver(
        new oracle.jdbc.driver.OracleDriver());
        conr = DriverManager.getConnection
              (connect_string);
    }
    Statement stmt = conn.createStatement();
    ResultSet rset = stmt.executeQuery (query);
    while (rset.next ())
        output.appendText(rset.getString
        ("p_name")+rset.getString("p_time")
         + "Wn");
  }
```

그림 6. 순수 JDBC 드라이버를 사용하는 JDBC 프로그램 예
Fig 6. An example of JDBC program using pure JDBC driver

그림 6은 순수 JDBC 드라이버를 사용하는 자바 애플릿 프로그램의 일부이다. 먼저 JDBC 기법에서 필요한 오라클 드라이버를 로드하기 위해 DriverManger 클래스의 registerDriver() 메서드를 사용하고 있다. 일단 드라이버가 로드되면 프로그램에서 DriverManger 클래스를 통하여 해당 드라이버와의 연결이 시도된다. DriverManger 클래스의 getConnection() 메서드는 실제 데이터베이스와 연결을 담당할 Connection 클래스 객체를 생성하는 역할을 담당하고 있다. 데이터베이스의 테이블에 저장된 데이터를 검색하기 위해 SQL 문장을 ResultSet 클래스의 executeQuery() 메서드를 이용하여 처리한다. SQL 질의문의 결과는 레코드 세트 형식으로 반환되기 때문에 복수 레코드를 반복 처리를 위해 while() 흐름 제어문을 사용한다. 또한 JDBC 드라이버에게 데이터베이스를 인식시키고 연결시키기 위해 다음과 같은 JDBC URL 형식을 사용한다.

```
jdbc:oracle:thin:scott/tiger@203.241.121.59:1521:
ORCL
```

"jdbc"는 URL의 시작 형식으로 프로토콜을 의미한다. "oracle:thin"은 오라클 db_protocol을 의미하며, 이 가운데 "thin"은 오라클 드라이버 지원 타입을 표시한다. 오라클에서는 "thin" 타입 외에 "oci" 타입도 지원하지만 웹에는 사용할 수 없다. "scott/tiger"은 오라클 데이터베이스에 등록된 scott라는 사용자와 비밀번호 tiger를 의미한다. "@203.241.121.59"은 이 IP-주소에 등록된 데이터베이스 기술자로서 오라클 데이터베이스의 "oracle net8 easy config" 라는 툴을 사용하여 해당 얼리어스(alias)를 미리 등록해 주어도 된다. 또한 "1521"은 포트번호를 의미하고 "ORCL"은 오라클 데이터베이스를 설치할 때 등록한 이름이다.

구현 방법적인 측면에서 JDBC 기법과 ISAPI 확장 기법을 비교해 보면 ISAPI 확장 기법의 처리는 기본적으로 IIS 웹서버의 기반에서 IDC라는 확장 라이브러리를 통하여 원격의 데이터베이스와 상호작용을 수행하고 있다. 데이터베이스와 인터페이스 작업의 대부분은 IDC가 담당하고 있기 때문에 외부적으로 많은 부분이 추상화되어 있어 사용자는 IDC와 통신하기 위한 IDC 파일만 구성하면 된다.

반면 JDBC 기법의 구현은 플랫폼에 독립된 운영을 보장하기 위해 특정 데이터베이스 업체에서 제공하는 드라이버를 설치하고 자바 클래스에서 제공하는 메서드를 실행 코드에 사용자가 일일이 기술해 주는 방법을 취하고 있다. 사용의 용이성 측면에서는 ISAPI 확장 기법이 다소 우수한 편이나 특정 웹 서버에 종속된다는 한계성을 지니고 있다.

1.2 JDBC 기법과 ISAPI 확장 기법의 수행시간 비교

본 연구의 실험 환경은 Pentium II Xeon 400MHz CPU와 130MB 메모리를 탑재하고 있는 Compaq SP700 워크스테이션에서 이루어졌다. 웹 연동에 사용된 데이터베이스는 가장 많은 사용자를 확보한 오라클 버전 8.0.5를 사용하였다. 두 기법의 벤치마킹 테스트는 보다 분명한 변별력을 갖도록 하기 위해 충분한 소요 시간이 예상되는 데이터베이스 관계 대수식인 프로젝션 연산과 카티션 프로덕트 연산식을 이용 하였다.

표 2는 JDBC 기법과 ISAPI 확장 기법을 이용하여 오라클 데이터베이스와 연동된 수행시간을 비교한 표이다. 먼저 첫 번째 연산(Π no[R1(1000)])은 1000개의 레코드를 가지고 있는 R1 테이블에서 no라는 칼럼을 기준으로 프로젝션(Projection) 연산을 수행한 결과로서 JDBC 기법에 비해 ISAPI 확장기법이 약 22배정도 빠른 것으로 나타났다. 또한 두 번째 연산(R1(1000) ⓧ R2(5))은 1000개의 레코드를 가지고 있는 R1 테이블과 5개의 레코드를 가지고 있는 R2라는 테이블과의 카티션 프로덕트(Cartesian Product)을 수행한 결과인데, ISAPI 확장 기법은 약 6초 정도가 소요되는데 반해 JDBC 기법은 약 30분 가까운 시간이 소요되었고, 세 번째 연산(R1(1000) ⓧ R2(10))은 R2 테이블의 레코드 수만 10으로 증가시키고 같은 연산을 수행한 결과 ISAPI 확장 기법은 약 11초 정도가 수행되는데 비해 JDBC 기법은 1시간 이상의 처리 시간이 소요되어 처리를 중간에서 중단하였다.

본 실험을 통하여 ISAPI 확장 기법은 데이터의 수가 많아져도 처리시간이 선형적으로 증가되는데 비해 JDBC 기법은 데이터의 수가 증가할수록 처리시간이 폭발적으로 증가하는 것으로 나타났다. 이와 같은 실험 결과는 ISAPI 확장 기법이 웹 연동을 위해 동적 링크 라이버러(DLL) 사용하여 크기가 작고 IIS 서버와 기억공간과 코드를 공유하고 있는 반면에 JDBC 기법은 플랫폼에 독립성을 보장하기 위해 프로세스별로 개별적인 기억 영역과 코드를 전용하고 있기 때문에 ISAPI 기법이 JDBC 기법에 비해 훨씬 빠른 응답 결과를 제공할 수 있는 것으로 보인다.

표 2. JDBC기법과 ISAPI 확장 기법의 수행시간 비교
Table 2. The Comparison of processing time with JDBC and ISAPI extension

방법론 / 연산	Π no[R1(1000)]	R1(1000) ⓧ R2(5)	R1(1000) ⓧ R2(10)
JDBC 기법	44초	29분 57초	-
ISAPI 확장 기법	2초	6초	11초

2. 웹 인터페이스의 구현 및 실험

본 연구에서는 JDBC 기법과 ISAPI 확장 기법을 구현하고 상호 비교하기 위해 두 방법론을 일본 관광지를 경유하는 일정 스케줄관리 프로그램에 적용하였다. 이 프로그램은 웹을 통하여 사용자가 자신이 방문하고자 하는 지방을 선택하면 선택된 지방의 관광 시간과 이동 시간이 합산되어 정해진 시간 안에 당일 관광이 가능한지를 판단하기 위한 웹 프로그램이다. 그림 7은 사용자가 왼편 프레임에서 일본 큐슈지방의 관광지를 선택하면 관광과 이동에 소요된 경과 시간을 합산하여 그 결과를 오른쪽 프레임에 출력하는 화면을 보여주고 있다.

이 프로그램에서는 두 가지 방법론 모두를 적용하였는데 처리 결과의 확인과 비교를 용이하게 하기 위해 별도의 처리 화면을 제시한다. 그림 8은 JDBC 기법을 적용한 경우의 검색 결과를 나타내고 있으며 그림 9는 ISAPI 확장 기법을 사용한 경우의 검색 결과를 나타내고 있다. 두 가지 방법은 구현상의 차이점 때문에 JDBC 기법의 경우는 자바에서 제공하는 "TextArea"라는 컴포넌트를 사용하여 텍스트 패널 상에 결과를 출력하였고 ISAPI 확장 기법의 경우는 "<TABLE>"이라는 HTML 태그를 사용하여 브라우즈 화면에 그 결과를 출력하였다. JDBC 기법의 경우 JDBC 드라이버는 오라클 사에서 제공하는 오라클 JDBC thin 드라이버를 사용하고 있기 때문에 프로그램 실행전에 오라클 리스너(listener)가 실행되고 있는 상태인지 반드시 확인해야 한다.

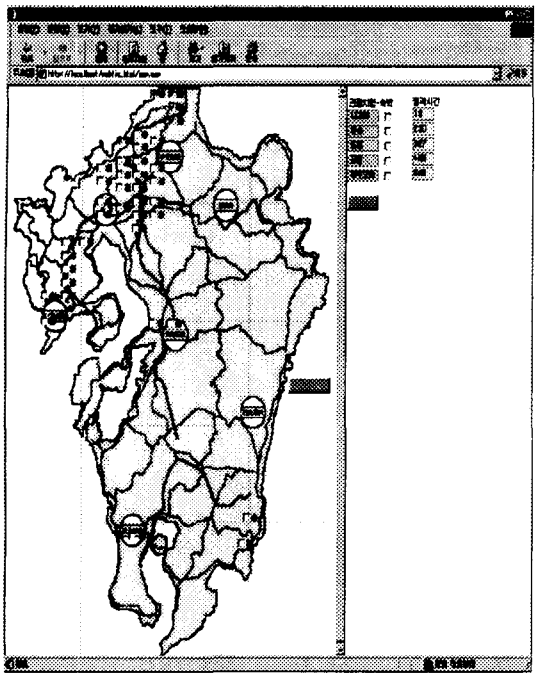

그림 7. 일본 큐슈지방 관광 스케줄관리 화면
Fig 7. Schedule management display for japan kyushu tour

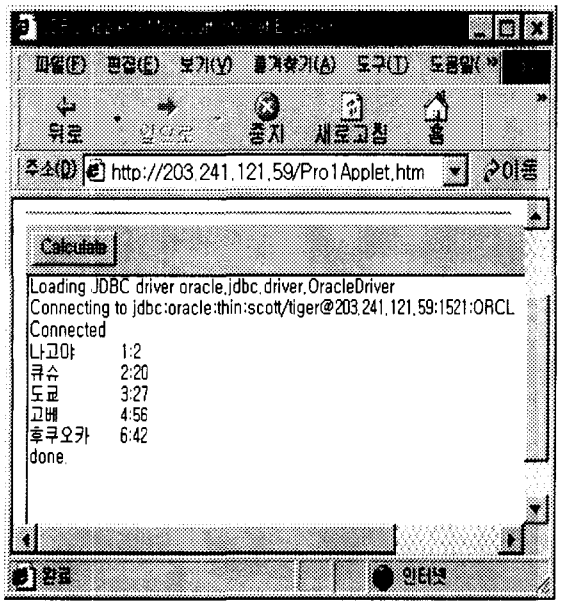

그림 8. JDBC 기법을 통한 데이터베이스 접근화면
Fig 8. Database access display through a JDBC

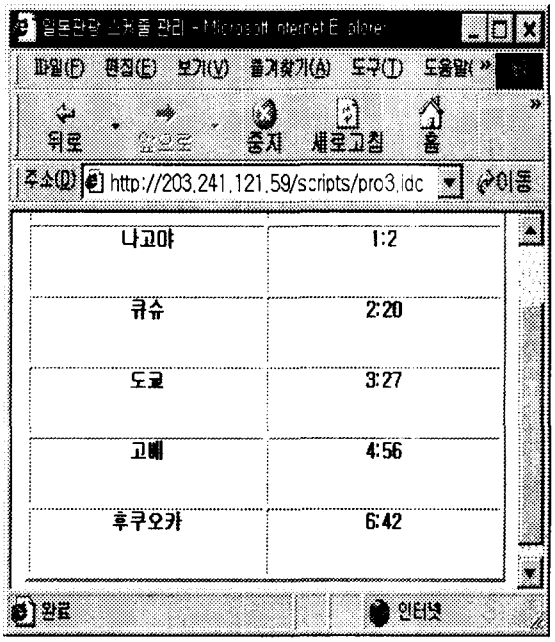

그림 9. ISAPI 확장 기법을 통한 데이터베이스접근화면
Fig 9. Database access display through an ISAPI Extension

V. 결 론

본 연구는 웹 환경에서 기존의 CGI기술 방식 보다 사용의 용이성과 처리속도 면에서 개선된 인터페이스 구현 방법을 제안하기 위해 JDBC-ODBC Bridge 기법과 ISAPI 확장 기법을 상호 비교 하는 것이 목적이었다. 그러나 JDBC-ODBC Bridge 드라이버 통한 JDBC 기법의 구현은 특정 웹 서버의 도움 없이 애플릿 단위로 웹 환경에 접근하는 것이 현실적으로 불가능하다는 사실과 JDBC 기법을 웹 환경에서 구현하기 위해서는 순수 JDBC 드라이버을 사용해야 한다는 사실을 실험을 통해 확인하였다. 또한 JDBC 기법과 ISAPI 확장 기법의 비교에서는 HTTPODBC.DLL을 사용하는 ISAPI 확장 기법이 특정 데이터베이스의 전용 드라이버를 사용하는 JDBC 기법에 비해 구현방법이 다소 용이하고 성능 면에서 10배 이상 빠른 응답시간을 갖는 것으로 확인되었다. 그러나 ISAPI 확장 기법은 특정 웹 서버에 종속적이며 이기종 DB의 동시 지원과 실시간 Result set의 지원이 어려운 반면 JDBC 기법은 다중 플랫폼과 다중 웹 서버의 지원이

가능하였다. 따라서 IIS 웹 서버 상에서 상용 웹 어플리케이션을 개발하는 경우에는 ISAPI 확장 기법을 사용하는 것이 유리한 것으로 생각된다. 반면 비 IIS 웹 서버 환경에서의 웹 어플리케이션 개발은 JDBC 기법이 가능한 대안으로 보인다. 또한 향후 코바(CORBA)와 같은 새로운 웹 인터페이스 환경에서 JDBC 기법이 활용되기 위해서는 보다 빠른 응답 시간을 제공하기 위한 보다 개선된 연구와 새로운 방법론의 제안이 있어야 할 것으로 사료된다.

참고 문헌

[1] 고휘정, "인터넷 웹사이트의 사용자 인터페이스 분석 및 평가에 대한 연구", 홍익대학교 석사논문, 1998.

[2] 구홍서, "www과 데이터베이스 연동기술의조사 분석", 정보과학회지 제18권 제4호, 2000.

[3] 김평철 · 김상욱, "월드와이드웹과 데이터베이스 시스템의 통합", 제2차 www 워크샵, 1995.

[4] 문장원, "WWW 환경에서의 데이터베이스 게이트웨이 설계 및 구현", 한국과학기술원 석사논문, 1996.

[5] 정의현 · 김성진, "자바 2", 도서출판 대림, 2000.

[6] 최유정, "웹 환경에서 데이터베이스관리시스템과의 연동을 이용한 검색시스템개발", 홍익대학교 석사논문, 1998.

[7] 하창승, "ISAPI확장을 이용한 전자상거래시스템의 웹 인터페이스 구현에 관한 연구", 동명논문집 제21권 제1호, 1999.

[8] Genusa(정우경 역), "Special Edition Using ISAPI", 인포북, 1997.

[9] George Reese, "Database Programming with JDBC and JDBC", O'RILLY, 1996.

[10] David Flanagan, "Java examples in a nutshell", O'RILLY, 1997.

[11] http://cs.woosong.ac.kr/study/.www/jdbc.html

[12] Jeffry Dwight & Michael Erwin(김영훈역), "Special Edition Using CGI", 정보문화사, 1996.

[13] Kevin Clements외 2(송혜원, 곽준기 역), "Inside Secrets ISAPI", 삼각형, 1997.

[14] Lab Manual, "Mastering Internet Development with Microsoft Active X Technology", Microsoft, 1998.

[15] Student workbook, "Windows Architecture for Developers", Microsoft, 1997.