「iモード用Javaアプリケーションによる授業情報提供システム構築」の制作と指導

関東ポリテクカレッジ
(関東職業能力開発大学校)　永野　秀浩

1．はじめに

第6回の「ポリテックビジョン」で行われた総合制作・研究の発表において，関東職業能力開発大学校専門課程情報技術科の学生が標記システムの発表を行った。

本来，授業の出席確認は教官が点呼などで行い，単位の履修状況は学生が把握すべきである。しかし，これが機能しない場合，双方に支障が生じる。

また，総合制作の課題として学生自身が携帯電話でのシステムを希望したこともあり，本制作を課題として提示した。今回，この学生に対する指導と問題点について述べる。

2．研究概要

本大学校の専門課程情報技術科のカリキュラムを表1に示す。2年の前期の「生産データベースⅠ,Ⅱ，実習」でデータベースの教科，後期の「生産画像処理，実習」でJava言語を実施している。

また，実習場の環境としては，コンピュータを利用する科目に限られるが，ほぼWindowsドメインに参加する。この情報は，Webブラウザから入手可能となり，Windows2000系のクライアントであれば，ログオンスクリプトで履歴を残すことも可能となる。しかし，LinuxをOSとする履修科目もある。

これらのことから出席情報の入手は，Webブラウザからブラウザに対応した認証方法で行うことが望ましいと判断した。

授業に関する情報の提示は，ほとんどの学生が所有する携帯電話をターゲットとした。現在，高い普及率で市場にあるNTT DoCoMoの携帯電話である503iシリーズ以降では，iモード用Javaアプリケーション（以下iアプリと記す）が動作し，一般ユーザが開発可能である。このiアプリは，携帯端末からWebを参照できるiモードと異なり，携帯端末にJavaプログラムをダウンロードし，情報に変更がない場合，通信の必要はない。緊急の情報であれば，電子メールまたはWebでの提示のほうが現実的であるが，出席数の確認などは通信コストを考慮できるiアプリであっても問題はないと考える。

以上のことを考慮して，授業の情報に関する提示

表1　情報技術科のカリキュラム

前　期	後　期
生産データベースⅠ 生産データベースⅡ 生産データベース実習	生産画像処理 生産画像処理実習

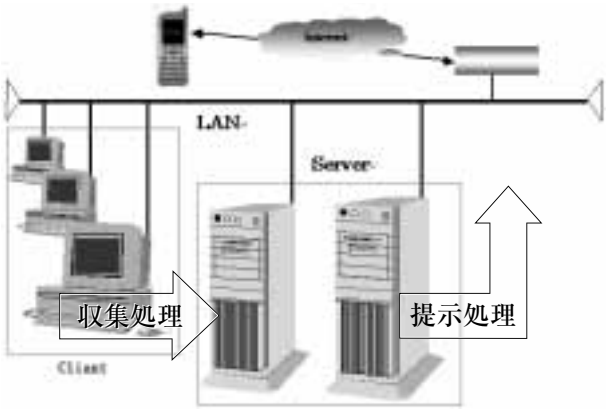


図1　授業情報提供システムの概要

処理は，iアプリで実装し，学習のために提示する環境についてもJava言語での実装をコンセプトとした。

図１にシステムの概要を示すが，データの収集処理と提示処理は，１人の総合制作では負担が大きいと考え別テーマとした。また，データの収集部はWindows系で開発し，提示部はLinux系でのJava言語環境とした。

なお，「ポリテックビジョン」での発表は提示部のみである。

３．システム環境

本データベースシステムの提示部のシステム概要を図２に示す。Linux系OSでのJava言語開発環境としたため，TomcatをWebスクリプトのシステムとして使用した。また，Apache Tomcat 3.3a以降であれば，Webシステムとの連携も考慮されているため，環境の構築は，ほぼセットアップのみで実現できる。

本来，学生に対する指導としてはソースファイルをコンパイルし構築するほうが望ましいと考える。しかし，市場にあるLinuxディストリビューションを理解することも重要であり，残念ながら本校の情報技術科の実習環境では任意のシステムの構築は難しい状況にある。また，総合制作の時間内では指導も難しい。さらに，Linux系にしろ一般にサーバ構築は常に最新の情報が必要となる。

そこで，本提示部のサーバのシステム環境は，以下を指定した。

OS：Red Hat Linux 7.2 FTP版
Webサーバ：Apache 1.3.20-16(Red Hat7.2)
Webスクリプト：Tomcat 3.3a(Servlet2.2/JSP1.1)
DataBabse：PostgreSQL7.1.3(Red Hat7.2)
Java：J2SE1.3.1_02

Tomcatは，ServletおよびJSP（JavaServer Pages）を動作させるソフトウェアである。両ソフトウェアは動的なＨＴＭＬを作成するシステムであるが，ServletはJava言語形式で記述，JSPはHTMLにJavaを埋め込む形式で記述する。

Webシステムやデータベースは，Red Hat7.2にパッケージされているApacheおよびPostgreSQLをそのまま使用し，Java環境はJ2SEのパッケージを使用した。

クライアントの環境は，学生自身がｉアプリの練習で使用した以下のソフトウェアを使用した。

OS：Windows98 Second Edition
Java：J2SE1.3.1_02
iアプリ開発環境：KToolbar
(J2ME Wireless SDK for the Doja r2.2)
iアプリエミュレータ：i-JADE Lite

次に，iアプリの動作環境を図３に示す。通常のJava言語はサン・マイクロシステムズ社が開発したJava仮装マシン（JVM：Ｊ２SEなど）を土台に動作する。しかし，ｉモードで動作するJavaは，KVM（J2ME）等のうえにDoCoMoが独自に定義し携帯用に最適化されたｉモードJava拡張ライブラリ，各端末メーカが定義した機種固有ライブラリを使用して動作している。

また，本校では，公開サーバは現時点で存在しな

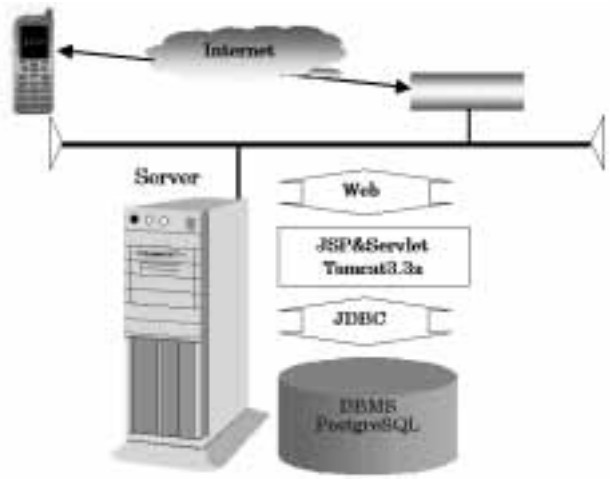


図２　提示部のシステム概要

図３　iアプリの動作環境

いため，iアプリ開発環境は，NTT DoCoMoから無償提供されているDoJa SDKのKToolbarを利用し，携帯端末のシミュレーションで行った。

4．システム動作概要

図4に本システムの提示部の動作概要を示す。携帯端末からの要求に対し，Servletが応答し，処理内容に従って授業情報のデータベースへJDBCで問い合わせを行い，その結果をServletが加工して携帯端末へ応答する仕組みとなる。

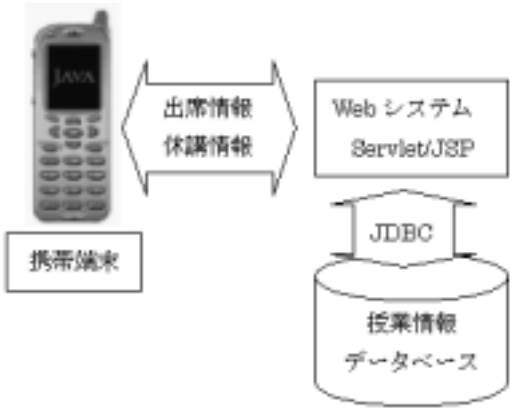


図4　提示部のシステム動作概要

JDBCは，データベースにSQL文の発行を行えるJava のAPIである。JDBCでデータベースへ接続する場合，タイプは4種類あるが，学生への指導は，Microsoft Accessを使用してType1での接続練習を行い，Type4の接続練習をPostgreSQLで行った。

また，生産データベースの授業においてDAO（Data Access Object）やADO（ActiveX Data Objects）については実施し，総合制作実習に当たって，ASP（Active Server Pages）についても実施しているため比較的スムーズに理解できるようであった。

ただし，TelnetやFTPを使用してソースファイルのコンパイル，更新およびサービスの再起動などを行うため，当初はスムーズにいかない面もあった。

次にデータベースの構造を図5に示す。フリーのデータベースサーバであるPostgreSQLはバージョンが7.0から外部キー，7.1から外部結合（OUTER JOIN）などがサポートされ，SQL92の重要な機能はサポートしていると考えられる。

本システムの全体構成としては，Oracle8iR8.1.6やMicrosoft SQL Server7.0で同様のデータベースを構築し，分散管理や開発効率を検討したいと考えているが，今回の提示部の学生への指導はPostgreSQLを用いた。図5は紙面での見やすさを考慮してSQL Serverのダイアグラムを示す。

図5　授業情報データベースの構造

4.1　サーバ側の構築

リスト1に休講テーブル（KyukouT）と出席テーブル（SyussekiT）の作成方法を示す。CONSTRAINT句で外部キーを指定している。

```
CREATE TABLE KyukouT (
        Hiduke              DATE,
    KamokuID                INTEGER
     CONSTRAINT cnstKyukouT_KamokuT
      REFERENCES KamokuT,
        PRIMARY KEY (Hiduke,KamokuID)
);
CREATE TABLE SyussekiT (
        Hiduke              DATE,
        Jigen               INTEGER,
        StudentID VARCHAR(5)
      CONSTRAINT cnstSyussekiT_StudentT
       REFERENCES StudentT,
        KamokuID                    INTEGER
      CONSTRAINT cnstSyussekiT_KamokuT
       REFERENCES KamokuT,
        FLG                 INTEGER,
        IPAddr              VARCHAR(15)
          CONSTRAINT cnstSyussekiT_MachineT
    REFERENCES MachineT,
        Memo                VARCHAR(50),
        PRIMARY KEY (Hiduke,Jigen,StudentID)
);
```

リスト1　CREATE TABLE文の例

リスト2に時間割を表示するVIEWを示す。外部結合の履修(RisyuuT)はLEFT JOINで時間割(TimeT)中のNULL要素を表示することができる。また，教官情報（TeacherT）も同様の理由でRIGHT JOINである。ただし，科目（KamokuT）はNULL要素のデータをもつため等結合（INNER JOIN）である。

```
DROP VIEW v_time;

CREATE VIEW v_time AS
SELECT
   TimeT.Tourokubi,TimeT.Nennji,TimeT.Unyou,
   TimeT.Youbi,TimeT.Jigen,KamokuT.Kamoku,
   TeacherT.Myouji,KamokuT.KamokuID,TeacherT.TeacherID
FROM
   TeacherT RIGHT JOIN
      (KamokuT INNER JOIN
         (TimeT LEFT JOIN RisyuuT
            ON TimeT.KamokuID = RisyuuT.KamokuID)
         ON KamokuT.KamokuID = TimeT.KamokuID)
      ON TeacherT.TeacherID = RisyuuT.TeacherID
ORDER BY
    TimeT.Tourokubi,TimeT.Nennji,TimeT.Unyou,
    TimeT.Youbi,TimeT.Jigen;
```

リスト2　VIEWでの外部結合（OUTER JOIN）の例

参考としてOracleの場合，リスト3に示すように外部結合はWHERE句のフィールドに(+)記号で表現する。日本語もほぼ問題なく指定できる。

```
DROP VIEW v_時間割;

CREATE VIEW v_時間割 AS
SELECT
 A.登録日,A.年次,A.運用,A.曜日,A.時限,B.科目,D.略称
FROM
   時間割T A,科目T B,履修T C,教官T D
WHERE A.科目ID = B.科目ID
 AND A.科目ID = C.科目ID(+)
 AND C.教官ID = D.教官ID(+)
ORDER BY A.登録日,A.年次,A.運用,A.曜日,A.時限;
```

リスト3　Oracleでの外部結合（OUTER JOIN）の例

また，リスト4にリレーションの例を示す。

```
DROP VIEW v_kyukou;

CREATE VIEW v_kyukou AS
   SELECT A.Hiduke,B.Kamoku
   FROM KyukouT A,KamokuT B
   WHERE A.KamokuID=B.KamokuID;
```

リスト4　休講情報のVIEWの例

リスト5にJSPでのプログラム例を示す。文字コードやJavaのパッケージを<%@ page～%>で指定し，<%!～%>には初期値，<%～%>内にコードを記述する。JDBCのドライバはTomcatのライブラリ位置にコピーすればよく，サービスの設定はデータベースに従う。

JDBC関連部分をリスト内の網掛けで示す。

また，outがJSPのデフォルトで用意されているHTMLの出力となるオブジェクトである。

```
<%@ page language="java"
  contentType="text/html;charset=EUC_JP"
  import="java.sql.*" %>
<%!
final String JDBCDriver = "org.postgresql.Driver";
final String url = "jdbc:postgresql:jyugyou";
%>
<html><head><title>Time Table</title></head>
<center>時間割情報<br><hr>
<body background="img/mback.gif">
   ・・・
<%
String query ="select youbi as 週,・・・
   ・・・
Connection con;
Statement stmt;
try {
   Class.forName(JDBCDriver);
} catch(java.lang.ClassNotFoundException e) {
   out.println("ClassNotFoundException:"+e.getMessage());
}
try {
   con = DriverManager.getConnection(url, "postgres", "");
   stmt = con.createStatement();
   ResultSet rst = stmt.executeQuery(cquery);
   rst.next();
   if(rst.getInt("cnt")==0){
      out.println("データがありません<br>");
   } else {
      ResultSet rs = stmt.executeQuery(query);
      ResultSetMetaData rsmd = rs.getMetaData();
      int cntCols = rsmd.getColumnCount();
      out.println("<table border=1><tr>");
      for(int i=1;i<=cntCols;i++) {
         String colName = rsmd.getColumnName(i);
         out.print("<th>" + colName + "</th>");
      }
      out.println("</tr>");
      int cntRow=0;
      while (rs.next()) {
         out.println("<tr>");
         if(cntRow==0){
            out.print("<td rowspan=4>"+
               wn[rs.getInt(1)]+"</td>");
         }
         for(int i=2;i<=cntCols;i++) {
            String s = rs.getString(i);
            if(s!=null && s.length()>0) {
               if(i!=3){
                  out.print("<td align=center>"+s+"</td>");
               } else {
                  out.print("<td>" + s + "</td>");
               }
            } else {
               out.print("<td>    </th>");
            }
         }
         out.println("</tr>");
         if(++cntRow>=cntCols) cntRow=0;
      }
      out.println("</table>");
   }
   stmt.close();
   con.close();
} catch(SQLException e) {
   out.println("SQLException: " + e.getMessage());
}
%>
</center></body></html>
```

リスト5　JSPのプログラム例

4.2　クライアント側の構築

iアプリでWebページの情報を入手するためには，標準Javaのjava.net.HttpURLConnectionと多少異なるがそのサブセットとみなせるHttpConnectionインタフェースを使用する。また，スクラッチパッドにデータを保存することで，一度サーバから受信したデータを保存することができる。スクラッチパッドは以下の形式で指定できる。

scratchpad:///0;pos=バイト数

リスト6にiアプリでURL情報をスクラッチパッドに格納する部分を示す。表示に関してはスクラッチパッドから任意の部分を取得すればよいことになる。

```
public void GetText(){
   try{
      txtB.setText("ダウンロード中…");

      HttpConnection httpC = null;
      long contL;

      httpC = (HttpConnection)
            Connector.open(url,Connector.READ,true);
      httpC.setRequestMethod(HttpConnection.GET);
      httpC.connect();

      contL = httpC.getLength();
      byte recvData[] = new byte[(int)contL];

      InputStream inpSt = httpC.openInputStream();
      inpSt.read(recvData);

      inpSt.close();
      httpC.close();

OutputStream os =
   Connector.openOutputStream("scratchpad:///0");

      os.write(recvData);

      os.close();

      txtB.setText("ダウンロード完了");
   }catch(IOException e){}
}
```

リスト6　ダウンロードの例

授業の出席数の情報は，科目ごとのカウントのみ保存すればよい。このためには，JSPではなくServletで記述し，クライアントの受信データにHTMLのタグなどが含まれないようにする。

また，JSPで「contentType="text/plain;"」とする方法もあるが，Servletで記述する場合，必ずコンパイル操作とTomcatの再起動の操作が必要となるため，学生の指導に注意が必要となった。

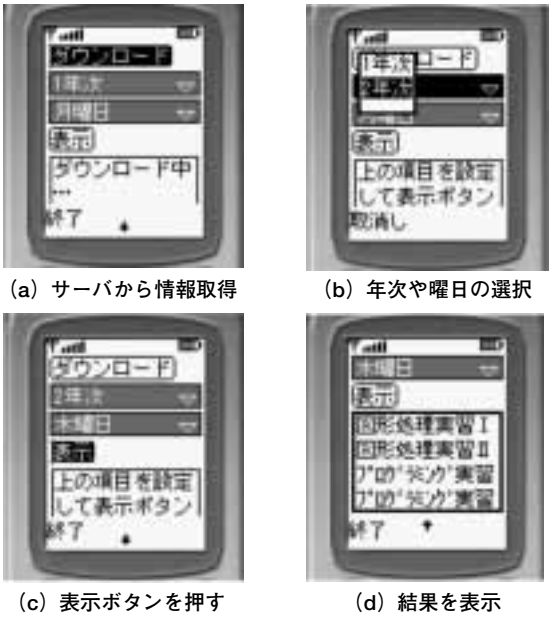
リスト7にServletでの例を示す。出力オブジェクトのoutはJSPと合わせた変数名にしている。また，出力する文字バイト数をスクラッチパッドから取得しやすいように固定長としている。

```
/*時間割を表示するサーブレット*/
import java.io.IOException;
import java.io.PrintWriter;
import java.io.OutputStreamWriter;
import java.lang.String;

import javax.servlet.ServletException;
import javax.servlet.http.HttpServlet;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;

import java.sql.*;

public class outtext extends HttpServlet{
   PrintWriter out;
   final String JDBCDriver = "org.postgresql.Driver";
   final String url = "jdbc:postgresql:cur";
   Connection con;
   Statement stmt;
   StringBuffer res;
   String query =
      "select jugyouS,tantouS from jikanwari;";
   public void doGet(HttpServletRequest
               rq,HttpServletResponse rp)
               throws IOException,ServletException{
      out = new PrintWriter(new OutputStreamWriter(
                        rp.getOutputStream(),"EUC_JP"));

      printOuttext();

      rp.setContentLength(res.toString().length());
      out.println(res.toString().length());
      rp.setContentType("text/plain");
      out.println(""+res.toString());
      out.close();
   }
   private void printOuttext(){
      try{
         Class.forName(JDBCDriver);
      }catch(java.lang.ClassNotFoundException e){
         out.println("ClassNotFoundException: " +
                              e.getMessage());
      }
      try{
         con = DriverManager.getConnection(
                           url,"postgres","postgres");
         stmt= con.createStatement();
         res = new StringBuffer();
         ResultSet rs = stmt.executeQuery(query);

         while(rs.next()){
            String s = rs.getString ("jugyouS");
            String p = rs.getString ("tantouS");
            int n=s.length();
            res.append(s);
            for(int i=n;i<=15;i++){res.append(" ");}
            n=res.length();
            res.append(p);
            for(int i=n;i<=30;i++){res.append(" ");}
            res.append("\n");
            stmt.close();
            con.close();
         }
      }catch(SQLException e){
         out.println("SQLException:" + e.getMessage());
      }
   }
}
```

リスト7　Servletの例

5．システム動作例

図6に，実用面としてはiアプリではなくiモードのほうが現実的であるが，実行例としてKtoolbarシミュレータでの時間割情報の取得結果を示す。



(a) サーバから情報取得　(b) 年次や曜日の選択

(c) 表示ボタンを押す　(d) 結果を表示

図6　時間割情報取得の実行例

6．現状の問題点と今後の課題

iアプリの仕様に関しては，プログラムサイズが10kバイト以内，記憶領域は5kバイト以内（メーカによる拡張あり）という制限がある。この制限からクライアントにいかに多くの情報を少ないデータ量で配信するかについての検討が必要である。また，液晶で表示できるサイズは120×120ドット程度であるため，表示する内容を必要最小限でわかりやすいものに工夫する必要がある。

本システムの問題点としては，提示部をiアプリでの作成としているが，データの取得に関しては，Webブラウザからページをクリックし,ドメイン情報からデータベースへログ形式で取得する。このため，時間割に合わせて重複した学生の判別およびマシンのIPから不正登録を区別する仕様である。このメカニズムはWindowsドメインにのみ有効であり，LinuxをOSとした授業では別の方法が必要となる。

また，実習場を利用する科目のみ対象となるため，携帯端末での登録も考えられるが一律に学生に強制はできない。

さらに，今回の環境として学習という観点から，Java言語ベースでの作成をコンセプトとしたが，実用性となれば，MySQLなどの処理の早いデータベースとPerl CGIまたはPHPのほうが優れる。また，日本語の扱いに関してはPostgreSQLでの指定とJavaでのコード体系も関連するがShift-JISでの機種依存コードに対しては変換テーブルなどが必要になる。これら日本語に関しては，Oracle8iやMicrosoft SQL Serverのほうが有利である。特にPostgreSQLでフィールド名を日本語にした場合，JDBCでのSQLの指定でもコード変換が必要になる。

最後に最大の問題点として，iアプリでのPOST要求がサーバに反映されずGET要求しか動作しない現象がある。シミュレータの問題の可能性もあるが，現状，学生分の応答ファイルを用意し，データベースからの情報を返す方法で対応している。

7．おわりに

本校の専門課程に転任し1年目であったが学生主体の総合研究は難しく，一部仕様を限定した形で各学生に分担した。本システムは4名が個別に情報の入手部と提示部を作成し，提示部に関してはデータベースとスクリプトをOracle8i+PHP,SQL Server+ASPそして今回示したPostgreSQL+Tomcatで行った。

結果として，1年目はそれぞれを作成する段階にとどまり，データベースの学習要素および各環境での問題点をシステムを通して理解させることはできていない。次期学生では統合したいと考えているが，要素が多くなる危惧もある。

最後に，情報技術科に実験回線として外部への公開も可能となるため，シミュレーションではなく実機で動作を確認したい。

<注>

＊）図3 iアプリの動作環境はNTT DoCoMoの「iモード対応Javaコンテンツ開発ガイド～詳細編～第1.1版」から引用